

Monoidal Categories

**Monoidal and monoidal (co)closed
categories**

2026.05-02

22 May 2026

**Mohamed Barakat
Sebastian Gutsche
Sebastian Posur
Tom Kuhmichel
Fabian Zickgraf**

Mohamed Barakat Email: mohamed.barakat@uni-siegen.de
Homepage: <http://algebra.mathematik.uni-siegen.de/barakat/>
Address: Walter-Flex-Str. 3
57068 Siegen
Germany

Sebastian Gutsche Email: sebastian.gutsche@gmail.com
Homepage: <http://algebra.mathematik.uni-siegen.de/gutsche/>
Address: Department Mathematik
Universität Siegen
Walter-Flex-Straße 3
57068 Siegen
Germany

Sebastian Posur Email: sebastian.posur@uni-muenster.de
Homepage: <http://algebra.mathematik.uni-siegen.de/posur/>
Address: Department Mathematik
Universität Siegen
Walter-Flex-Straße 3
57068 Siegen
Germany

Tom Kuhmichel Email: tom.kuhmichel@student.uni-siegen.de
Homepage: <https://github.com/TKuh>
Address: Department Mathematik
Universität Siegen
Walter-Flex-Straße 3
57068 Siegen
Germany

Fabian Zickgraf Email: f.zickgraf@dashdos.com
Homepage: <https://github.com/zickgraf/>
Address: Walter-Flex-Str. 3
57068 Siegen
Germany

Contents

1	Monoidal Categories	3
1.1	Monoidal Categories	3
1.2	Additive Monoidal Categories	7
1.3	Braided Monoidal Categories	8
1.4	Symmetric Monoidal Categories	9
1.5	Left Closed Monoidal Categories	9
1.6	Closed Monoidal Categories	16
1.7	Left Coclosed Monoidal Categories	25
1.8	Coclosed Monoidal Categories	32
1.9	Symmetric Closed Monoidal Categories	40
1.10	Symmetric Coclosed Monoidal Categories	41
1.11	Rigid Symmetric Closed Monoidal Categories	41
1.12	Rigid Symmetric Coclosed Monoidal Categories	43
1.13	Convenience Methods	46
1.14	Add-methods	46
2	Examples and Tests	107
2.1	Test functions	107
3	Code Generation for Monoidal Categories	117
3.1	Monoidal Categories	117
3.2	Closed Monoidal Categories	117
3.3	Coclosed Monoidal Categories	118
4	The terminal category with multiple objects	119
4.1	Constructors	119
4.2	GAP Categories	119
5	Legacy Operations and Synonyms	120
5.1	Legacy operations	120
5.2	Synonyms for legacy operations	121
6	MonoidalCategories automatic generated documentation	124
6.1	MonoidalCategories automatic generated documentation of properties	124
	Index	126

Chapter 1

Monoidal Categories

1.1 Monoidal Categories

A 6-tuple $(\mathbf{C}, \otimes, 1, \alpha, \lambda, \rho)$ consisting of

- a category \mathbf{C} ,
- a functor $\otimes : \mathbf{C} \times \mathbf{C} \rightarrow \mathbf{C}$ compatible with the congruence of morphisms,
- an object $1 \in \mathbf{C}$,
- a natural isomorphism $\alpha_{a,b,c} : a \otimes (b \otimes c) \cong (a \otimes b) \otimes c$,
- a natural isomorphism $\lambda_a : 1 \otimes a \cong a$,
- a natural isomorphism $\rho_a : a \otimes 1 \cong a$,

is called a *monoidal category*, if

- for all objects a, b, c, d , the pentagon identity holds:

$$(\alpha_{a,b,c} \otimes \text{id}_d) \circ \alpha_{a,b \otimes c,d} \circ (\text{id}_a \otimes \alpha_{b,c,d}) \sim \alpha_{a \otimes b,c,d} \circ \alpha_{a,b,c \otimes d},$$

- for all objects a, c , the triangle identity holds:

$$(\rho_a \otimes \text{id}_c) \circ \alpha_{a,1,c} \sim \text{id}_a \otimes \lambda_c.$$

The corresponding GAP property is given by `IsMonoidalCategory`.

1.1.1 TensorProductOnMorphisms (for IsCapCategoryMorphism, IsCapCategory-Morphism)

▷ `TensorProductOnMorphisms(alpha, beta)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, a' \otimes b')$

The arguments are two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$. The output is the tensor product $\alpha \otimes \beta$.

1.1.2 TensorProductOnMorphismsWithGivenTensorProducts (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `TensorProductOnMorphismsWithGivenTensorProducts(s, alpha, beta, r)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, a' \otimes b')$

The arguments are an object $s = a \otimes b$, two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$, and an object $r = a' \otimes b'$. The output is the tensor product $\alpha \otimes \beta$.

1.1.3 TensorProductOnMorphismAndObject (for IsCapCategoryMorphism, IsCapCategoryObject)

▷ `TensorProductOnMorphismAndObject(alpha, b)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, a' \otimes b)$

The arguments are a morphism $\alpha : a \rightarrow a'$ and an object b . The output is the tensor product $\alpha \otimes \text{id}_b$.

1.1.4 TensorProductOnMorphismAndObjectWithGivenTensorProducts (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject, IsCapCategoryObject)

▷ `TensorProductOnMorphismAndObjectWithGivenTensorProducts(s, alpha, b, r)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, a' \otimes b)$

The arguments are an object $s = a \otimes b$, a morphism $\alpha : a \rightarrow a'$, an object b and an object $r = a' \otimes b$. The output is the tensor product $\alpha \otimes \text{id}_b$.

1.1.5 TensorProductOnObjectAndMorphism (for IsCapCategoryObject, IsCapCategoryMorphism)

▷ `TensorProductOnObjectAndMorphism(a, beta)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, a \otimes b')$

The arguments are an object a and a morphism $\beta : b \rightarrow b'$. The output is the tensor product $\text{id}_a \otimes \beta$.

1.1.6 TensorProductOnObjectAndMorphismWithGivenTensorProducts (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `TensorProductOnObjectAndMorphismWithGivenTensorProducts(s, a, beta, r)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, a \otimes b')$

The arguments are an object $s = a \otimes b$, an object a , a morphism $\beta : b \rightarrow b'$ and an object $r = a \otimes b'$. The output is the tensor product $\text{id}_a \otimes \beta$.

1.1.7 AssociatorRightToLeft (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ AssociatorRightToLeft(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(a \otimes (b \otimes c), (a \otimes b) \otimes c)$.

The arguments are three objects a, b, c . The output is the associator $\alpha_{a,(b,c)} : a \otimes (b \otimes c) \rightarrow (a \otimes b) \otimes c$.

1.1.8 AssociatorRightToLeftWithGivenTensorProducts (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ AssociatorRightToLeftWithGivenTensorProducts(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(a \otimes (b \otimes c), (a \otimes b) \otimes c)$.

The arguments are an object $s = a \otimes (b \otimes c)$, three objects a, b, c , and an object $r = (a \otimes b) \otimes c$. The output is the associator $\alpha_{a,(b,c)} : a \otimes (b \otimes c) \rightarrow (a \otimes b) \otimes c$.

1.1.9 AssociatorLeftToRight (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ AssociatorLeftToRight(a, b, c) (operation)

Returns: a morphism in $\text{Hom}((a \otimes b) \otimes c \rightarrow a \otimes (b \otimes c))$.

The arguments are three objects a, b, c . The output is the associator $\alpha_{(a,b),c} : (a \otimes b) \otimes c \rightarrow a \otimes (b \otimes c)$.

1.1.10 AssociatorLeftToRightWithGivenTensorProducts (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ AssociatorLeftToRightWithGivenTensorProducts(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}((a \otimes b) \otimes c \rightarrow a \otimes (b \otimes c))$.

The arguments are an object $s = (a \otimes b) \otimes c$, three objects a, b, c , and an object $r = a \otimes (b \otimes c)$. The output is the associator $\alpha_{(a,b),c} : (a \otimes b) \otimes c \rightarrow a \otimes (b \otimes c)$.

1.1.11 LeftUnitor (for IsCapCategoryObject)

▷ LeftUnitor(a) (attribute)

Returns: a morphism in $\text{Hom}(1 \otimes a, a)$

The argument is an object a . The output is the left unitor $\lambda_a : 1 \otimes a \rightarrow a$.

1.1.12 LeftUnitorWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftUnitorWithGivenTensorProduct(a, s) (operation)

Returns: a morphism in $\text{Hom}(1 \otimes a, a)$

The arguments are an object a and an object $s = 1 \otimes a$. The output is the left unitor $\lambda_a : 1 \otimes a \rightarrow a$.

1.1.13 LeftUnitorInverse (for IsCapCategoryObject)

▷ LeftUnitorInverse(a) (attribute)

Returns: a morphism in $\text{Hom}(a, 1 \otimes a)$

The argument is an object a . The output is the inverse of the left unitor $\lambda_a^{-1} : a \rightarrow 1 \otimes a$.

1.1.14 LeftUnitorInverseWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftUnitorInverseWithGivenTensorProduct(a, r) (operation)

Returns: a morphism in $\text{Hom}(a, 1 \otimes a)$

The argument is an object a and an object $r = 1 \otimes a$. The output is the inverse of the left unitor $\lambda_a^{-1} : a \rightarrow 1 \otimes a$.

1.1.15 RightUnitor (for IsCapCategoryObject)

▷ RightUnitor(a) (attribute)

Returns: a morphism in $\text{Hom}(a \otimes 1, a)$

The argument is an object a . The output is the right unitor $\rho_a : a \otimes 1 \rightarrow a$.

1.1.16 RightUnitorWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject)

▷ RightUnitorWithGivenTensorProduct(a, s) (operation)

Returns: a morphism in $\text{Hom}(a \otimes 1, a)$

The arguments are an object a and an object $s = a \otimes 1$. The output is the right unitor $\rho_a : a \otimes 1 \rightarrow a$.

1.1.17 RightUnitorInverse (for IsCapCategoryObject)

▷ RightUnitorInverse(a) (attribute)

Returns: a morphism in $\text{Hom}(a, a \otimes 1)$

The argument is an object a . The output is the inverse of the right unitor $\rho_a^{-1} : a \rightarrow a \otimes 1$.

1.1.18 RightUnitorInverseWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject)

▷ RightUnitorInverseWithGivenTensorProduct(a, r) (operation)

Returns: a morphism in $\text{Hom}(a, a \otimes 1)$

The arguments are an object a and an object $r = a \otimes 1$. The output is the inverse of the right unitor $\rho_a^{-1} : a \rightarrow a \otimes 1$.

1.1.19 TensorProductOnObjects (for IsCapCategoryObject, IsCapCategoryObject)

▷ TensorProductOnObjects(a, b) (operation)

Returns: an object

The arguments are two objects a, b . The output is the tensor product $a \otimes b$.

1.1.20 TensorUnit (for IsCapCategory)

▷ `TensorUnit(C)` (attribute)

Returns: an object

The argument is a category C . The output is the tensor unit 1 of C .

1.2 Additive Monoidal Categories

1.2.1 LeftDistributivityExpanding (for IsCapCategoryObject, IsList)

▷ `LeftDistributivityExpanding(a, L)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes (b_1 \oplus \dots \oplus b_n), (a \otimes b_1) \oplus \dots \oplus (a \otimes b_n))$

The arguments are an object a and a list of objects $L = (b_1, \dots, b_n)$. The output is the left distributivity morphism $a \otimes (b_1 \oplus \dots \oplus b_n) \rightarrow (a \otimes b_1) \oplus \dots \oplus (a \otimes b_n)$.

1.2.2 LeftDistributivityExpandingWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ `LeftDistributivityExpandingWithGivenObjects(s, a, L, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = a \otimes (b_1 \oplus \dots \oplus b_n)$, an object a , a list of objects $L = (b_1, \dots, b_n)$, and an object $r = (a \otimes b_1) \oplus \dots \oplus (a \otimes b_n)$. The output is the left distributivity morphism $s \rightarrow r$.

1.2.3 LeftDistributivityFactoring (for IsCapCategoryObject, IsList)

▷ `LeftDistributivityFactoring(a, L)` (operation)

Returns: a morphism in $\text{Hom}((a \otimes b_1) \oplus \dots \oplus (a \otimes b_n), a \otimes (b_1 \oplus \dots \oplus b_n))$

The arguments are an object a and a list of objects $L = (b_1, \dots, b_n)$. The output is the left distributivity morphism $(a \otimes b_1) \oplus \dots \oplus (a \otimes b_n) \rightarrow a \otimes (b_1 \oplus \dots \oplus b_n)$.

1.2.4 LeftDistributivityFactoringWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ `LeftDistributivityFactoringWithGivenObjects(s, a, L, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = (a \otimes b_1) \oplus \dots \oplus (a \otimes b_n)$, an object a , a list of objects $L = (b_1, \dots, b_n)$, and an object $r = a \otimes (b_1 \oplus \dots \oplus b_n)$. The output is the left distributivity morphism $s \rightarrow r$.

1.2.5 RightDistributivityExpanding (for IsList, IsCapCategoryObject)

▷ `RightDistributivityExpanding(L, a)` (operation)

Returns: a morphism in $\text{Hom}((b_1 \oplus \dots \oplus b_n) \otimes a, (b_1 \otimes a) \oplus \dots \oplus (b_n \otimes a))$

The arguments are a list of objects $L = (b_1, \dots, b_n)$ and an object a . The output is the right distributivity morphism $(b_1 \oplus \dots \oplus b_n) \otimes a \rightarrow (b_1 \otimes a) \oplus \dots \oplus (b_n \otimes a)$.

1.2.6 RightDistributivityExpandingWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject, IsCapCategoryObject)

▷ RightDistributivityExpandingWithGivenObjects(s, L, a, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = (b_1 \oplus \cdots \oplus b_n) \otimes a$, a list of objects $L = (b_1, \dots, b_n)$, an object a , and an object $r = (b_1 \otimes a) \oplus \cdots \oplus (b_n \otimes a)$. The output is the right distributivity morphism $s \rightarrow r$.

1.2.7 RightDistributivityFactoring (for IsList, IsCapCategoryObject)

▷ RightDistributivityFactoring(L, a) (operation)

Returns: a morphism in $\text{Hom}((b_1 \otimes a) \oplus \cdots \oplus (b_n \otimes a), (b_1 \oplus \cdots \oplus b_n) \otimes a)$

The arguments are a list of objects $L = (b_1, \dots, b_n)$ and an object a . The output is the right distributivity morphism $(b_1 \otimes a) \oplus \cdots \oplus (b_n \otimes a) \rightarrow (b_1 \oplus \cdots \oplus b_n) \otimes a$.

1.2.8 RightDistributivityFactoringWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject, IsCapCategoryObject)

▷ RightDistributivityFactoringWithGivenObjects(s, L, a, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = (b_1 \otimes a) \oplus \cdots \oplus (b_n \otimes a)$, a list of objects $L = (b_1, \dots, b_n)$, an object a , and an object $r = (b_1 \oplus \cdots \oplus b_n) \otimes a$. The output is the right distributivity morphism $s \rightarrow r$.

1.3 Braided Monoidal Categories

A monoidal category \mathbf{C} equipped with a natural isomorphism $B_{a,b} : a \otimes b \cong b \otimes a$ is called a *braided monoidal category* if

- $\lambda_a \circ B_{a,1} \sim \rho_a$,
- $(B_{c,a} \otimes \text{id}_b) \circ \alpha_{c,a,b} \circ B_{a \otimes b, c} \sim \alpha_{a,c,b} \circ (\text{id}_a \otimes B_{b,c}) \circ \alpha_{a,b,c}^{-1}$,
- $(\text{id}_b \otimes B_{c,a}) \circ \alpha_{b,c,a}^{-1} \circ B_{a,b \otimes c} \sim \alpha_{b,a,c}^{-1} \circ (B_{a,b} \otimes \text{id}_c) \circ \alpha_{a,b,c}$.

The corresponding GAP property is given by IsBraidedMonoidalCategory.

1.3.1 Braiding (for IsCapCategoryObject, IsCapCategoryObject)

▷ Braiding(a, b) (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, b \otimes a)$.

The arguments are two objects a, b . The output is the braiding $B_{a,b} : a \otimes b \rightarrow b \otimes a$.

1.3.2 BraidingWithGivenTensorProducts (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ BraidingWithGivenTensorProducts(s, a, b, r) (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, b \otimes a)$.

The arguments are an object $s = a \otimes b$, two objects a, b , and an object $r = b \otimes a$. The output is the braiding $B_{a,b} : a \otimes b \rightarrow b \otimes a$.

1.3.3 BraidingInverse (for IsCapCategoryObject, IsCapCategoryObject)

▷ BraidingInverse(a, b) (operation)

Returns: a morphism in $\text{Hom}(b \otimes a, a \otimes b)$.

The arguments are two objects a, b . The output is the inverse braiding $B_{a,b}^{-1} : b \otimes a \rightarrow a \otimes b$.

1.3.4 BraidingInverseWithGivenTensorProducts (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ BraidingInverseWithGivenTensorProducts(s, a, b, r) (operation)

Returns: a morphism in $\text{Hom}(b \otimes a, a \otimes b)$.

The arguments are an object $s = b \otimes a$, two objects a, b , and an object $r = a \otimes b$. The output is the inverse braiding $B_{a,b}^{-1} : b \otimes a \rightarrow a \otimes b$.

1.4 Symmetric Monoidal Categories

A braided monoidal category \mathbf{C} is called *symmetric monoidal category* if $B_{a,b}^{-1} \sim B_{b,a}$. The corresponding GAP property is given by `IsSymmetricMonoidalCategory`.

1.5 Left Closed Monoidal Categories

A monoidal category \mathbf{C} which has for each functor $- \otimes b : \mathbf{C} \rightarrow \mathbf{C}$ a right adjoint (denoted by $\underline{\text{Hom}}_\ell(b, -)$) is called a *left closed monoidal category*.

If no operations involving left duals are installed manually, the left dual objects will be derived as $a^\vee := \underline{\text{Hom}}_\ell(a, 1)$.

The corresponding GAP property is called `IsLeftClosedMonoidalCategory`.

1.5.1 LeftInternalHomOnObjects (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftInternalHomOnObjects(a, b) (operation)

Returns: an object

The arguments are two objects a, b . The output is the internal hom object $\underline{\text{Hom}}_\ell(a, b)$.

1.5.2 LeftInternalHomOnMorphisms (for IsCapCategoryMorphism, IsCapCategoryMorphism)

▷ LeftInternalHomOnMorphisms(alpha, beta) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}_\ell(a', b), \underline{\text{Hom}}_\ell(a, b'))$

The arguments are two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$. The output is the internal hom morphism $\underline{\text{Hom}}_\ell(\alpha, \beta) : \underline{\text{Hom}}_\ell(a', b) \rightarrow \underline{\text{Hom}}_\ell(a, b')$.

1.5.3 LeftInternalHomOnMorphismsWithGivenLeftInternalHoms (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryMorphism, IsCapCategoryObject)

▷ LeftInternalHomOnMorphismsWithGivenLeftInternalHoms(s, alpha, beta, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = \underline{\text{Hom}}_\ell(a', b)$, two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$, and an object $r = \underline{\text{Hom}}_\ell(a, b')$. The output is the internal hom morphism $\underline{\text{Hom}}_\ell(\alpha, \beta) : \underline{\text{Hom}}_\ell(a', b) \rightarrow \underline{\text{Hom}}_\ell(a, b')$.

1.5.4 LeftClosedMonoidalEvaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalEvaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}_\ell(a, b) \otimes a, b)$.

The arguments are two objects a, b . The output is the evaluation morphism $\text{ev}_{a,b} : \underline{\text{Hom}}_\ell(a, b) \otimes a \rightarrow b$, i.e., the counit of the tensor hom adjunction.

1.5.5 LeftClosedMonoidalEvaluationMorphismWithGivenSource (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalEvaluationMorphismWithGivenSource(a, b, s) (operation)

Returns: a morphism in $\text{Hom}(s, b)$.

The arguments are two objects a, b and an object $s = \underline{\text{Hom}}_\ell(a, b) \otimes a$. The output is the evaluation morphism $\text{ev}_{a,b} : \underline{\text{Hom}}_\ell(a, b) \otimes a \rightarrow b$, i.e., the counit of the tensor hom adjunction.

1.5.6 LeftClosedMonoidalCoevaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalCoevaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(b, \underline{\text{Hom}}_\ell(a, b \otimes a))$.

The arguments are two objects a, b . The output is the coevaluation morphism $\text{coev}_{a,b} : b \rightarrow \underline{\text{Hom}}_\ell(a, b \otimes a)$, i.e., the unit of the tensor hom adjunction.

1.5.7 LeftClosedMonoidalCoevaluationMorphismWithGivenRange (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalCoevaluationMorphismWithGivenRange(a, b, r) (operation)

Returns: a morphism in $\text{Hom}(b, r)$.

The arguments are two objects a, b and an object $r = \underline{\text{Hom}}_\ell(a, b \otimes a)$. The output is the coevaluation morphism $\text{coev}_{a,b} : b \rightarrow \underline{\text{Hom}}_\ell(a, b \otimes a)$, i.e., the unit of the tensor hom adjunction.

1.5.8 TensorProductToLeftInternalHomAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ TensorProductToLeftInternalHomAdjunctMorphism(a, b, f) (operation)

Returns: a morphism in $\text{Hom}(a, \underline{\text{Hom}}_\ell(b, c))$.

The arguments are two objects a, b and a morphism $f : a \otimes b \rightarrow c$. The output is a morphism $g : a \rightarrow \underline{\text{Hom}}_\ell(b, c)$ corresponding to f under the tensor hom adjunction.

1.5.9 TensorProductToLeftInternalHomAdjunctMorphismWithGivenLeftInternalHom (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ TensorProductToLeftInternalHomAdjunctMorphismWithGivenLeftInternalHom(a, b, f, i) (operation)

Returns: a morphism in $\text{Hom}(a, i)$.

The arguments are two objects a, b , a morphism $f : a \otimes b \rightarrow c$ and an object $i = \underline{\text{Hom}}_\ell(b, c)$. The output is a morphism $g : a \rightarrow \underline{\text{Hom}}_\ell(b, c)$ corresponding to f under the tensor hom adjunction.

1.5.10 LeftInternalHomToTensorProductAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ LeftInternalHomToTensorProductAdjunctMorphism(b, c, g) (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, c)$.

The arguments are two objects b, c and a morphism $g : a \rightarrow \underline{\text{Hom}}_\ell(b, c)$. The output is a morphism $f : a \otimes b \rightarrow c$ corresponding to g under the tensor hom adjunction.

1.5.11 LeftInternalHomToTensorProductAdjunctMorphismWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ LeftInternalHomToTensorProductAdjunctMorphismWithGivenTensorProduct(b, c, g, t) (operation)

Returns: a morphism in $\text{Hom}(t, c)$.

The arguments are two objects b, c , a morphism $g : a \rightarrow \underline{\text{Hom}}_\ell(b, c)$ and an object $t = a \otimes b$. The output is a morphism $f : a \otimes b \rightarrow c$ corresponding to g under the tensor hom adjunction.

1.5.12 LeftClosedMonoidalPreComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalPreComposeMorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}_\ell(a, b) \otimes \underline{\text{Hom}}_\ell(b, c), \underline{\text{Hom}}_\ell(a, c))$.

The arguments are three objects a, b, c . The output is the precomposition morphism $\text{LeftClosedMonoidalPreComposeMorphism}_{a,b,c} : \underline{\text{Hom}}_\ell(a, b) \otimes \underline{\text{Hom}}_\ell(b, c) \rightarrow \underline{\text{Hom}}_\ell(a, c)$.

1.5.13 LeftClosedMonoidalPreComposeMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalPreComposeMorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{Hom}}_\ell(a, b) \otimes \underline{\text{Hom}}_\ell(b, c)$, three objects a, b, c , and an object $r = \underline{\text{Hom}}_\ell(a, c)$. The output is the precomposition morphism $\text{LeftClosedMonoidalPreComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{Hom}}_\ell(a, b) \otimes \underline{\text{Hom}}_\ell(b, c) \rightarrow \underline{\text{Hom}}_\ell(a, c)$.

1.5.14 LeftClosedMonoidalPostComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalPostComposeMorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}_\ell(b, c) \otimes \underline{\text{Hom}}_\ell(a, b), \underline{\text{Hom}}_\ell(a, c))$.

The arguments are three objects a, b, c . The output is the postcomposition morphism $\text{LeftClosedMonoidalPostComposeMorphism}_{a,b,c} : \underline{\text{Hom}}_\ell(b, c) \otimes \underline{\text{Hom}}_\ell(a, b) \rightarrow \underline{\text{Hom}}_\ell(a, c)$.

1.5.15 LeftClosedMonoidalPostComposeMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalPostComposeMorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{Hom}}_\ell(b, c) \otimes \underline{\text{Hom}}_\ell(a, b)$, three objects a, b, c , and an object $r = \underline{\text{Hom}}_\ell(a, c)$. The output is the postcomposition morphism $\text{LeftClosedMonoidalPostComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{Hom}}_\ell(b, c) \otimes \underline{\text{Hom}}_\ell(a, b) \rightarrow \underline{\text{Hom}}_\ell(a, c)$.

1.5.16 LeftDualOnObjects (for IsCapCategoryObject)

▷ LeftDualOnObjects(a) (attribute)

Returns: an object

The argument is an object a . The output is its dual object a^\vee .

1.5.17 LeftDualOnMorphisms (for IsCapCategoryMorphism)

▷ LeftDualOnMorphisms(α) (attribute)

Returns: a morphism in $\text{Hom}(b^\vee, a^\vee)$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is its dual morphism $\alpha^\vee : b^\vee \rightarrow a^\vee$.

1.5.18 LeftDualOnMorphismsWithGivenLeftDuals (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ LeftDualOnMorphismsWithGivenLeftDuals(s, α, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The argument is an object $s = b^\vee$, a morphism $\alpha : a \rightarrow b$, and an object $r = a^\vee$. The output is the dual morphism $\alpha^\vee : b^\vee \rightarrow a^\vee$.

1.5.19 LeftClosedMonoidalEvaluationForLeftDual (for IsCapCategoryObject)

▷ LeftClosedMonoidalEvaluationForLeftDual(a) (attribute)

Returns: a morphism in $\text{Hom}(a^\vee \otimes a, 1)$.

The argument is an object a . The output is the evaluation morphism $\text{ev}_a : a^\vee \otimes a \rightarrow 1$.

1.5.20 LeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct(s , a , r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = a^\vee \otimes a$, an object a , and an object $r = 1$. The output is the evaluation morphism $\text{ev}_a : a^\vee \otimes a \rightarrow 1$.

1.5.21 MorphismToLeftBidual (for IsCapCategoryObject)

▷ MorphismToLeftBidual(a) (attribute)

Returns: a morphism in $\text{Hom}(a, (a^\vee)^\vee)$.

The argument is an object a . The output is the morphism to the bidual $a \rightarrow (a^\vee)^\vee$.

1.5.22 MorphismToLeftBidualWithGivenLeftBidual (for IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismToLeftBidualWithGivenLeftBidual(a , r) (operation)

Returns: a morphism in $\text{Hom}(a, r)$.

The arguments are an object a , and an object $r = (a^\vee)^\vee$. The output is the morphism to the bidual $a \rightarrow (a^\vee)^\vee$.

1.5.23 TensorProductLeftInternalHomCompatibilityMorphism (for IsList)

▷ TensorProductLeftInternalHomCompatibilityMorphism($list$) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}_\ell(a, a') \otimes \underline{\text{Hom}}_\ell(b, b'), \underline{\text{Hom}}_\ell(a \otimes b, a' \otimes b'))$.

The argument is a list of four objects $[a, a', b, b']$. The output is the natural morphism $\text{TensorProductLeftInternalHomCompatibilityMorphism}_{a, a', b, b'} : \underline{\text{Hom}}_\ell(a, a') \otimes \underline{\text{Hom}}_\ell(b, b') \rightarrow \underline{\text{Hom}}_\ell(a \otimes b, a' \otimes b')$.

1.5.24 TensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ TensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects(s , $list$, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are a list of four objects $[a, a', b, b']$, and two objects $s = \underline{\text{Hom}}_\ell(a, a') \otimes \underline{\text{Hom}}_\ell(b, b')$ and $r = \underline{\text{Hom}}_\ell(a \otimes b, a' \otimes b')$. The output is the natural morphism $\text{TensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects}_{a, a', b, b'} : \underline{\text{Hom}}_\ell(a, a') \otimes \underline{\text{Hom}}_\ell(b, b') \rightarrow \underline{\text{Hom}}_\ell(a \otimes b, a' \otimes b')$.

1.5.25 TensorProductLeftDualityCompatibilityMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ TensorProductLeftDualityCompatibilityMorphism(a , b) (operation)

Returns: a morphism in $\text{Hom}(a^\vee \otimes b^\vee, (a \otimes b)^\vee)$.

The arguments are two objects a, b . The output is the natural morphism $\text{TensorProductLeftDualityCompatibilityMorphism} : a^\vee \otimes b^\vee \rightarrow (a \otimes b)^\vee$.

1.5.26 **TensorProductLeftDualityCompatibilityMorphismWithGivenObjects** (for **IsCapCategoryObject**, **IsCapCategoryObject**, **IsCapCategoryObject**, **IsCapCategoryObject**)

▷ $\text{TensorProductLeftDualityCompatibilityMorphismWithGivenObjects}(s, a, b, r)$ (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = a^\vee \otimes b^\vee$, two objects a, b , and an object $r = (a \otimes b)^\vee$. The output is the natural morphism $\text{TensorProductLeftDualityCompatibilityMorphismWithGivenObjects}_{a,b} : a^\vee \otimes b^\vee \rightarrow (a \otimes b)^\vee$.

1.5.27 **MorphismFromTensorProductToLeftInternalHom** (for **IsCapCategoryObject**, **IsCapCategoryObject**)

▷ $\text{MorphismFromTensorProductToLeftInternalHom}(a, b)$ (operation)

Returns: a morphism in $\text{Hom}(a^\vee \otimes b, \underline{\text{Hom}}_\ell(a, b))$.

The arguments are two objects a, b . The output is the natural morphism $\text{MorphismFromTensorProductToLeftInternalHom}_{a,b} : a^\vee \otimes b \rightarrow \underline{\text{Hom}}_\ell(a, b)$.

1.5.28 **MorphismFromTensorProductToLeftInternalHomWithGivenObjects** (for **IsCapCategoryObject**, **IsCapCategoryObject**, **IsCapCategoryObject**, **IsCapCategoryObject**)

▷ $\text{MorphismFromTensorProductToLeftInternalHomWithGivenObjects}(s, a, b, r)$ (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = a^\vee \otimes b$, two objects a, b , and an object $r = \underline{\text{Hom}}_\ell(a, b)$. The output is the natural morphism $\text{MorphismFromTensorProductToLeftInternalHomWithGivenObjects}_{a,b} : a^\vee \otimes b \rightarrow \underline{\text{Hom}}_\ell(a, b)$.

1.5.29 **IsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit** (for **IsCapCategoryObject**)

▷ $\text{IsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit}(a)$ (attribute)

Returns: a morphism in $\text{Hom}(a^\vee, \underline{\text{Hom}}_\ell(a, 1))$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit}_a : a^\vee \rightarrow \underline{\text{Hom}}_\ell(a, 1)$.

1.5.30 **IsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject** (for **IsCapCategoryObject**)

▷ $\text{IsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject}(a)$ (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}_\ell(a, 1), a^\vee)$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject}_a : \underline{\text{Hom}}_\ell(a, 1) \rightarrow a^\vee$.

1.5.31 UniversalPropertyOfLeftDual (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ UniversalPropertyOfLeftDual(t , a , α) (operation)

Returns: a morphism in $\text{Hom}(t, a^\vee)$.

The arguments are two objects t, a , and a morphism $\alpha : t \otimes a \rightarrow 1$. The output is the morphism $t \rightarrow a^\vee$ given by the universal property of a^\vee .

1.5.32 LeftClosedMonoidalLambdaIntroduction (for IsCapCategoryMorphism)

▷ LeftClosedMonoidalLambdaIntroduction(α) (attribute)

Returns: a morphism in $\text{Hom}(1, \underline{\text{Hom}}_\ell(a, b))$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is the corresponding morphism $1 \rightarrow \underline{\text{Hom}}_\ell(a, b)$ under the tensor hom adjunction.

1.5.33 LeftClosedMonoidalLambdaElimination (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ LeftClosedMonoidalLambdaElimination(a , b , α) (operation)

Returns: a morphism in $\text{Hom}(a, b)$.

The arguments are two objects a, b , and a morphism $\alpha : 1 \rightarrow \underline{\text{Hom}}_\ell(a, b)$. The output is a morphism $a \rightarrow b$ corresponding to α under the tensor hom adjunction.

1.5.34 IsomorphismFromObjectToLeftInternalHom (for IsCapCategoryObject)

▷ IsomorphismFromObjectToLeftInternalHom(a) (attribute)

Returns: a morphism in $\text{Hom}(a, \underline{\text{Hom}}_\ell(1, a))$.

The argument is an object a . The output is the natural isomorphism $a \rightarrow \underline{\text{Hom}}_\ell(1, a)$.

1.5.35 IsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ IsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom(a , r) (operation)

Returns: a morphism in $\text{Hom}(a, r)$.

The argument is an object a , and an object $r = \underline{\text{Hom}}_\ell(1, a)$. The output is the natural isomorphism $a \rightarrow \underline{\text{Hom}}_\ell(1, a)$.

1.5.36 IsomorphismFromLeftInternalHomToObject (for IsCapCategoryObject)

▷ IsomorphismFromLeftInternalHomToObject(a) (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}_\ell(1, a), a)$.

The argument is an object a . The output is the natural isomorphism $\underline{\text{Hom}}_\ell(1, a) \rightarrow a$.

1.5.37 IsomorphismFromLeftInternalHomToObjectWithGivenLeftInternalHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ IsomorphismFromLeftInternalHomToObjectWithGivenLeftInternalHom(a , s) (operation)

Returns: a morphism in $\text{Hom}(s, a)$.

The argument is an object a , and an object $s = \underline{\text{Hom}}_\ell(1, a)$. The output is the natural isomorphism $\underline{\text{Hom}}_\ell(1, a) \rightarrow a$.

1.6 Closed Monoidal Categories

A monoidal category \mathbf{C} which has for each functor $- \otimes b : \mathbf{C} \rightarrow \mathbf{C}$ a right adjoint (denoted by $\underline{\text{Hom}}_\ell(b, -)$) is called a *closed monoidal category*.

If no operations involving duals are installed manually, the dual objects will be derived as $a^\vee := \underline{\text{Hom}}_\ell(a, 1)$.

The corresponding GAP property is called `IsClosedMonoidalCategory`.

1.6.1 InternalHomOnObjects (for IsCapCategoryObject, IsCapCategoryObject)

▷ `InternalHomOnObjects(a, b)` (operation)

Returns: an object

The arguments are two objects a, b . The output is the internal hom object $\underline{\text{Hom}}(a, b)$.

1.6.2 InternalHomOnMorphisms (for IsCapCategoryMorphism, IsCapCategoryMorphism)

▷ `InternalHomOnMorphisms(alpha, beta)` (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a', b), \underline{\text{Hom}}(a, b'))$

The arguments are two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$. The output is the internal hom morphism $\underline{\text{Hom}}(\alpha, \beta) : \underline{\text{Hom}}(a', b) \rightarrow \underline{\text{Hom}}(a, b')$.

1.6.3 InternalHomOnMorphismsWithGivenInternalHoms (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `InternalHomOnMorphismsWithGivenInternalHoms(s, alpha, beta, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = \underline{\text{Hom}}(a', b)$, two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$, and an object $r = \underline{\text{Hom}}(a, b')$. The output is the internal hom morphism $\underline{\text{Hom}}(\alpha, \beta) : \underline{\text{Hom}}(a', b) \rightarrow \underline{\text{Hom}}(a, b')$.

1.6.4 ClosedMonoidalRightEvaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ `ClosedMonoidalRightEvaluationMorphism(a, b)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes \underline{\text{Hom}}(a, b), b)$.

The arguments are two objects a, b . The output is the right evaluation morphism $\text{ev}_{a,b} : a \otimes \underline{\text{Hom}}(a, b) \rightarrow b$, i.e., the counit of the tensor hom adjunction.

1.6.5 ClosedMonoidalRightEvaluationMorphismWithGivenSource (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `ClosedMonoidalRightEvaluationMorphismWithGivenSource(a, b, s)` (operation)

Returns: a morphism in $\text{Hom}(s, b)$.

The arguments are two objects a, b and an object $s = a \otimes \underline{\text{Hom}}(a, b)$. The output is the right evaluation morphism $\text{ev}_{a,b} : a \otimes \underline{\text{Hom}}(a, b) \rightarrow b$, i.e., the counit of the tensor hom adjunction.

1.6.6 ClosedMonoidalRightCoevaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ ClosedMonoidalRightCoevaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(b, \underline{\text{Hom}}(a, a \otimes b))$.

The arguments are two objects a, b . The output is the right coevaluation morphism $\text{coev}_{a,b} : b \rightarrow \underline{\text{Hom}}(a, a \otimes b)$, i.e., the unit of the tensor hom adjunction.

1.6.7 ClosedMonoidalRightCoevaluationMorphismWithGivenRange (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ ClosedMonoidalRightCoevaluationMorphismWithGivenRange(a, b, r) (operation)

Returns: a morphism in $\text{Hom}(b, r)$.

The arguments are two objects a, b and an object $r = \underline{\text{Hom}}(a, a \otimes b)$. The output is the right coevaluation morphism $\text{coev}_{a,b} : b \rightarrow \underline{\text{Hom}}(a, a \otimes b)$, i.e., the unit of the tensor hom adjunction.

1.6.8 TensorProductToInternalHomRightAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ TensorProductToInternalHomRightAdjunctMorphism(a, b, f) (operation)

Returns: a morphism in $\text{Hom}(b, \underline{\text{Hom}}(a, c))$.

The arguments are two objects a, b and a morphism $f : a \otimes b \rightarrow c$. The output is a morphism $g : b \rightarrow \underline{\text{Hom}}(a, c)$ corresponding to f under the tensor hom adjunction.

1.6.9 TensorProductToInternalHomRightAdjunctMorphismWithGivenInternalHom (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ TensorProductToInternalHomRightAdjunctMorphismWithGivenInternalHom(a, b, f, i) (operation)

Returns: a morphism in $\text{Hom}(b, i)$.

The arguments are two objects a, b , a morphism $f : a \otimes b \rightarrow c$ and an object $i = \underline{\text{Hom}}(a, c)$. The output is a morphism $g : b \rightarrow i$ corresponding to f under the tensor hom adjunction.

1.6.10 TensorProductToInternalHomRightAdjunctionIsomorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ TensorProductToInternalHomRightAdjunctionIsomorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(H(a \otimes b, c), H(b, \underline{\text{Hom}}(a, c)))$.

The arguments are three objects a, b, c . The output is the tri-natural isomorphism $H(a \otimes b, c) \rightarrow H(b, \underline{\text{Hom}}(a, c))$ in the range category of the homomorphism structure H .

1.6.11 TensorProductToInternalHomRightAdjunctionIsomorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `TensorProductToInternalHomRightAdjunctionIsomorphismWithGivenObjects(s, a, b, c, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are five objects s, a, b, c, r where $s = H(a \otimes b, c)$ and $r = H(b, \underline{\text{Hom}}(a, c))$. The output is the tri-natural isomorphism $s \rightarrow r$ in the range category of the homomorphism structure H .

1.6.12 InternalHomToTensorProductRightAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `InternalHomToTensorProductRightAdjunctMorphism(a, c, g)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, c)$.

The arguments are two objects a, c and a morphism $g : b \rightarrow \underline{\text{Hom}}(a, c)$. The output is a morphism $f : a \otimes b \rightarrow c$ corresponding to g under the tensor hom adjunction.

1.6.13 InternalHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `InternalHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct(a, c, g, s)` (operation)

Returns: a morphism in $\text{Hom}(s, c)$.

The arguments are two objects a, c , a morphism $g : b \rightarrow \underline{\text{Hom}}(a, c)$ and an object $s = a \otimes b$. The output is a morphism $f : s \rightarrow c$ corresponding to g under the tensor hom adjunction.

1.6.14 InternalHomToTensorProductRightAdjunctionIsomorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `InternalHomToTensorProductRightAdjunctionIsomorphism(a, b, c)` (operation)

Returns: a morphism in $\text{Hom}(H(b, \underline{\text{Hom}}(a, c)), H(a \otimes b, c))$.

The arguments are three objects a, b, c . The output is the tri-natural isomorphism $H(b, \underline{\text{Hom}}(a, c)) \rightarrow H(a \otimes b, c)$ in the range category of the homomorphism structure H .

1.6.15 InternalHomToTensorProductRightAdjunctionIsomorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `InternalHomToTensorProductRightAdjunctionIsomorphismWithGivenObjects(s, a, b, c, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are five objects s, a, b, c, r where $s = H(b, \underline{\text{Hom}}(a, c))$ and $r = H(a \otimes b, c)$. The output is the tri-natural isomorphism $s \rightarrow r$ in the range category of the homomorphism structure H .

1.6.16 ClosedMonoidalLeftEvaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ ClosedMonoidalLeftEvaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a, b) \otimes a, b)$.

The arguments are two objects a, b . The output is the left evaluation morphism $\text{ev}_{a,b} : \underline{\text{Hom}}(a, b) \otimes a \rightarrow b$, i.e., the counit of the tensor hom adjunction.

1.6.17 ClosedMonoidalLeftEvaluationMorphismWithGivenSource (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ ClosedMonoidalLeftEvaluationMorphismWithGivenSource(a, b, s) (operation)

Returns: a morphism in $\text{Hom}(s, b)$.

The arguments are two objects a, b and an object $s = \underline{\text{Hom}}(a, b) \otimes a$. The output is the left evaluation morphism $\text{ev}_{a,b} : \underline{\text{Hom}}(a, b) \otimes a \rightarrow b$, i.e., the counit of the tensor hom adjunction.

1.6.18 ClosedMonoidalLeftCoevaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ ClosedMonoidalLeftCoevaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(b, \underline{\text{Hom}}(a, b \otimes a))$.

The arguments are two objects a, b . The output is the left coevaluation morphism $\text{coev}_{a,b} : b \rightarrow \underline{\text{Hom}}(a, b \otimes a)$, i.e., the unit of the tensor hom adjunction.

1.6.19 ClosedMonoidalLeftCoevaluationMorphismWithGivenRange (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ ClosedMonoidalLeftCoevaluationMorphismWithGivenRange(a, b, r) (operation)

Returns: a morphism in $\text{Hom}(b, r)$.

The arguments are two objects a, b and an object $r = \underline{\text{Hom}}(a, b \otimes a)$. The output is the left coevaluation morphism $\text{coev}_{a,b} : b \rightarrow \underline{\text{Hom}}(a, b \otimes a)$, i.e., the unit of the tensor hom adjunction.

1.6.20 TensorProductToInternalHomLeftAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ TensorProductToInternalHomLeftAdjunctMorphism(a, b, f) (operation)

Returns: a morphism in $\text{Hom}(a, \underline{\text{Hom}}(b, c))$.

The arguments are two objects a, b and a morphism $f : a \otimes b \rightarrow c$. The output is a morphism $g : a \rightarrow \underline{\text{Hom}}(b, c)$ corresponding to f under the tensor hom adjunction.

1.6.21 TensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ TensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom(a, b, f, i) (operation)

Returns: a morphism in $\text{Hom}(a, i)$.

The arguments are two objects a, b , a morphism $f : a \otimes b \rightarrow c$ and an object $i = \underline{\text{Hom}}(b, c)$. The output is a morphism $g : a \rightarrow i$ corresponding to f under the tensor hom adjunction.

1.6.22 TensorProductToInternalHomLeftAdjunctionIsomorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `TensorProductToInternalHomLeftAdjunctionIsomorphism(a, b, c)` (operation)

Returns: a morphism in $\text{Hom}(H(a \otimes b, c), H(a, \underline{\text{Hom}}(b, c)))$.

The arguments are three objects a, b, c . The output is the tri-natural isomorphism $H(a \otimes b, c) \rightarrow H(a, \underline{\text{Hom}}(b, c))$ in the range category of the homomorphism structure H .

1.6.23 TensorProductToInternalHomLeftAdjunctionIsomorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `TensorProductToInternalHomLeftAdjunctionIsomorphismWithGivenObjects(s, a, b, c, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are five objects s, a, b, c, r where $s = H(a \otimes b, c)$ and $r = H(a, \underline{\text{Hom}}(b, c))$. The output is the tri-natural isomorphism $s \rightarrow r$ in the range category of the homomorphism structure H .

1.6.24 InternalHomToTensorProductLeftAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `InternalHomToTensorProductLeftAdjunctMorphism(b, c, g)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes b, c)$.

The arguments are two objects b, c and a morphism $g : a \rightarrow \underline{\text{Hom}}(b, c)$. The output is a morphism $f : a \otimes b \rightarrow c$ corresponding to g under the tensor hom adjunction.

1.6.25 InternalHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `InternalHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct(b, c, g, s)` (operation)

Returns: a morphism in $\text{Hom}(s, c)$.

The arguments are two objects b, c , a morphism $g : a \rightarrow \underline{\text{Hom}}(b, c)$ and an object $s = a \otimes b$. The output is a morphism $f : s \rightarrow c$ corresponding to g under the tensor hom adjunction.

1.6.26 InternalHomToTensorProductLeftAdjunctionIsomorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `InternalHomToTensorProductLeftAdjunctionIsomorphism(a, b, c)` (operation)

Returns: a morphism in $\text{Hom}(H(a, \underline{\text{Hom}}(b, c)), H(a \otimes b, c))$.

The arguments are three objects a, b, c . The output is the tri-natural isomorphism $H(a, \underline{\text{Hom}}(b, c)) \rightarrow H(a \otimes b, c)$ in the range category of the homomorphism structure H .

1.6.27 InternalHomToTensorProductLeftAdjunctionIsomorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ InternalHomToTensorProductLeftAdjunctionIsomorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are five objects s, a, b, c, r where $s = H(a, \underline{\text{Hom}}(b, c))$ and $r = H(a \otimes b, c)$. The output is the tri-natural isomorphism $s \rightarrow r$ in the range category of the homomorphism structure H .

1.6.28 MonoidalPreComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MonoidalPreComposeMorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a, b) \otimes \underline{\text{Hom}}(b, c), \underline{\text{Hom}}(a, c))$.

The arguments are three objects a, b, c . The output is the precomposition morphism $\text{MonoidalPreComposeMorphism}_{a,b,c} : \underline{\text{Hom}}(a, b) \otimes \underline{\text{Hom}}(b, c) \rightarrow \underline{\text{Hom}}(a, c)$.

1.6.29 MonoidalPreComposeMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MonoidalPreComposeMorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{Hom}}(a, b) \otimes \underline{\text{Hom}}(b, c)$, three objects a, b, c , and an object $r = \underline{\text{Hom}}(a, c)$. The output is the precomposition morphism $\text{MonoidalPreComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{Hom}}(a, b) \otimes \underline{\text{Hom}}(b, c) \rightarrow \underline{\text{Hom}}(a, c)$.

1.6.30 MonoidalPostComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MonoidalPostComposeMorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(b, c) \otimes \underline{\text{Hom}}(a, b), \underline{\text{Hom}}(a, c))$.

The arguments are three objects a, b, c . The output is the postcomposition morphism $\text{MonoidalPostComposeMorphism}_{a,b,c} : \underline{\text{Hom}}(b, c) \otimes \underline{\text{Hom}}(a, b) \rightarrow \underline{\text{Hom}}(a, c)$.

1.6.31 MonoidalPostComposeMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MonoidalPostComposeMorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{Hom}}(b, c) \otimes \underline{\text{Hom}}(a, b)$, three objects a, b, c , and an object $r = \underline{\text{Hom}}(a, c)$. The output is the postcomposition morphism $\text{MonoidalPostComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{Hom}}(b, c) \otimes \underline{\text{Hom}}(a, b) \rightarrow \underline{\text{Hom}}(a, c)$.

1.6.32 DualOnObjects (for IsCapCategoryObject)

▷ `DualOnObjects(a)` (attribute)

Returns: an object

The argument is an object a . The output is its dual object a^\vee .

1.6.33 DualOnMorphisms (for IsCapCategoryMorphism)

▷ `DualOnMorphisms(alpha)` (attribute)

Returns: a morphism in $\text{Hom}(b^\vee, a^\vee)$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is its dual morphism $\alpha^\vee : b^\vee \rightarrow a^\vee$.

1.6.34 DualOnMorphismsWithGivenDuals (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `DualOnMorphismsWithGivenDuals(s, alpha, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The argument is an object $s = b^\vee$, a morphism $\alpha : a \rightarrow b$, and an object $r = a^\vee$. The output is the dual morphism $\alpha^\vee : b^\vee \rightarrow a^\vee$.

1.6.35 EvaluationForDual (for IsCapCategoryObject)

▷ `EvaluationForDual(a)` (attribute)

Returns: a morphism in $\text{Hom}(a^\vee \otimes a, 1)$.

The argument is an object a . The output is the evaluation morphism $\text{ev}_a : a^\vee \otimes a \rightarrow 1$.

1.6.36 EvaluationForDualWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `EvaluationForDualWithGivenTensorProduct(s, a, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = a^\vee \otimes a$, an object a , and an object $r = 1$. The output is the evaluation morphism $\text{ev}_a : a^\vee \otimes a \rightarrow 1$.

1.6.37 MorphismToBidual (for IsCapCategoryObject)

▷ `MorphismToBidual(a)` (attribute)

Returns: a morphism in $\text{Hom}(a, (a^\vee)^\vee)$.

The argument is an object a . The output is the morphism to the bidual $a \rightarrow (a^\vee)^\vee$.

1.6.38 MorphismToBidualWithGivenBidual (for IsCapCategoryObject, IsCapCategoryObject)

▷ `MorphismToBidualWithGivenBidual(a, r)` (operation)

Returns: a morphism in $\text{Hom}(a, r)$.

The arguments are an object a , and an object $r = (a^\vee)^\vee$. The output is the morphism to the bidual $a \rightarrow (a^\vee)^\vee$.

1.6.39 TensorProductInternalHomCompatibilityMorphism (for IsList)

▷ `TensorProductInternalHomCompatibilityMorphism(list)` (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b'), \underline{\text{Hom}}(a \otimes b, a' \otimes b'))$.

The argument is a list of four objects $[a, a', b, b']$. The output is the natural morphism $\text{TensorProductInternalHomCompatibilityMorphism}_{a, a', b, b'} : \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b') \rightarrow \underline{\text{Hom}}(a \otimes b, a' \otimes b')$.

1.6.40 TensorProductInternalHomCompatibilityMorphismWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ `TensorProductInternalHomCompatibilityMorphismWithGivenObjects(s, list, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are a list of four objects $[a, a', b, b']$, and two objects $s = \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b')$ and $r = \underline{\text{Hom}}(a \otimes b, a' \otimes b')$. The output is the natural morphism $\text{TensorProductInternalHomCompatibilityMorphismWithGivenObjects}_{a, a', b, b'} : \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b') \rightarrow \underline{\text{Hom}}(a \otimes b, a' \otimes b')$.

1.6.41 TensorProductDualityCompatibilityMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ `TensorProductDualityCompatibilityMorphism(a, b)` (operation)

Returns: a morphism in $\text{Hom}(a^\vee \otimes b^\vee, (a \otimes b)^\vee)$.

The arguments are two objects a, b . The output is the natural morphism $\text{TensorProductDualityCompatibilityMorphism} : a^\vee \otimes b^\vee \rightarrow (a \otimes b)^\vee$.

1.6.42 TensorProductDualityCompatibilityMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `TensorProductDualityCompatibilityMorphismWithGivenObjects(s, a, b, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = a^\vee \otimes b^\vee$, two objects a, b , and an object $r = (a \otimes b)^\vee$. The output is the natural morphism $\text{TensorProductDualityCompatibilityMorphismWithGivenObjects}_{a, b} : a^\vee \otimes b^\vee \rightarrow (a \otimes b)^\vee$.

1.6.43 MorphismFromTensorProductToInternalHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `MorphismFromTensorProductToInternalHom(a, b)` (operation)

Returns: a morphism in $\text{Hom}(a^\vee \otimes b, \underline{\text{Hom}}(a, b))$.

The arguments are two objects a, b . The output is the natural morphism $\text{MorphismFromTensorProductToInternalHom}_{a, b} : a^\vee \otimes b \rightarrow \underline{\text{Hom}}(a, b)$.

1.6.44 MorphismFromTensorProductToInternalHomWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromTensorProductToInternalHomWithGivenObjects(s, a, b, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = a^\vee \otimes b$, two objects a, b , and an object $r = \underline{\text{Hom}}(a, b)$. The output is the natural morphism $\text{MorphismFromTensorProductToInternalHomWithGivenObjects}_{a,b} : a^\vee \otimes b \rightarrow \underline{\text{Hom}}(a, b)$.

1.6.45 IsomorphismFromDualObjectToInternalHomIntoTensorUnit (for IsCapCategoryObject)

▷ IsomorphismFromDualObjectToInternalHomIntoTensorUnit(a) (attribute)

Returns: a morphism in $\text{Hom}(a^\vee, \underline{\text{Hom}}(a, 1))$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromDualObjectToInternalHomIntoTensorUnit}_a : a^\vee \rightarrow \underline{\text{Hom}}(a, 1)$.

1.6.46 IsomorphismFromInternalHomIntoTensorUnitToDualObject (for IsCapCategoryObject)

▷ IsomorphismFromInternalHomIntoTensorUnitToDualObject(a) (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a, 1), a^\vee)$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromInternalHomIntoTensorUnitToDualObject}_a : \underline{\text{Hom}}(a, 1) \rightarrow a^\vee$.

1.6.47 UniversalPropertyOfDual (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ UniversalPropertyOfDual(t, a, α) (operation)

Returns: a morphism in $\text{Hom}(t, a^\vee)$.

The arguments are two objects t, a , and a morphism $\alpha : t \otimes a \rightarrow 1$. The output is the morphism $t \rightarrow a^\vee$ given by the universal property of a^\vee .

1.6.48 LambdaIntroduction (for IsCapCategoryMorphism)

▷ LambdaIntroduction(α) (attribute)

Returns: a morphism in $\text{Hom}(1, \underline{\text{Hom}}(a, b))$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is the corresponding morphism $1 \rightarrow \underline{\text{Hom}}(a, b)$ under the tensor hom adjunction.

1.6.49 LambdaElimination (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ LambdaElimination(a, b, α) (operation)

Returns: a morphism in $\text{Hom}(a, b)$.

The arguments are two objects a, b , and a morphism $\alpha : 1 \rightarrow \underline{\text{Hom}}(a, b)$. The output is a morphism $a \rightarrow b$ corresponding to α under the tensor hom adjunction.

1.6.50 IsomorphismFromObjectToInternalHom (for IsCapCategoryObject)

▷ `IsomorphismFromObjectToInternalHom(a)` (attribute)

Returns: a morphism in $\text{Hom}(a, \underline{\text{Hom}}(1, a))$.

The argument is an object a . The output is the natural isomorphism $a \rightarrow \underline{\text{Hom}}(1, a)$.

1.6.51 IsomorphismFromObjectToInternalHomWithGivenInternalHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromObjectToInternalHomWithGivenInternalHom(a, r)` (operation)

Returns: a morphism in $\text{Hom}(a, r)$.

The argument is an object a , and an object $r = \underline{\text{Hom}}(1, a)$. The output is the natural isomorphism $a \rightarrow \underline{\text{Hom}}(1, a)$.

1.6.52 IsomorphismFromInternalHomToObject (for IsCapCategoryObject)

▷ `IsomorphismFromInternalHomToObject(a)` (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(1, a), a)$.

The argument is an object a . The output is the natural isomorphism $\underline{\text{Hom}}(1, a) \rightarrow a$.

1.6.53 IsomorphismFromInternalHomToObjectWithGivenInternalHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromInternalHomToObjectWithGivenInternalHom(a, s)` (operation)

Returns: a morphism in $\text{Hom}(s, a)$.

The argument is an object a , and an object $s = \underline{\text{Hom}}(1, a)$. The output is the natural isomorphism $\underline{\text{Hom}}(1, a) \rightarrow a$.

1.7 Left Coclosed Monoidal Categories

A monoidal category \mathbf{C} which has for each functor $- \otimes b : \mathbf{C} \rightarrow \mathbf{C}$ a left adjoint (denoted by $\underline{\text{coHom}}(-, b)$) is called a *left coclosed monoidal category*.

If no operations involving left coduals are installed manually, the left codual objects will be derived as $a_\vee := \underline{\text{coHom}}(1, a)$.

The corresponding GAP property is called `IsLeftCoclosedMonoidalCategory`.

1.7.1 LeftInternalCoHomOnObjects (for IsCapCategoryObject, IsCapCategoryObject)

▷ `LeftInternalCoHomOnObjects(a, b)` (operation)

Returns: an object

The arguments are two objects a, b . The output is the internal cohom object $\underline{\text{coHom}}_\ell(a, b)$.

1.7.2 LeftInternalCoHomOnMorphisms (for IsCapCategoryMorphism, IsCapCategoryMorphism)

▷ `LeftInternalCoHomOnMorphisms(alpha, beta)` (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}_\ell(a, b'), \underline{\text{coHom}}_\ell(a', b))$

The arguments are two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$. The output is the internal cohom morphism $\text{coHom}_\ell(\alpha, \beta) : \text{coHom}_\ell(a, b') \rightarrow \text{coHom}_\ell(a', b)$.

1.7.3 LeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHoms (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryMorphism, IsCapCategoryObject)

▷ LeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHoms(s, α, β, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = \text{coHom}_\ell(a, b')$, two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$, and an object $r = \text{coHom}_\ell(a', b)$. The output is the internal cohom morphism $\text{coHom}_\ell(\alpha, \beta) : \text{coHom}_\ell(a, b') \rightarrow \text{coHom}_\ell(a', b)$.

1.7.4 LeftCoclosedMonoidalEvaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalEvaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(b, \text{coHom}_\ell(b, a) \otimes a)$.

The arguments are two objects a, b . The output is the coclosed evaluation morphism $\text{coclev}_{a,b} : b \rightarrow \text{coHom}_\ell(b, a) \otimes a$, i.e., the unit of the cohom tensor adjunction.

1.7.5 LeftCoclosedMonoidalEvaluationMorphismWithGivenRange (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalEvaluationMorphismWithGivenRange(a, b, r) (operation)

Returns: a morphism in $\text{Hom}(b, r)$.

The arguments are two objects a, b and an object $r = \text{coHom}_\ell(b, a) \otimes a$. The output is the coclosed evaluation morphism $\text{coclev}_{a,b} : b \rightarrow \text{coHom}_\ell(b, a) \otimes a$, i.e., the unit of the cohom tensor adjunction.

1.7.6 LeftCoclosedMonoidalCoevaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalCoevaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(\text{coHom}_\ell(b \otimes a, a), b)$.

The arguments are two objects a, b . The output is the coclosed coevaluation morphism $\text{coclcoev}_{a,b} : \text{coHom}_\ell(b \otimes a, a) \rightarrow b$, i.e., the counit of the cohom tensor adjunction.

1.7.7 LeftCoclosedMonoidalCoevaluationMorphismWithGivenSource (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalCoevaluationMorphismWithGivenSource(a, b, s) (operation)

Returns: a morphism in $\text{Hom}(s, b)$.

The arguments are two objects a, b and an object $s = \text{coHom}_\ell(b \otimes a, a)$. The output is the coclosed coevaluation morphism $\text{coclcoev}_{a,b} : \text{coHom}_\ell(b \otimes a, a) \rightarrow b$, i.e., the unit of the cohom tensor adjunction.

1.7.8 TensorProductToLeftInternalCoHomAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `TensorProductToLeftInternalCoHomAdjunctMorphism(b, c, g)` (operation)

Returns: a morphism in $\text{Hom}(\text{coHom}_\ell(a, c), b)$.

The arguments are two objects b, c and a morphism $g : a \rightarrow b \otimes c$. The output is a morphism $f : \text{coHom}_\ell(a, c) \rightarrow b$ corresponding to g under the cohom tensor adjunction.

1.7.9 TensorProductToLeftInternalCoHomAdjunctMorphismWithGivenLeftInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `TensorProductToLeftInternalCoHomAdjunctMorphismWithGivenLeftInternalCoHom(b, c, g, i)` (operation)

Returns: a morphism in $\text{Hom}(i, b)$.

The arguments are two objects b, c , a morphism $g : a \rightarrow b \otimes c$ and an object $i = \text{coHom}_\ell(a, c)$. The output is a morphism $f : \text{coHom}_\ell(a, c) \rightarrow b$ corresponding to g under the cohom tensor adjunction.

1.7.10 LeftInternalCoHomToTensorProductAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `LeftInternalCoHomToTensorProductAdjunctMorphism(a, c, f)` (operation)

Returns: a morphism in $\text{Hom}(a, b \otimes c)$.

The arguments are two objects a, c and a morphism $f : \text{coHom}_\ell(a, c) \rightarrow b$. The output is a morphism $g : a \rightarrow b \otimes c$ corresponding to f under the cohom tensor adjunction.

1.7.11 LeftInternalCoHomToTensorProductAdjunctMorphismWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `LeftInternalCoHomToTensorProductAdjunctMorphismWithGivenTensorProduct(a, c, f, t)` (operation)

Returns: a morphism in $\text{Hom}(a, t)$.

The arguments are two objects a, c , a morphism $f : \text{coHom}_\ell(a, c) \rightarrow b$ and an object $t = b \otimes c$. The output is a morphism $g : a \rightarrow b \otimes c$ corresponding to f under the cohom tensor adjunction.

1.7.12 LeftCoclosedMonoidalPreCoComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `LeftCoclosedMonoidalPreCoComposeMorphism(a, b, c)` (operation)

Returns: a morphism in $\text{Hom}(\text{coHom}_\ell(a, c), \text{coHom}_\ell(b, c) \otimes \text{coHom}_\ell(a, b))$.

The arguments are three objects a, b, c . The output is the precocomposition morphism $\text{LeftCoclosedMonoidalPreCoComposeMorphism}_{a,b,c} : \text{coHom}_\ell(a, c) \rightarrow \text{coHom}_\ell(b, c) \otimes \text{coHom}_\ell(a, b)$.

1.7.13 LeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{coHom}}_\ell(a, c)$, three objects a, b, c , and an object $r = \underline{\text{coHom}}_\ell(a, b) \otimes \underline{\text{coHom}}_\ell(b, c)$. The output is the precocomposition morphism $\text{LeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{coHom}}_\ell(a, c) \rightarrow \underline{\text{coHom}}_\ell(b, c) \otimes \underline{\text{coHom}}_\ell(a, b)$.

1.7.14 LeftCoclosedMonoidalPostCoComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalPostCoComposeMorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}_\ell(a, c), \underline{\text{coHom}}_\ell(a, b) \otimes \underline{\text{coHom}}_\ell(b, c))$.

The arguments are three objects a, b, c . The output is the postcocomposition morphism $\text{LeftCoclosedMonoidalPostCoComposeMorphism}_{a,b,c} : \underline{\text{coHom}}_\ell(a, c) \rightarrow \underline{\text{coHom}}_\ell(a, b) \otimes \underline{\text{coHom}}_\ell(b, c)$.

1.7.15 LeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{coHom}}_\ell(a, c)$, three objects a, b, c , and an object $r = \underline{\text{coHom}}_\ell(b, c) \otimes \underline{\text{coHom}}_\ell(a, b)$. The output is the postcocomposition morphism $\text{LeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{coHom}}_\ell(a, c) \rightarrow \underline{\text{coHom}}_\ell(a, b) \otimes \underline{\text{coHom}}_\ell(b, c)$.

1.7.16 LeftCoDualOnObjects (for IsCapCategoryObject)

▷ LeftCoDualOnObjects(a) (attribute)

Returns: an object

The argument is an object a . The output is its codual object a_\vee .

1.7.17 LeftCoDualOnMorphisms (for IsCapCategoryMorphism)

▷ LeftCoDualOnMorphisms(α) (attribute)

Returns: a morphism in $\text{Hom}(b_\vee, a_\vee)$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is its codual morphism $\alpha_\vee : b_\vee \rightarrow a_\vee$.

1.7.18 LeftCoDualOnMorphismsWithGivenLeftCoDuals (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ LeftCoDualOnMorphismsWithGivenLeftCoDuals(s , α , r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The argument is an object $s = b_\vee$, a morphism $\alpha : a \rightarrow b$, and an object $r = a_\vee$. The output is the dual morphism $\alpha_\vee : b^\vee \rightarrow a^\vee$.

1.7.19 LeftCoclosedMonoidalEvaluationForLeftCoDual (for IsCapCategoryObject)

▷ LeftCoclosedMonoidalEvaluationForLeftCoDual(a) (attribute)

Returns: a morphism in $\text{Hom}(1, a_\vee \otimes a)$.

The argument is an object a . The output is the coclosed evaluation morphism $\text{coclev}_a : 1 \rightarrow a_\vee \otimes a$.

1.7.20 LeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct(s , a , r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = 1$, an object a , and an object $r = a_\vee \otimes a$. The output is the coclosed evaluation morphism $\text{coclev}_a : 1 \rightarrow a_\vee \otimes a$.

1.7.21 MorphismFromLeftCoBidual (for IsCapCategoryObject)

▷ MorphismFromLeftCoBidual(a) (attribute)

Returns: a morphism in $\text{Hom}((a_\vee)_\vee, a)$.

The argument is an object a . The output is the morphism from the cobidual $(a_\vee)_\vee \rightarrow a$.

1.7.22 MorphismFromLeftCoBidualWithGivenLeftCoBidual (for IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromLeftCoBidualWithGivenLeftCoBidual(a , s) (operation)

Returns: a morphism in $\text{Hom}(s, a)$.

The arguments are an object a , and an object $s = (a_\vee)_\vee$. The output is the morphism from the cobidual $(a_\vee)_\vee \rightarrow a$.

1.7.23 LeftInternalCoHomTensorProductCompatibilityMorphism (for IsList)

▷ LeftInternalCoHomTensorProductCompatibilityMorphism($list$) (operation)

Returns: a morphism in $\text{Hom}(\text{coHom}_\ell(a \otimes a', b \otimes b'), \text{coHom}_\ell(a, b) \otimes \text{coHom}_\ell(a', b'))$.

The argument is a list of four objects $[a, a', b, b']$. The output is the natural morphism $\text{LeftInternalCoHomTensorProductCompatibilityMorphism}_{a, a', b, b'} : \text{coHom}_\ell(a \otimes a', b \otimes b') \rightarrow \text{coHom}_\ell(a, b) \otimes \text{coHom}_\ell(a', b')$.

1.7.24 LeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ LeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects(s , $list$, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are a list of four objects $[a, a', b, b']$, and two objects $s = \text{coHom}_\ell(a \otimes a', b \otimes b')$ and $r = \text{coHom}_\ell(a, b) \otimes \text{coHom}_\ell(a', b')$. The output is the natural morphism $\text{LeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects}_{a, a', b, b'} : \text{coHom}_\ell(a \otimes a', b \otimes b') \rightarrow \text{coHom}_\ell(a, b) \otimes \text{coHom}_\ell(a', b')$.

1.7.25 LeftCoDualityTensorProductCompatibilityMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoDualityTensorProductCompatibilityMorphism(a , b) (operation)

Returns: a morphism in $\text{Hom}((a \otimes b)_\vee, a_\vee \otimes b_\vee)$.

The arguments are two objects a, b . The output is the natural morphism $\text{LeftCoDualityTensorProductCompatibilityMorphism} : (a \otimes b)_\vee \rightarrow a_\vee \otimes b_\vee$.

1.7.26 LeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ LeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects(s , a , b , r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = (a \otimes b)_\vee$, two objects a, b , and an object $r = a_\vee \otimes b_\vee$. The output is the natural morphism $\text{LeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects}_{a, b} : (a \otimes b)_\vee \rightarrow a_\vee \otimes b_\vee$.

1.7.27 MorphismFromLeftInternalCoHomToTensorProduct (for IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromLeftInternalCoHomToTensorProduct(a , b) (operation)

Returns: a morphism in $\text{Hom}(\text{coHom}_\ell(a, b), b_\vee \otimes a)$.

The arguments are two objects a, b . The output is the natural morphism $\text{MorphismFromLeftInternalCoHomToTensorProduct}_{a, b} : \text{coHom}_\ell(a, b) \rightarrow b_\vee \otimes a$.

1.7.28 MorphismFromLeftInternalCoHomToTensorProductWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromLeftInternalCoHomToTensorProductWithGivenObjects(s , a , b , r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \text{coHom}_\ell(a, b)$, two objects a, b , and an object $r = b_\vee \otimes a$. The output is the natural morphism $\text{MorphismFromLeftInternalCoHomToTensorProductWithGivenObjects}_{a, b} : \text{coHom}_\ell(a, b) \rightarrow a \otimes b_\vee$.

1.7.29 IsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit (for IsCapCategoryObject)

▷ `IsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit(a)` (attribute)

Returns: a morphism in $\text{Hom}(a_\vee, \underline{\text{coHom}}_\ell(1, a))$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit}_a : a_\vee \rightarrow \underline{\text{coHom}}_\ell(1, a)$.

1.7.30 IsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject (for IsCapCategoryObject)

▷ `IsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject(a)` (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}_\ell(1, a), a_\vee)$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject}_a : \underline{\text{coHom}}_\ell(1, a) \rightarrow a_\vee$.

1.7.31 UniversalPropertyOfLeftCoDual (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `UniversalPropertyOfLeftCoDual(t, a, alpha)` (operation)

Returns: a morphism in $\text{Hom}(a_\vee, t)$.

The arguments are two objects t, a , and a morphism $\alpha : 1 \rightarrow t \otimes a$. The output is the morphism $a_\vee \rightarrow t$ given by the universal property of a_\vee .

1.7.32 LeftCoclosedMonoidalLambdaIntroduction (for IsCapCategoryMorphism)

▷ `LeftCoclosedMonoidalLambdaIntroduction(alpha)` (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}_\ell(a, b), 1)$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is the corresponding morphism $\underline{\text{coHom}}_\ell(a, b) \rightarrow 1$ under the cohom tensor adjunction.

1.7.33 LeftCoclosedMonoidalLambdaElimination (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `LeftCoclosedMonoidalLambdaElimination(a, b, alpha)` (operation)

Returns: a morphism in $\text{Hom}(a, b)$.

The arguments are two objects a, b , and a morphism $\alpha : \underline{\text{coHom}}_\ell(a, b) \rightarrow 1$. The output is a morphism $a \rightarrow b$ corresponding to α under the cohom tensor adjunction.

1.7.34 IsomorphismFromObjectToLeftInternalCoHom (for IsCapCategoryObject)

▷ `IsomorphismFromObjectToLeftInternalCoHom(a)` (attribute)

Returns: a morphism in $\text{Hom}(a, \underline{\text{coHom}}_\ell(a, 1))$.

The argument is an object a . The output is the natural isomorphism $a \rightarrow \underline{\text{coHom}}_\ell(a, 1)$.

1.7.35 IsomorphismFromObjectToLeftInternalCoHomWithGivenLeftInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromObjectToLeftInternalCoHomWithGivenLeftInternalCoHom(a, r)` (operation)

Returns: a morphism in $\text{Hom}(a, r)$.

The argument is an object a , and an object $r = \underline{\text{coHom}}_\ell(a, 1)$. The output is the natural isomorphism $a \rightarrow \underline{\text{coHom}}_\ell(a, 1)$.

1.7.36 IsomorphismFromLeftInternalCoHomToObject (for IsCapCategoryObject)

▷ `IsomorphismFromLeftInternalCoHomToObject(a)` (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}_\ell(a, 1), a)$.

The argument is an object a . The output is the natural isomorphism $\underline{\text{coHom}}_\ell(a, 1) \rightarrow a$.

1.7.37 IsomorphismFromLeftInternalCoHomToObjectWithGivenLeftInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromLeftInternalCoHomToObjectWithGivenLeftInternalCoHom(a, s)` (operation)

Returns: a morphism in $\text{Hom}(s, a)$.

The argument is an object a , and an object $s = \underline{\text{coHom}}_\ell(a, 1)$. The output is the natural isomorphism $\underline{\text{coHom}}_\ell(a, 1) \rightarrow a$.

1.8 Coclosed Monoidal Categories

A monoidal category \mathbf{C} which has for each functor $- \otimes b : \mathbf{C} \rightarrow \mathbf{C}$ a left adjoint (denoted by $\underline{\text{coHom}}(-, b)$) is called a *coclosed monoidal category*.

If no operations involving coduals are installed manually, the codual objects will be derived as $a_\vee := \underline{\text{coHom}}(1, a)$.

The corresponding GAP property is called `IsCoclosedMonoidalCategory`.

1.8.1 InternalCoHomOnObjects (for IsCapCategoryObject, IsCapCategoryObject)

▷ `InternalCoHomOnObjects(a, b)` (operation)

Returns: an object

The arguments are two objects a, b . The output is the internal cohom object $\underline{\text{coHom}}(a, b)$.

1.8.2 InternalCoHomOnMorphisms (for IsCapCategoryMorphism, IsCapCategoryMorphism)

▷ `InternalCoHomOnMorphisms(alpha, beta)` (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, b'), \underline{\text{coHom}}(a', b))$

The arguments are two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$. The output is the internal cohom morphism $\underline{\text{coHom}}(\alpha, \beta) : \underline{\text{coHom}}(a, b') \rightarrow \underline{\text{coHom}}(a', b)$.

1.8.3 InternalCoHomOnMorphismsWithGivenInternalCoHoms (for IsCapCategory-Object, IsCapCategoryMorphism, IsCapCategoryMorphism, IsCapCategory-Object)

▷ InternalCoHomOnMorphismsWithGivenInternalCoHoms(s , α , β , r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$

The arguments are an object $s = \underline{\text{coHom}}(a, b')$, two morphisms $\alpha : a \rightarrow a', \beta : b \rightarrow b'$, and an object $r = \underline{\text{coHom}}(a', b)$. The output is the internal cohom morphism $\underline{\text{coHom}}(\alpha, \beta) : \underline{\text{coHom}}(a, b') \rightarrow \underline{\text{coHom}}(a', b)$.

1.8.4 CoclosedMonoidalRightEvaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ CoclosedMonoidalRightEvaluationMorphism(a , b) (operation)

Returns: a morphism in $\text{Hom}(b, a \otimes \underline{\text{coHom}}(b, a))$.

The arguments are two objects a, b . The output is the coclosed right evaluation morphism $\text{coclev}_{a,b} : b \rightarrow a \otimes \underline{\text{coHom}}(b, a)$, i.e., the unit of the cohom tensor adjunction.

1.8.5 CoclosedMonoidalRightEvaluationMorphismWithGivenRange (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ CoclosedMonoidalRightEvaluationMorphismWithGivenRange(a , b , r) (operation)

Returns: a morphism in $\text{Hom}(b, r)$.

The arguments are two objects a, b and an object $r = a \otimes \underline{\text{coHom}}(b, a)$. The output is the coclosed right evaluation morphism $\text{coclev}_{a,b} : b \rightarrow a \otimes \underline{\text{coHom}}(b, a)$, i.e., the unit of the cohom tensor adjunction.

1.8.6 CoclosedMonoidalRightCoevaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ CoclosedMonoidalRightCoevaluationMorphism(a , b) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a \otimes b, a), b)$.

The arguments are two objects a, b . The output is the coclosed right coevaluation morphism $\text{coclcov}_{a,b} : \underline{\text{coHom}}(a \otimes b, a) \rightarrow b$, i.e., the counit of the cohom tensor adjunction.

1.8.7 CoclosedMonoidalRightCoevaluationMorphismWithGivenSource (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ CoclosedMonoidalRightCoevaluationMorphismWithGivenSource(a , b , s) (operation)

Returns: a morphism in $\text{Hom}(s, b)$.

The arguments are two objects a, b and an object $s = \underline{\text{coHom}}(a \otimes b, a)$. The output is the coclosed right coevaluation morphism $\text{coclcov}_{a,b} : \underline{\text{coHom}}(a \otimes b, a) \rightarrow b$, i.e., the unit of the cohom tensor adjunction.

1.8.8 TensorProductToInternalCoHomRightAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `TensorProductToInternalCoHomRightAdjunctMorphism(b, c, g)` (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, b), c)$.

The arguments are two objects b, c and a morphism $g : a \rightarrow b \otimes c$. The output is a morphism $f : \underline{\text{coHom}}(a, b) \rightarrow c$ corresponding to g under the cohom tensor adjunction.

1.8.9 TensorProductToInternalCoHomRightAdjunctMorphismWithGivenInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `TensorProductToInternalCoHomRightAdjunctMorphismWithGivenInternalCoHom(b, c, g, i)` (operation)

Returns: a morphism in $\text{Hom}(i, c)$.

The arguments are two objects b, c , a morphism $g : a \rightarrow b \otimes c$ and an object $i = \underline{\text{coHom}}(a, b)$. The output is a morphism $f : \underline{\text{coHom}}(a, b) \rightarrow c$ corresponding to g under the cohom tensor adjunction.

1.8.10 InternalCoHomToTensorProductRightAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `InternalCoHomToTensorProductRightAdjunctMorphism(a, b, f)` (operation)

Returns: a morphism in $\text{Hom}(a, b \otimes c)$.

The arguments are two objects a, b and a morphism $f : \underline{\text{coHom}}(a, b) \rightarrow c$. The output is a morphism $g : a \rightarrow b \otimes c$ corresponding to f under the cohom tensor adjunction.

1.8.11 InternalCoHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `InternalCoHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct(a, b, f, t)` (operation)

Returns: a morphism in $\text{Hom}(a, t)$.

The arguments are two objects a, b , a morphism $f : \underline{\text{coHom}}(a, b) \rightarrow c$ and an object $t = b \otimes c$. The output is a morphism $g : a \rightarrow t$ corresponding to f under the cohom tensor adjunction.

1.8.12 CoclosedMonoidalLeftEvaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ `CoclosedMonoidalLeftEvaluationMorphism(a, b)` (operation)

Returns: a morphism in $\text{Hom}(b, \underline{\text{coHom}}(b, a) \otimes a)$.

The arguments are two objects a, b . The output is the coclosed left evaluation morphism $\text{coclev}_{a,b} : b \rightarrow \underline{\text{coHom}}(b, a) \otimes a$, i.e., the unit of the cohom tensor adjunction.

1.8.13 CoclosedMonoidalLeftEvaluationMorphismWithGivenRange (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ CoclosedMonoidalLeftEvaluationMorphismWithGivenRange(a, b, r) (operation)

Returns: a morphism in $\text{Hom}(b, r)$.

The arguments are two objects a, b and an object $r = \underline{\text{coHom}}(b, a) \otimes a$. The output is the coclosed left evaluation morphism $\text{coclev}_{a,b} : b \rightarrow \underline{\text{coHom}}(b, a) \otimes a$, i.e., the unit of the cohom tensor adjunction.

1.8.14 CoclosedMonoidalLeftCoevaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ CoclosedMonoidalLeftCoevaluationMorphism(a, b) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(b \otimes a, a), b)$.

The arguments are two objects a, b . The output is the coclosed left coevaluation morphism $\text{coclcov}_{a,b} : \underline{\text{coHom}}(b \otimes a, a) \rightarrow b$, i.e., the counit of the cohom tensor adjunction.

1.8.15 CoclosedMonoidalLeftCoevaluationMorphismWithGivenSource (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ CoclosedMonoidalLeftCoevaluationMorphismWithGivenSource(a, b, s) (operation)

Returns: a morphism in $\text{Hom}(s, b)$.

The arguments are two objects a, b and an object $s = \underline{\text{coHom}}(b \otimes a, a)$. The output is the coclosed left coevaluation morphism $\text{coclcov}_{a,b} : \underline{\text{coHom}}(b \otimes a, a) \rightarrow b$, i.e., the unit of the cohom tensor adjunction.

1.8.16 TensorProductToInternalCoHomLeftAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ TensorProductToInternalCoHomLeftAdjunctMorphism(b, c, g) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, c), b)$.

The arguments are two objects b, c and a morphism $g : a \rightarrow b \otimes c$. The output is a morphism $f : \underline{\text{coHom}}(a, c) \rightarrow b$ corresponding to g under the cohom tensor adjunction.

1.8.17 TensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ TensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom(b, c, g, i) (operation)

Returns: a morphism in $\text{Hom}(i, b)$.

The arguments are two objects b, c , a morphism $g : a \rightarrow b \otimes c$ and an object $i = \underline{\text{coHom}}(a, c)$. The output is a morphism $f : \underline{\text{coHom}}(a, c) \rightarrow b$ corresponding to g under the cohom tensor adjunction.

1.8.18 InternalCoHomToTensorProductLeftAdjunctMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ InternalCoHomToTensorProductLeftAdjunctMorphism(a, c, f) (operation)

Returns: a morphism in $\text{Hom}(a, b \otimes c)$.

The arguments are two objects a, c and a morphism $f : \underline{\text{coHom}}(a, c) \rightarrow b$. The output is a morphism $g : a \rightarrow b \otimes c$ corresponding to f under the cohom tensor adjunction.

1.8.19 InternalCoHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ InternalCoHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct(a, c, f, t) (operation)

Returns: a morphism in $\text{Hom}(a, t)$.

The arguments are two objects a, c , a morphism $f : \underline{\text{coHom}}(a, c) \rightarrow b$ and an object $t = b \otimes c$. The output is a morphism $g : a \rightarrow t$ corresponding to f under the cohom tensor adjunction.

1.8.20 MonoidalPreCoComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MonoidalPreCoComposeMorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, c), \underline{\text{coHom}}(b, c) \otimes \underline{\text{coHom}}(a, b))$.

The arguments are three objects a, b, c . The output is the precocomposition morphism $\text{MonoidalPreCoComposeMorphism}_{a,b,c} : \underline{\text{coHom}}(a, c) \rightarrow \underline{\text{coHom}}(b, c) \otimes \underline{\text{coHom}}(a, b)$.

1.8.21 MonoidalPreCoComposeMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MonoidalPreCoComposeMorphismWithGivenObjects(s, a, b, c, r) (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{coHom}}(a, c)$, three objects a, b, c , and an object $r = \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(b, c)$. The output is the precocomposition morphism $\text{MonoidalPreCoComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{coHom}}(a, c) \rightarrow \underline{\text{coHom}}(b, c) \otimes \underline{\text{coHom}}(a, b)$.

1.8.22 MonoidalPostCoComposeMorphism (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MonoidalPostCoComposeMorphism(a, b, c) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, c), \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(b, c))$.

The arguments are three objects a, b, c . The output is the postcocomposition morphism $\text{MonoidalPostCoComposeMorphism}_{a,b,c} : \underline{\text{coHom}}(a, c) \rightarrow \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(b, c)$.

1.8.23 MonoidalPostCoComposeMorphismWithGivenObjects (for IsCapCategory-Object, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `MonoidalPostCoComposeMorphismWithGivenObjects(s, a, b, c, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \underline{\text{coHom}}(a, c)$, three objects a, b, c , and an object $r = \underline{\text{coHom}}(b, c) \otimes \underline{\text{coHom}}(a, b)$. The output is the postcomposition morphism $\text{MonoidalPostCoComposeMorphismWithGivenObjects}_{a,b,c} : \underline{\text{coHom}}(a, c) \rightarrow \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(b, c)$.

1.8.24 CoDualOnObjects (for IsCapCategoryObject)

▷ `CoDualOnObjects(a)` (attribute)

Returns: an object

The argument is an object a . The output is its codual object a_\vee .

1.8.25 CoDualOnMorphisms (for IsCapCategoryMorphism)

▷ `CoDualOnMorphisms(alpha)` (attribute)

Returns: a morphism in $\text{Hom}(b_\vee, a_\vee)$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is its codual morphism $\alpha_\vee : b_\vee \rightarrow a_\vee$.

1.8.26 CoDualOnMorphismsWithGivenCoDuals (for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject)

▷ `CoDualOnMorphismsWithGivenCoDuals(s, alpha, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The argument is an object $s = b_\vee$, a morphism $\alpha : a \rightarrow b$, and an object $r = a_\vee$. The output is the dual morphism $\alpha_\vee : b^\vee \rightarrow a^\vee$.

1.8.27 CoclosedEvaluationForCoDual (for IsCapCategoryObject)

▷ `CoclosedEvaluationForCoDual(a)` (attribute)

Returns: a morphism in $\text{Hom}(1, a_\vee \otimes a)$.

The argument is an object a . The output is the coclosed evaluation morphism $\text{coclev}_a : 1 \rightarrow a_\vee \otimes a$.

1.8.28 CoclosedEvaluationForCoDualWithGivenTensorProduct (for IsCapCategory-Object, IsCapCategoryObject, IsCapCategoryObject)

▷ `CoclosedEvaluationForCoDualWithGivenTensorProduct(s, a, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = 1$, an object a , and an object $r = a_\vee \otimes a$. The output is the coclosed evaluation morphism $\text{coclev}_a : 1 \rightarrow a_\vee \otimes a$.

1.8.29 MorphismFromCoBidual (for IsCapCategoryObject)

▷ `MorphismFromCoBidual(a)` (attribute)

Returns: a morphism in $\text{Hom}((a_V)_V, a)$.

The argument is an object a . The output is the morphism from the cobidual $(a_V)_V \rightarrow a$.

1.8.30 MorphismFromCoBidualWithGivenCoBidual (for IsCapCategoryObject, IsCapCategoryObject)

▷ `MorphismFromCoBidualWithGivenCoBidual(a, s)` (operation)

Returns: a morphism in $\text{Hom}(s, a)$.

The arguments are an object a , and an object $s = (a_V)_V$. The output is the morphism from the cobidual $(a_V)_V \rightarrow a$.

1.8.31 InternalCoHomTensorProductCompatibilityMorphism (for IsList)

▷ `InternalCoHomTensorProductCompatibilityMorphism(list)` (operation)

Returns: a morphism in $\text{Hom}(\text{coHom}(a \otimes a', b \otimes b'), \text{coHom}(a, b) \otimes \text{coHom}(a', b'))$.

The argument is a list of four objects $[a, a', b, b']$. The output is the natural morphism $\text{InternalCoHomTensorProductCompatibilityMorphism}_{a, a', b, b'} : \text{coHom}(a \otimes a', b \otimes b') \rightarrow \text{coHom}(a, b) \otimes \text{coHom}(a', b')$.

1.8.32 InternalCoHomTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ `InternalCoHomTensorProductCompatibilityMorphismWithGivenObjects(s, list, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are a list of four objects $[a, a', b, b']$, and two objects $s = \text{coHom}(a \otimes a', b \otimes b')$ and $r = \text{coHom}(a, b) \otimes \text{coHom}(a', b')$. The output is the natural morphism $\text{InternalCoHomTensorProductCompatibilityMorphismWithGivenObjects}_{a, a', b, b'} : \text{coHom}(a \otimes a', b \otimes b') \rightarrow \text{coHom}(a, b) \otimes \text{coHom}(a', b')$.

1.8.33 CoDualityTensorProductCompatibilityMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ `CoDualityTensorProductCompatibilityMorphism(a, b)` (operation)

Returns: a morphism in $\text{Hom}((a \otimes b)_V, a_V \otimes b_V)$.

The arguments are two objects a, b . The output is the natural morphism $\text{CoDualityTensorProductCompatibilityMorphism} : (a \otimes b)_V \rightarrow a_V \otimes b_V$.

1.8.34 CoDualityTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `CoDualityTensorProductCompatibilityMorphismWithGivenObjects(s, a, b, r)` (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = (a \otimes b)_V$, two objects a, b , and an object $r = a_V \otimes b_V$. The output is the natural morphism $\text{CoDualityTensorProductCompatibilityMorphismWithGivenObjects}_{a,b} : (a \otimes b)_V \rightarrow a_V \otimes b_V$.

1.8.35 MorphismFromInternalCoHomToTensorProduct (for IsCapCategoryObject, IsCapCategoryObject)

▷ $\text{MorphismFromInternalCoHomToTensorProduct}(a, b)$ (operation)

Returns: a morphism in $\text{Hom}(\text{coHom}(a, b), b_V \otimes a)$.

The arguments are two objects a, b . The output is the natural morphism $\text{MorphismFromInternalCoHomToTensorProduct}_{a,b} : \text{coHom}(a, b) \rightarrow b_V \otimes a$.

1.8.36 MorphismFromInternalCoHomToTensorProductWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ $\text{MorphismFromInternalCoHomToTensorProductWithGivenObjects}(s, a, b, r)$ (operation)

Returns: a morphism in $\text{Hom}(s, r)$.

The arguments are an object $s = \text{coHom}(a, b)$, two objects a, b , and an object $r = b_V \otimes a$. The output is the natural morphism $\text{MorphismFromInternalCoHomToTensorProductWithGivenObjects}_{a,b} : \text{coHom}(a, b) \rightarrow a \otimes b_V$.

1.8.37 IsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit (for IsCapCategoryObject)

▷ $\text{IsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit}(a)$ (attribute)

Returns: a morphism in $\text{Hom}(a_V, \text{coHom}(1, a))$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit}_a : a_V \rightarrow \text{coHom}(1, a)$.

1.8.38 IsomorphismFromInternalCoHomFromTensorUnitToCoDualObject (for IsCapCategoryObject)

▷ $\text{IsomorphismFromInternalCoHomFromTensorUnitToCoDualObject}(a)$ (attribute)

Returns: a morphism in $\text{Hom}(\text{coHom}(1, a), a_V)$.

The argument is an object a . The output is the isomorphism $\text{IsomorphismFromInternalCoHomFromTensorUnitToCoDualObject}_a : \text{coHom}(1, a) \rightarrow a_V$.

1.8.39 UniversalPropertyOfCoDual (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ $\text{UniversalPropertyOfCoDual}(t, a, \alpha)$ (operation)

Returns: a morphism in $\text{Hom}(a_V, t)$.

The arguments are two objects t, a , and a morphism $\alpha : 1 \rightarrow t \otimes a$. The output is the morphism $a_V \rightarrow t$ given by the universal property of a_V .

1.8.40 CoLambdaIntroduction (for IsCapCategoryMorphism)

▷ `CoLambdaIntroduction(alpha)` (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, b), 1)$.

The argument is a morphism $\alpha : a \rightarrow b$. The output is the corresponding morphism $\underline{\text{coHom}}(a, b) \rightarrow 1$ under the cohom tensor adjunction.

1.8.41 CoLambdaElimination (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism)

▷ `CoLambdaElimination(a, b, alpha)` (operation)

Returns: a morphism in $\text{Hom}(a, b)$.

The arguments are two objects a, b , and a morphism $\alpha : \underline{\text{coHom}}(a, b) \rightarrow 1$. The output is a morphism $a \rightarrow b$ corresponding to α under the cohom tensor adjunction.

1.8.42 IsomorphismFromObjectToInternalCoHom (for IsCapCategoryObject)

▷ `IsomorphismFromObjectToInternalCoHom(a)` (attribute)

Returns: a morphism in $\text{Hom}(a, \underline{\text{coHom}}(a, 1))$.

The argument is an object a . The output is the natural isomorphism $a \rightarrow \underline{\text{coHom}}(a, 1)$.

1.8.43 IsomorphismFromObjectToInternalCoHomWithGivenInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromObjectToInternalCoHomWithGivenInternalCoHom(a, r)` (operation)

Returns: a morphism in $\text{Hom}(a, r)$.

The argument is an object a , and an object $r = \underline{\text{coHom}}(a, 1)$. The output is the natural isomorphism $a \rightarrow \underline{\text{coHom}}(a, 1)$.

1.8.44 IsomorphismFromInternalCoHomToObject (for IsCapCategoryObject)

▷ `IsomorphismFromInternalCoHomToObject(a)` (attribute)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, 1), a)$.

The argument is an object a . The output is the natural isomorphism $\underline{\text{coHom}}(a, 1) \rightarrow a$.

1.8.45 IsomorphismFromInternalCoHomToObjectWithGivenInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromInternalCoHomToObjectWithGivenInternalCoHom(a, s)` (operation)

Returns: a morphism in $\text{Hom}(s, a)$.

The argument is an object a , and an object $s = \underline{\text{coHom}}(a, 1)$. The output is the natural isomorphism $\underline{\text{coHom}}(a, 1) \rightarrow a$.

1.9 Symmetric Closed Monoidal Categories

A monoidal category \mathbf{C} which is symmetric and closed is called a *symmetric closed monoidal category*.

The corresponding GAP property is given by `IsSymmetricClosedMonoidalCategory`.

1.10 Symmetric Coclosed Monoidal Categories

A monoidal category \mathbf{C} which is symmetric and coclosed is called a *symmetric coclosed monoidal category*.

The corresponding GAP property is given by `IsSymmetricCoclosedMonoidalCategory`.

1.11 Rigid Symmetric Closed Monoidal Categories

A symmetric closed monoidal category \mathbf{C} satisfying

- the natural morphism

$\underline{\text{Hom}}_\ell(a, a') \otimes \underline{\text{Hom}}_\ell(b, b') \rightarrow \underline{\text{Hom}}_\ell(a \otimes b, a' \otimes b')$ is an isomorphism,

- the natural morphism

$a \rightarrow \underline{\text{Hom}}_\ell(\underline{\text{Hom}}_\ell(a, 1), 1)$ is an isomorphism is called a *rigid symmetric closed monoidal category*.

If no operations involving the closed structure are installed manually, the internal hom objects will be derived as $\underline{\text{Hom}}_\ell(a, b) := a^\vee \otimes b$ and, in particular, $\underline{\text{Hom}}_\ell(a, 1) := a^\vee \otimes 1$.

The corresponding GAP property is given by `IsRigidSymmetricClosedMonoidalCategory`.

1.11.1 IsomorphismFromTensorProductWithDualObjectToInternalHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromTensorProductWithDualObjectToInternalHom(a, b)` (operation)

Returns: a morphism in $\text{Hom}(a^\vee \otimes b, \underline{\text{Hom}}(a, b))$.

The arguments are two objects a, b . The output is the natural morphism $\text{IsomorphismFromTensorProductWithDualObjectToInternalHom}_{a,b} : a^\vee \otimes b \rightarrow \underline{\text{Hom}}(a, b)$.

1.11.2 IsomorphismFromInternalHomToTensorProductWithDualObject (for IsCapCategoryObject, IsCapCategoryObject)

▷ `IsomorphismFromInternalHomToTensorProductWithDualObject(a, b)` (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a, b), a^\vee \otimes b)$.

The arguments are two objects a, b . The output is the inverse of `IsomorphismFromTensorProductWithDualObjectToInternalHom`, namely $\text{IsomorphismFromInternalHomToTensorProductWithDualObject}_{a,b} : \underline{\text{Hom}}(a, b) \rightarrow a^\vee \otimes b$.

1.11.3 MorphismFromInternalHomToTensorProduct (for IsCapCategoryObject, IsCapCategoryObject)

▷ `MorphismFromInternalHomToTensorProduct(a, b)` (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a, b), a^\vee \otimes b)$.

The arguments are two objects a, b . The output is the inverse of `MorphismFromTensorProductToInternalHomWithGivenObjects`, namely $\text{MorphismFromInternalHomToTensorProductWithGivenObjects}_{a,b} : \underline{\text{Hom}}(a, b) \rightarrow a^\vee \otimes b$.

1.11.4 MorphismFromInternalHomToTensorProductWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromInternalHomToTensorProductWithGivenObjects(s, a, b, r) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a, b), a^\vee \otimes b)$.

The arguments are an object $s = \underline{\text{Hom}}(a, b)$, two objects a, b , and an object $r = a^\vee \otimes b$. The output is the inverse of MorphismFromTensorProductToInternalHomWithGivenObjects, namely MorphismFromInternalHomToTensorProductWithGivenObjects $_{a,b} : \underline{\text{Hom}}(a, b) \rightarrow a^\vee \otimes b$.

1.11.5 TensorProductInternalHomCompatibilityMorphismInverse (for IsList)

▷ TensorProductInternalHomCompatibilityMorphismInverse($list$) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a \otimes b, a' \otimes b'), \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b'))$.

The argument is a list of four objects $[a, a', b, b']$. The output is the natural morphism TensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects $_{a,a',b,b'} : \underline{\text{Hom}}(a \otimes b, a' \otimes b') \rightarrow \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b')$.

1.11.6 TensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ TensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects($s, list, r$) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{Hom}}(a \otimes b, a' \otimes b'), \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b'))$.

The arguments are a list of four objects $[a, a', b, b']$, and two objects $s = \underline{\text{Hom}}(a \otimes b, a' \otimes b')$ and $r = \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b')$. The output is the natural morphism TensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects $_{a,a',b,b'} : \underline{\text{Hom}}(a \otimes b, a' \otimes b') \rightarrow \underline{\text{Hom}}(a, a') \otimes \underline{\text{Hom}}(b, b')$.

1.11.7 CoevaluationForDual (for IsCapCategoryObject)

▷ CoevaluationForDual(a) (attribute)

Returns: a morphism in $\text{Hom}(1, a \otimes a^\vee)$.

The argument is an object a . The output is the coevaluation morphism $\text{coev}_a : 1 \rightarrow a \otimes a^\vee$.

1.11.8 CoevaluationForDualWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ CoevaluationForDualWithGivenTensorProduct(s, a, r) (operation)

Returns: a morphism in $\text{Hom}(1, a \otimes a^\vee)$.

The arguments are an object $s = 1$, an object a , and an object $r = a \otimes a^\vee$. The output is the coevaluation morphism $\text{coev}_a : 1 \rightarrow a \otimes a^\vee$.

1.11.9 TraceMap (for IsCapCategoryMorphism)

▷ TraceMap($alpha$) (attribute)

Returns: a morphism in $\text{Hom}(1, 1)$.

The argument is an endomorphism $\alpha : a \rightarrow a$. The output is the trace morphism $\text{trace}_\alpha : 1 \rightarrow 1$.

1.11.10 RankMorphism (for IsCapCategoryObject)

▷ RankMorphism(a) (attribute)

Returns: a morphism in $\text{Hom}(1, 1)$.

The argument is an object a . The output is the rank morphism $\text{rank}_a : 1 \rightarrow 1$.

1.11.11 MorphismFromBidual (for IsCapCategoryObject)

▷ MorphismFromBidual(a) (attribute)

Returns: a morphism in $\text{Hom}((a^\vee)^\vee, a)$.

The argument is an object a . The output is the inverse of the morphism to the bidual $(a^\vee)^\vee \rightarrow a$.

1.11.12 MorphismFromBidualWithGivenBidual (for IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromBidualWithGivenBidual(a, s) (operation)

Returns: a morphism in $\text{Hom}((a^\vee)^\vee, a)$.

The argument is an object a , and an object $s = (a^\vee)^\vee$. The output is the inverse of the morphism to the bidual $(a^\vee)^\vee \rightarrow a$.

1.12 Rigid Symmetric Coclosed Monoidal Categories

A symmetric coclosed monoidal category \mathbf{C} satisfying

- the natural morphism

$\underline{\text{coHom}}(a \otimes a', b \otimes b') \rightarrow \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(a', b')$ is an isomorphism,

- the natural morphism

$\underline{\text{coHom}}(1, \underline{\text{coHom}}(1, a)) \rightarrow a$ is an isomorphism is called a *rigid symmetric coclosed monoidal category*.

If no operations involving the coclosed structure are installed manually, the internal cohom objects will be derived as $\underline{\text{coHom}}(a, b) := a \otimes b_\vee$ and, in particular, $\underline{\text{coHom}}(1, a) := 1 \otimes a_\vee$.

The corresponding GAP property is given by IsRigidSymmetricCoclosedMonoidalCategory.

1.12.1 IsomorphismFromInternalCoHomToTensorProductWithCoDualObject (for IsCapCategoryObject, IsCapCategoryObject)

▷ IsomorphismFromInternalCoHomToTensorProductWithCoDualObject(a, b) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, b), b_\vee \otimes a)$.

The arguments are two objects a, b . The output is the natural morphism $\text{IsomorphismFromInternalCoHomToTensorProductWithCoDualObjectWithGivenObjects}_{a,b} : \underline{\text{coHom}}(a, b) \rightarrow b_\vee \otimes a$.

1.12.2 IsomorphismFromTensorProductWithCoDualObjectToInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ IsomorphismFromTensorProductWithCoDualObjectToInternalCoHom(a, b) (operation)

Returns: a morphism in $\text{Hom}(a_{\vee} \otimes b, \underline{\text{coHom}}(b, a))$.

The arguments are two objects a, b . The output is the inverse of IsomorphismFromInternalCoHomToTensorProductWithCoDualObject, namely IsomorphismFromTensorProductWithCoDualObjectToInternalCoHom $_{a,b} : a_{\vee} \otimes b \rightarrow \underline{\text{coHom}}(b, a)$.

1.12.3 MorphismFromTensorProductToInternalCoHom (for IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromTensorProductToInternalCoHom(a, b) (operation)

Returns: a morphism in $\text{Hom}(a_{\vee} \otimes b, \underline{\text{coHom}}(b, a))$.

The arguments are two objects a, b . The output is the inverse of MorphismFromInternalCoHomToTensorProductWithGivenObjects, namely MorphismFromTensorProductToInternalCoHomWithGivenObjects $_{a,b} : a_{\vee} \otimes b \rightarrow \underline{\text{coHom}}(b, a)$.

1.12.4 MorphismFromTensorProductToInternalCoHomWithGivenObjects (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ MorphismFromTensorProductToInternalCoHomWithGivenObjects(s, a, b, r) (operation)

Returns: a morphism in $\text{Hom}(a_{\vee} \otimes b, \underline{\text{coHom}}(b, a))$.

The arguments are an object $s_{\vee} = a \otimes b$, two objects a, b , and an object $r = \underline{\text{coHom}}(b, a)$. The output is the inverse of MorphismFromInternalCoHomToTensorProductWithGivenObjects, namely MorphismFromTensorProductToInternalCoHomWithGivenObjects $_{a,b} : a_{\vee} \otimes b \rightarrow \underline{\text{coHom}}(b, a)$.

1.12.5 InternalCoHomTensorProductCompatibilityMorphismInverse (for IsList)

▷ InternalCoHomTensorProductCompatibilityMorphismInverse($list$) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(a', b'), \underline{\text{coHom}}(a \otimes a', b \otimes b'))$.

The argument is a list of four objects $[a, a', b, b']$. The output is the natural morphism InternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects $_{a,a',b,b'} : \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(a', b') \rightarrow \underline{\text{coHom}}(a \otimes a', b \otimes b')$.

1.12.6 InternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects (for IsCapCategoryObject, IsList, IsCapCategoryObject)

▷ InternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects($s, list, r$) (operation)

Returns: a morphism in $\text{Hom}(\underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(a', b'), \underline{\text{coHom}}(a \otimes a', b \otimes b'))$.

The arguments are a list of four objects $[a, a', b, b']$, and two objects $s = \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(a', b')$ and $r = \underline{\text{coHom}}(a \otimes a', b \otimes b')$. The output is the natural morphism InternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects $_{a,a',b,b'} : \underline{\text{coHom}}(a, b) \otimes \underline{\text{coHom}}(a', b') \rightarrow \underline{\text{coHom}}(a \otimes a', b \otimes b')$.

1.12.7 CoclosedCoevaluationForCoDual (for IsCapCategoryObject)

▷ `CoclosedCoevaluationForCoDual(a)` (attribute)

Returns: a morphism in $\text{Hom}(a \otimes a_V, 1)$.

The argument is an object a . The output is the coclosed coevaluation morphism $\text{coclcov}_a : a \otimes a_V \rightarrow 1$.

1.12.8 CoclosedCoevaluationForCoDualWithGivenTensorProduct (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `CoclosedCoevaluationForCoDualWithGivenTensorProduct(s, a, r)` (operation)

Returns: a morphism in $\text{Hom}(a \otimes a_V, 1)$.

The arguments are an object $s = a \otimes a_V$, an object a , and an object $r = 1$. The output is the coclosed coevaluation morphism $\text{coclcov}_a : a \otimes a_V \rightarrow 1$.

1.12.9 CoTraceMap (for IsCapCategoryMorphism)

▷ `CoTraceMap(alpha)` (attribute)

Returns: a morphism in $\text{Hom}(1, 1)$.

The argument is an endomorphism $\alpha : a \rightarrow a$. The output is the cotrace morphism $\text{cotrace}_\alpha : 1 \rightarrow 1$.

1.12.10 CoRankMorphism (for IsCapCategoryObject)

▷ `CoRankMorphism(a)` (attribute)

Returns: a morphism in $\text{Hom}(1, 1)$.

The argument is an object a . The output is the corank morphism $\text{corank}_a : 1 \rightarrow 1$.

1.12.11 MorphismToCoBidual (for IsCapCategoryObject)

▷ `MorphismToCoBidual(a)` (attribute)

Returns: a morphism in $\text{Hom}(a, (a_V)_V)$.

The argument is an object a . The output is the inverse of the morphism from the cobidual $a \rightarrow (a_V)_V$.

1.12.12 MorphismToCoBidualWithGivenCoBidual (for IsCapCategoryObject, IsCapCategoryObject)

▷ `MorphismToCoBidualWithGivenCoBidual(a, r)` (operation)

Returns: a morphism in $\text{Hom}(a, (a_V)_V)$.

The argument is an object a , and an object $r = (a_V)_V$. The output is the inverse of the morphism from the cobidual $a \rightarrow (a_V)_V$.

1.13 Convenience Methods

1.13.1 InternalHom (for IsCapCategoryCell, IsCapCategoryCell)

▷ InternalHom(a, b) (operation)
Returns: a cell

This is a convenience method. The arguments are two cells a, b . The output is the internal hom cell. If a, b are two CAP objects the output is the internal Hom object $\underline{\text{Hom}}(a, b)$. If at least one of the arguments is a CAP morphism the output is a CAP morphism, namely the internal hom on morphisms, where any object is replaced by its identity morphism.

1.13.2 InternalCoHom (for IsCapCategoryCell, IsCapCategoryCell)

▷ InternalCoHom(a, b) (operation)
Returns: a cell

This is a convenience method. The arguments are two cells a, b . The output is the internal cohom cell. If a, b are two CAP objects the output is the internal cohom object $\underline{\text{coHom}}(a, b)$. If at least one of the arguments is a CAP morphism the output is a CAP morphism, namely the internal cohom on morphisms, where any object is replaced by its identity morphism.

1.13.3 LeftInternalHom (for IsCapCategoryCell, IsCapCategoryCell)

▷ LeftInternalHom(a, b) (operation)
Returns: a cell

This is a convenience method. The arguments are two cells a, b . The output is the internal hom cell. If a, b are two CAP objects the output is the internal Hom object $\underline{\text{Hom}}_\ell(a, b)$. If at least one of the arguments is a CAP morphism the output is a CAP morphism, namely the internal hom on morphisms, where any object is replaced by its identity morphism.

1.13.4 LeftInternalCoHom (for IsCapCategoryCell, IsCapCategoryCell)

▷ LeftInternalCoHom(a, b) (operation)
Returns: a cell

This is a convenience method. The arguments are two cells a, b . The output is the internal cohom cell. If a, b are two CAP objects the output is the internal cohom object $\underline{\text{coHom}}_\ell(a, b)$. If at least one of the arguments is a CAP morphism the output is a CAP morphism, namely the internal cohom on morphisms, where any object is replaced by its identity morphism.

1.14 Add-methods

1.14.1 AddLeftDistributivityExpanding (for IsCapCategory, IsFunction)

▷ AddLeftDistributivityExpanding(C, F) (operation)
 ▷ AddLeftDistributivityExpanding(C, F, weight) (operation)
Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftDistributivityExpanding. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational

complexity of the function (lower weight = less complex = faster execution). $F : (a, L) \mapsto \text{LeftDistributivityExpanding}(a, L)$.

1.14.2 AddLeftDistributivityExpandingWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ `AddLeftDistributivityExpandingWithGivenObjects(C, F)` (operation)
- ▷ `AddLeftDistributivityExpandingWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftDistributivityExpandingWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, L, r) \mapsto \text{LeftDistributivityExpandingWithGivenObjects}(s, a, L, r)$.

1.14.3 AddLeftDistributivityFactoring (for IsCapCategory, IsFunction)

- ▷ `AddLeftDistributivityFactoring(C, F)` (operation)
- ▷ `AddLeftDistributivityFactoring(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftDistributivityFactoring`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, L) \mapsto \text{LeftDistributivityFactoring}(a, L)$.

1.14.4 AddLeftDistributivityFactoringWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ `AddLeftDistributivityFactoringWithGivenObjects(C, F)` (operation)
- ▷ `AddLeftDistributivityFactoringWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftDistributivityFactoringWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, L, r) \mapsto \text{LeftDistributivityFactoringWithGivenObjects}(s, a, L, r)$.

1.14.5 AddRightDistributivityExpanding (for IsCapCategory, IsFunction)

- ▷ `AddRightDistributivityExpanding(C, F)` (operation)
- ▷ `AddRightDistributivityExpanding(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `RightDistributivityExpanding`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (L, a) \mapsto \text{RightDistributivityExpanding}(L, a)$.

1.14.6 AddRightDistributivityExpandingWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddRightDistributivityExpandingWithGivenObjects(C, F) (operation)
- ▷ AddRightDistributivityExpandingWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation RightDistributivityExpandingWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, L, a, r) \mapsto \text{RightDistributivityExpandingWithGivenObjects}(s, L, a, r)$.

1.14.7 AddRightDistributivityFactoring (for IsCapCategory, IsFunction)

- ▷ AddRightDistributivityFactoring(C, F) (operation)
- ▷ AddRightDistributivityFactoring($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation RightDistributivityFactoring. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (L, a) \mapsto \text{RightDistributivityFactoring}(L, a)$.

1.14.8 AddRightDistributivityFactoringWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddRightDistributivityFactoringWithGivenObjects(C, F) (operation)
- ▷ AddRightDistributivityFactoringWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation RightDistributivityFactoringWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, L, a, r) \mapsto \text{RightDistributivityFactoringWithGivenObjects}(s, L, a, r)$.

1.14.9 AddBraiding (for IsCapCategory, IsFunction)

- ▷ AddBraiding(C, F) (operation)
- ▷ AddBraiding($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation Braiding. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{Braiding}(a, b)$.

1.14.10 AddBraidingInverse (for IsCapCategory, IsFunction)

- ▷ AddBraidingInverse(C, F) (operation)
- ▷ AddBraidingInverse($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation BraidingInverse. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{BraidingInverse}(a, b)$.

1.14.11 AddBraidingInverseWithGivenTensorProducts (for IsCapCategory, IsFunction)

- ▷ AddBraidingInverseWithGivenTensorProducts(C, F) (operation)
- ▷ AddBraidingInverseWithGivenTensorProducts($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation BraidingInverseWithGivenTensorProducts. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{BraidingInverseWithGivenTensorProducts}(s, a, b, r)$.

1.14.12 AddBraidingWithGivenTensorProducts (for IsCapCategory, IsFunction)

- ▷ AddBraidingWithGivenTensorProducts(C, F) (operation)
- ▷ AddBraidingWithGivenTensorProducts($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation BraidingWithGivenTensorProducts. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{BraidingWithGivenTensorProducts}(s, a, b, r)$.

1.14.13 AddClosedMonoidalLeftCoevaluationMorphism (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalLeftCoevaluationMorphism(C, F) (operation)
- ▷ AddClosedMonoidalLeftCoevaluationMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalLeftCoevaluationMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{ClosedMonoidalLeftCoevaluationMorphism}(a, b)$.

1.14.14 AddClosedMonoidalLeftCoevaluationMorphismWithGivenRange (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalLeftCoevaluationMorphismWithGivenRange(C , F) (operation)
- ▷ AddClosedMonoidalLeftCoevaluationMorphismWithGivenRange(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalLeftCoevaluationMorphismWithGivenRange. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, r) \mapsto \text{ClosedMonoidalLeftCoevaluationMorphismWithGivenRange}(a, b, r)$.

1.14.15 AddClosedMonoidalLeftEvaluationMorphism (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalLeftEvaluationMorphism(C , F) (operation)
- ▷ AddClosedMonoidalLeftEvaluationMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalLeftEvaluationMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{ClosedMonoidalLeftEvaluationMorphism}(a, b)$.

1.14.16 AddClosedMonoidalLeftEvaluationMorphismWithGivenSource (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalLeftEvaluationMorphismWithGivenSource(C , F) (operation)
- ▷ AddClosedMonoidalLeftEvaluationMorphismWithGivenSource(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalLeftEvaluationMorphismWithGivenSource. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, s) \mapsto \text{ClosedMonoidalLeftEvaluationMorphismWithGivenSource}(a, b, s)$.

1.14.17 AddClosedMonoidalRightCoevaluationMorphism (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalRightCoevaluationMorphism(C , F) (operation)
- ▷ AddClosedMonoidalRightCoevaluationMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalRightCoevaluationMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{ClosedMonoidalRightCoevaluationMorphism}(a, b)$.

1.14.18 AddClosedMonoidalRightCoevaluationMorphismWithGivenRange (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalRightCoevaluationMorphismWithGivenRange(C , F) (operation)
- ▷ AddClosedMonoidalRightCoevaluationMorphismWithGivenRange(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalRightCoevaluationMorphismWithGivenRange. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, r) \mapsto \text{ClosedMonoidalRightCoevaluationMorphismWithGivenRange}(a, b, r)$.

1.14.19 AddClosedMonoidalRightEvaluationMorphism (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalRightEvaluationMorphism(C , F) (operation)
- ▷ AddClosedMonoidalRightEvaluationMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalRightEvaluationMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{ClosedMonoidalRightEvaluationMorphism}(a, b)$.

1.14.20 AddClosedMonoidalRightEvaluationMorphismWithGivenSource (for IsCapCategory, IsFunction)

- ▷ AddClosedMonoidalRightEvaluationMorphismWithGivenSource(C , F) (operation)
- ▷ AddClosedMonoidalRightEvaluationMorphismWithGivenSource(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation ClosedMonoidalRightEvaluationMorphismWithGivenSource. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, s) \mapsto \text{ClosedMonoidalRightEvaluationMorphismWithGivenSource}(a, b, s)$.

1.14.21 AddDualOnMorphisms (for IsCapCategory, IsFunction)

- ▷ AddDualOnMorphisms(C , F) (operation)
- ▷ AddDualOnMorphisms(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation DualOnMorphisms. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha) \mapsto \text{DualOnMorphisms}(alpha)$.

1.14.22 AddDualOnMorphismsWithGivenDuals (for IsCapCategory, IsFunction)

- ▷ AddDualOnMorphismsWithGivenDuals(C, F) (operation)
- ▷ AddDualOnMorphismsWithGivenDuals($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation DualOnMorphismsWithGivenDuals. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, alpha, r) \mapsto \text{DualOnMorphismsWithGivenDuals}(s, alpha, r)$.

1.14.23 AddDualOnObjects (for IsCapCategory, IsFunction)

- ▷ AddDualOnObjects(C, F) (operation)
- ▷ AddDualOnObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation DualOnObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{DualOnObjects}(a)$.

1.14.24 AddEvaluationForDual (for IsCapCategory, IsFunction)

- ▷ AddEvaluationForDual(C, F) (operation)
- ▷ AddEvaluationForDual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation EvaluationForDual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{EvaluationForDual}(a)$.

1.14.25 AddEvaluationForDualWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ AddEvaluationForDualWithGivenTensorProduct(C, F) (operation)
- ▷ AddEvaluationForDualWithGivenTensorProduct($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation EvaluationForDualWithGivenTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, r) \mapsto \text{EvaluationForDualWithGivenTensorProduct}(s, a, r)$.

1.14.26 AddInternalHomOnMorphisms (for IsCapCategory, IsFunction)

- ▷ AddInternalHomOnMorphisms(C, F) (operation)
- ▷ AddInternalHomOnMorphisms($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomOnMorphisms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (\alpha, \beta) \mapsto \text{InternalHomOnMorphisms}(\alpha, \beta)$.

1.14.27 AddInternalHomOnMorphismsWithGivenInternalHoms (for IsCapCategory, IsFunction)

▷ `AddInternalHomOnMorphismsWithGivenInternalHoms(C, F)` (operation)

▷ `AddInternalHomOnMorphismsWithGivenInternalHoms(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomOnMorphismsWithGivenInternalHoms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, \alpha, \beta, r) \mapsto \text{InternalHomOnMorphismsWithGivenInternalHoms}(s, \alpha, \beta, r)$.

1.14.28 AddInternalHomOnObjects (for IsCapCategory, IsFunction)

▷ `AddInternalHomOnObjects(C, F)` (operation)

▷ `AddInternalHomOnObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomOnObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{InternalHomOnObjects}(a, b)$.

1.14.29 AddInternalHomToTensorProductLeftAdjunctMorphism (for IsCapCategory, IsFunction)

▷ `AddInternalHomToTensorProductLeftAdjunctMorphism(C, F)` (operation)

▷ `AddInternalHomToTensorProductLeftAdjunctMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomToTensorProductLeftAdjunctMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g) \mapsto \text{InternalHomToTensorProductLeftAdjunctMorphism}(b, c, g)$.

1.14.30 AddInternalHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct (for IsCapCategory, IsFunction)

▷ `AddInternalHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct(C, F)` (operation)

▷ `AddInternalHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct(C, F,`

`weight)`

`(operation)`

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g, s) \mapsto \text{InternalHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct}(b, c, g, s)$.

1.14.31 AddInternalHomToTensorProductLeftAdjunctionIsomorphism (for IsCapCategory, IsFunction)

▷ `AddInternalHomToTensorProductLeftAdjunctionIsomorphism(C, F)` `(operation)`

▷ `AddInternalHomToTensorProductLeftAdjunctionIsomorphism(C, F, weight)` `(operation)`

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomToTensorProductLeftAdjunctionIsomorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{InternalHomToTensorProductLeftAdjunctionIsomorphism}(a, b, c)$.

1.14.32 AddInternalHomToTensorProductLeftAdjunctionIsomorphismWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddInternalHomToTensorProductLeftAdjunctionIsomorphismWithGivenObjects(C, F)` `(operation)`

▷ `AddInternalHomToTensorProductLeftAdjunctionIsomorphismWithGivenObjects(C, F, weight)` `(operation)`

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomToTensorProductLeftAdjunctionIsomorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{InternalHomToTensorProductLeftAdjunctionIsomorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.33 AddInternalHomToTensorProductRightAdjunctMorphism (for IsCapCategory, IsFunction)

▷ `AddInternalHomToTensorProductRightAdjunctMorphism(C, F)` `(operation)`

▷ `AddInternalHomToTensorProductRightAdjunctMorphism(C, F, weight)` `(operation)`

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalHomToTensorProductRightAdjunctMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, c, g) \mapsto \text{InternalHomToTensorProductRightAdjunctMorphism}(a, c, g)$.

1.14.34 AddInternalHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ AddInternalHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct(C , F) (operation)
- ▷ AddInternalHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, c, g, s) \mapsto \text{InternalHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct}(a, c, g, s)$.

1.14.35 AddInternalHomToTensorProductRightAdjunctionIsomorphism (for IsCapCategory, IsFunction)

- ▷ AddInternalHomToTensorProductRightAdjunctionIsomorphism(C , F) (operation)
- ▷ AddInternalHomToTensorProductRightAdjunctionIsomorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalHomToTensorProductRightAdjunctionIsomorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{InternalHomToTensorProductRightAdjunctionIsomorphism}(a, b, c)$.

1.14.36 AddInternalHomToTensorProductRightAdjunctionIsomorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddInternalHomToTensorProductRightAdjunctionIsomorphismWithGivenObjects(C , F) (operation)
- ▷ AddInternalHomToTensorProductRightAdjunctionIsomorphismWithGivenObjects(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalHomToTensorProductRightAdjunctionIsomorphismWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{InternalHomToTensorProductRightAdjunctionIsomorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.37 AddIsomorphismFromDualObjectToInternalHomIntoTensorUnit (for IsCapCategory, IsFunction)

- ▷ AddIsomorphismFromDualObjectToInternalHomIntoTensorUnit(C , F) (operation)
- ▷ AddIsomorphismFromDualObjectToInternalHomIntoTensorUnit(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromDualObjectToInternalHomIntoTensorUnit`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromDualObjectToInternalHomIntoTensorUnit}(a)$.

1.14.38 AddIsomorphismFromInternalHomIntoTensorUnitToDualObject (for IsCapCategory, IsFunction)

▷ `AddIsomorphismFromInternalHomIntoTensorUnitToDualObject(C, F)` (operation)

▷ `AddIsomorphismFromInternalHomIntoTensorUnitToDualObject(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromInternalHomIntoTensorUnitToDualObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromInternalHomIntoTensorUnitToDualObject}(a)$.

1.14.39 AddIsomorphismFromInternalHomToObject (for IsCapCategory, IsFunction)

▷ `AddIsomorphismFromInternalHomToObject(C, F)` (operation)

▷ `AddIsomorphismFromInternalHomToObject(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromInternalHomToObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromInternalHomToObject}(a)$.

1.14.40 AddIsomorphismFromInternalHomToObjectWithGivenInternalHom (for IsCapCategory, IsFunction)

▷ `AddIsomorphismFromInternalHomToObjectWithGivenInternalHom(C, F)` (operation)

▷ `AddIsomorphismFromInternalHomToObjectWithGivenInternalHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromInternalHomToObjectWithGivenInternalHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,s) \mapsto \text{IsomorphismFromInternalHomToObjectWithGivenInternalHom}(a,s)$.

1.14.41 AddIsomorphismFromObjectToInternalHom (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromObjectToInternalHom(C , F) (operation)

▷ AddIsomorphismFromObjectToInternalHom(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation IsomorphismFromObjectToInternalHom. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromObjectToInternalHom}(a)$.

1.14.42 AddIsomorphismFromObjectToInternalHomWithGivenInternalHom (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromObjectToInternalHomWithGivenInternalHom(C , F) (operation)

▷ AddIsomorphismFromObjectToInternalHomWithGivenInternalHom(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation IsomorphismFromObjectToInternalHomWithGivenInternalHom. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{IsomorphismFromObjectToInternalHomWithGivenInternalHom}(a, r)$.

1.14.43 AddLambdaElimination (for IsCapCategory, IsFunction)

▷ AddLambdaElimination(C , F) (operation)

▷ AddLambdaElimination(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LambdaElimination. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, alpha) \mapsto \text{LambdaElimination}(a, b, alpha)$.

1.14.44 AddLambdaIntroduction (for IsCapCategory, IsFunction)

▷ AddLambdaIntroduction(C , F) (operation)

▷ AddLambdaIntroduction(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LambdaIntroduction. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha) \mapsto \text{LambdaIntroduction}(alpha)$.

1.14.45 AddMonoidalPostComposeMorphism (for IsCapCategory, IsFunction)

- ▷ AddMonoidalPostComposeMorphism(C, F) (operation)
- ▷ AddMonoidalPostComposeMorphism(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPostComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{MonoidalPostComposeMorphism}(a, b, c)$.

1.14.46 AddMonoidalPostComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddMonoidalPostComposeMorphismWithGivenObjects(C, F) (operation)
- ▷ AddMonoidalPostComposeMorphismWithGivenObjects(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPostComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{MonoidalPostComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.47 AddMonoidalPreComposeMorphism (for IsCapCategory, IsFunction)

- ▷ AddMonoidalPreComposeMorphism(C, F) (operation)
- ▷ AddMonoidalPreComposeMorphism(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPreComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{MonoidalPreComposeMorphism}(a, b, c)$.

1.14.48 AddMonoidalPreComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddMonoidalPreComposeMorphismWithGivenObjects(C, F) (operation)
- ▷ AddMonoidalPreComposeMorphismWithGivenObjects(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPreComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{MonoidalPreComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.49 AddMorphismFromTensorProductToInternalHom (for IsCapCategory, IsFunction)

▷ AddMorphismFromTensorProductToInternalHom(C, F) (operation)

▷ AddMorphismFromTensorProductToInternalHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismFromTensorProductToInternalHom. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{MorphismFromTensorProductToInternalHom}(a, b)$.

1.14.50 AddMorphismFromTensorProductToInternalHomWithGivenObjects (for IsCapCategory, IsFunction)

▷ AddMorphismFromTensorProductToInternalHomWithGivenObjects(C, F) (operation)

▷ AddMorphismFromTensorProductToInternalHomWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismFromTensorProductToInternalHomWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{MorphismFromTensorProductToInternalHomWithGivenObjects}(s, a, b, r)$.

1.14.51 AddMorphismToBidual (for IsCapCategory, IsFunction)

▷ AddMorphismToBidual(C, F) (operation)

▷ AddMorphismToBidual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismToBidual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{MorphismToBidual}(a)$.

1.14.52 AddMorphismToBidualWithGivenBidual (for IsCapCategory, IsFunction)

▷ AddMorphismToBidualWithGivenBidual(C, F) (operation)

▷ AddMorphismToBidualWithGivenBidual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismToBidualWithGivenBidual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{MorphismToBidualWithGivenBidual}(a, r)$.

1.14.53 AddTensorProductDualityCompatibilityMorphism (for IsCapCategory, IsFunction)

- ▷ AddTensorProductDualityCompatibilityMorphism(C, F) (operation)
- ▷ AddTensorProductDualityCompatibilityMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductDualityCompatibilityMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{TensorProductDualityCompatibilityMorphism}(a, b)$.

1.14.54 AddTensorProductDualityCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddTensorProductDualityCompatibilityMorphismWithGivenObjects(C, F) (operation)
- ▷ AddTensorProductDualityCompatibilityMorphismWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductDualityCompatibilityMorphismWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{TensorProductDualityCompatibilityMorphismWithGivenObjects}(s, a, b, r)$.

1.14.55 AddTensorProductInternalHomCompatibilityMorphism (for IsCapCategory, IsFunction)

- ▷ AddTensorProductInternalHomCompatibilityMorphism(C, F) (operation)
- ▷ AddTensorProductInternalHomCompatibilityMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductInternalHomCompatibilityMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (list) \mapsto \text{TensorProductInternalHomCompatibilityMorphism}(list)$.

1.14.56 AddTensorProductInternalHomCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddTensorProductInternalHomCompatibilityMorphismWithGivenObjects(C, F) (operation)
- ▷ AddTensorProductInternalHomCompatibilityMorphismWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation

TensorProductInternalHomCompatibilityMorphismWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (source, list, range) \mapsto \text{TensorProductInternalHomCompatibilityMorphismWithGivenObjects}(source, list, range)$.

1.14.57 AddTensorProductToInternalHomLeftAdjunctMorphism (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalHomLeftAdjunctMorphism(C, F) (operation)
- ▷ AddTensorProductToInternalHomLeftAdjunctMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToInternalHomLeftAdjunctMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f) \mapsto \text{TensorProductToInternalHomLeftAdjunctMorphism}(a, b, f)$.

1.14.58 AddTensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom(C, F) (operation)
- ▷ AddTensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f, i) \mapsto \text{TensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom}(a, b, f, i)$.

1.14.59 AddTensorProductToInternalHomLeftAdjunctionIsomorphism (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalHomLeftAdjunctionIsomorphism(C, F) (operation)
- ▷ AddTensorProductToInternalHomLeftAdjunctionIsomorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToInternalHomLeftAdjunctionIsomorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{TensorProductToInternalHomLeftAdjunctionIsomorphism}(a, b, c)$.

1.14.60 AddTensorProductToInternalHomLeftAdjunctionIsomorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalHomLeftAdjunctionIsomorphismWithGivenObjects(C, F) (operation)
- ▷ AddTensorProductToInternalHomLeftAdjunctionIsomorphismWithGivenObjects(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToInternalHomLeftAdjunctionIsomorphismWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{TensorProductToInternalHomLeftAdjunctionIsomorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.61 AddTensorProductToInternalHomRightAdjunctMorphism (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalHomRightAdjunctMorphism(C, F) (operation)
- ▷ AddTensorProductToInternalHomRightAdjunctMorphism(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToInternalHomRightAdjunctMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f) \mapsto \text{TensorProductToInternalHomRightAdjunctMorphism}(a, b, f)$.

1.14.62 AddTensorProductToInternalHomRightAdjunctMorphismWithGivenInternalHom (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalHomRightAdjunctMorphismWithGivenInternalHom(C, F) (operation)
- ▷ AddTensorProductToInternalHomRightAdjunctMorphismWithGivenInternalHom(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToInternalHomRightAdjunctMorphismWithGivenInternalHom. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f, i) \mapsto \text{TensorProductToInternalHomRightAdjunctMorphismWithGivenInternalHom}(a, b, f, i)$.

1.14.63 AddTensorProductToInternalHomRightAdjunctionIsomorphism (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalHomRightAdjunctionIsomorphism(C, F) (operation)
- ▷ AddTensorProductToInternalHomRightAdjunctionIsomorphism(C, F, weight) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductToInternalHomRightAdjunctionIsomorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{TensorProductToInternalHomRightAdjunctionIsomorphism}(a, b, c)$.

1.14.64 AddTensorProductToInternalHomRightAdjunctionIsomorphismWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddTensorProductToInternalHomRightAdjunctionIsomorphismWithGivenObjects(C, F)` (operation)
 ▷ `AddTensorProductToInternalHomRightAdjunctionIsomorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductToInternalHomRightAdjunctionIsomorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{TensorProductToInternalHomRightAdjunctionIsomorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.65 AddUniversalPropertyOfDual (for IsCapCategory, IsFunction)

▷ `AddUniversalPropertyOfDual(C, F)` (operation)
 ▷ `AddUniversalPropertyOfDual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `UniversalPropertyOfDual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (t, a, \alpha) \mapsto \text{UniversalPropertyOfDual}(t, a, \alpha)$.

1.14.66 AddCoDualOnMorphisms (for IsCapCategory, IsFunction)

▷ `AddCoDualOnMorphisms(C, F)` (operation)
 ▷ `AddCoDualOnMorphisms(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoDualOnMorphisms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (\alpha) \mapsto \text{CoDualOnMorphisms}(\alpha)$.

1.14.67 AddCoDualOnMorphismsWithGivenCoDuals (for IsCapCategory, IsFunction)

▷ `AddCoDualOnMorphismsWithGivenCoDuals(C, F)` (operation)
 ▷ `AddCoDualOnMorphismsWithGivenCoDuals(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoDualOnMorphismsWithGivenCoDuals`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, \alpha, r) \mapsto \text{CoDualOnMorphismsWithGivenCoDuals}(s, \alpha, r)$.

1.14.68 AddCoDualOnObjects (for IsCapCategory, IsFunction)

- ▷ `AddCoDualOnObjects(C, F)` (operation)
- ▷ `AddCoDualOnObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoDualOnObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{CoDualOnObjects}(a)$.

1.14.69 AddCoDualityTensorProductCompatibilityMorphism (for IsCapCategory, IsFunction)

- ▷ `AddCoDualityTensorProductCompatibilityMorphism(C, F)` (operation)
- ▷ `AddCoDualityTensorProductCompatibilityMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoDualityTensorProductCompatibilityMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{CoDualityTensorProductCompatibilityMorphism}(a, b)$.

1.14.70 AddCoDualityTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ `AddCoDualityTensorProductCompatibilityMorphismWithGivenObjects(C, F)` (operation)
- ▷ `AddCoDualityTensorProductCompatibilityMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoDualityTensorProductCompatibilityMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{CoDualityTensorProductCompatibilityMorphismWithGivenObjects}(s, a, b, r)$.

1.14.71 AddCoLambdaElimination (for IsCapCategory, IsFunction)

- ▷ `AddCoLambdaElimination(C, F)` (operation)
- ▷ `AddCoLambdaElimination(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoLambdaElimination`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, \alpha) \mapsto \text{CoLambdaElimination}(a, b, \alpha)$.

1.14.72 AddCoLambdaIntroduction (for IsCapCategory, IsFunction)

- ▷ `AddCoLambdaIntroduction(C, F)` (operation)
- ▷ `AddCoLambdaIntroduction(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoLambdaIntroduction`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (\alpha) \mapsto \text{CoLambdaIntroduction}(\alpha)$.

1.14.73 AddCoclosedEvaluationForCoDual (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedEvaluationForCoDual(C, F)` (operation)
- ▷ `AddCoclosedEvaluationForCoDual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedEvaluationForCoDual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{CoclosedEvaluationForCoDual}(a)$.

1.14.74 AddCoclosedEvaluationForCoDualWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedEvaluationForCoDualWithGivenTensorProduct(C, F)` (operation)
- ▷ `AddCoclosedEvaluationForCoDualWithGivenTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedEvaluationForCoDualWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, r) \mapsto \text{CoclosedEvaluationForCoDualWithGivenTensorProduct}(s, a, r)$.

1.14.75 AddCoclosedMonoidalLeftCoevaluationMorphism (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalLeftCoevaluationMorphism(C, F)` (operation)
- ▷ `AddCoclosedMonoidalLeftCoevaluationMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalLeftCoevaluationMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computa-

tional complexity of the function (lower weight = less complex = faster execution). $F : (a,b) \mapsto \text{CoclosedMonoidalLeftCoevaluationMorphism}(a,b)$.

1.14.76 AddCoclosedMonoidalLeftCoevaluationMorphismWithGivenSource (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalLeftCoevaluationMorphismWithGivenSource(C, F)` (operation)
- ▷ `AddCoclosedMonoidalLeftCoevaluationMorphismWithGivenSource(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalLeftCoevaluationMorphismWithGivenSource`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b,s) \mapsto \text{CoclosedMonoidalLeftCoevaluationMorphismWithGivenSource}(a,b,s)$.

1.14.77 AddCoclosedMonoidalLeftEvaluationMorphism (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalLeftEvaluationMorphism(C, F)` (operation)
- ▷ `AddCoclosedMonoidalLeftEvaluationMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalLeftEvaluationMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b) \mapsto \text{CoclosedMonoidalLeftEvaluationMorphism}(a,b)$.

1.14.78 AddCoclosedMonoidalLeftEvaluationMorphismWithGivenRange (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalLeftEvaluationMorphismWithGivenRange(C, F)` (operation)
- ▷ `AddCoclosedMonoidalLeftEvaluationMorphismWithGivenRange(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalLeftEvaluationMorphismWithGivenRange`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b,r) \mapsto \text{CoclosedMonoidalLeftEvaluationMorphismWithGivenRange}(a,b,r)$.

1.14.79 AddCoclosedMonoidalRightCoevaluationMorphism (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalRightCoevaluationMorphism(C, F)` (operation)
- ▷ `AddCoclosedMonoidalRightCoevaluationMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalRightCoevaluationMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{CoclosedMonoidalRightCoevaluationMorphism}(a, b)$.

1.14.80 AddCoclosedMonoidalRightCoevaluationMorphismWithGivenSource (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalRightCoevaluationMorphismWithGivenSource(C, F)` (operation)
- ▷ `AddCoclosedMonoidalRightCoevaluationMorphismWithGivenSource(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalRightCoevaluationMorphismWithGivenSource`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, s) \mapsto \text{CoclosedMonoidalRightCoevaluationMorphismWithGivenSource}(a, b, s)$.

1.14.81 AddCoclosedMonoidalRightEvaluationMorphism (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalRightEvaluationMorphism(C, F)` (operation)
- ▷ `AddCoclosedMonoidalRightEvaluationMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalRightEvaluationMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{CoclosedMonoidalRightEvaluationMorphism}(a, b)$.

1.14.82 AddCoclosedMonoidalRightEvaluationMorphismWithGivenRange (for IsCapCategory, IsFunction)

- ▷ `AddCoclosedMonoidalRightEvaluationMorphismWithGivenRange(C, F)` (operation)
- ▷ `AddCoclosedMonoidalRightEvaluationMorphismWithGivenRange(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `CoclosedMonoidalRightEvaluationMorphismWithGivenRange`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, r) \mapsto \text{CoclosedMonoidalRightEvaluationMorphismWithGivenRange}(a, b, r)$.

1.14.83 AddInternalCoHomOnMorphisms (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomOnMorphisms(C, F) (operation)
- ▷ AddInternalCoHomOnMorphisms($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomOnMorphisms. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha, beta) \mapsto \text{InternalCoHomOnMorphisms}(alpha, beta)$.

1.14.84 AddInternalCoHomOnMorphismsWithGivenInternalCoHoms (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomOnMorphismsWithGivenInternalCoHoms(C, F) (operation)
- ▷ AddInternalCoHomOnMorphismsWithGivenInternalCoHoms($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomOnMorphismsWithGivenInternalCoHoms. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, alpha, beta, r) \mapsto \text{InternalCoHomOnMorphismsWithGivenInternalCoHoms}(s, alpha, beta, r)$.

1.14.85 AddInternalCoHomOnObjects (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomOnObjects(C, F) (operation)
- ▷ AddInternalCoHomOnObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomOnObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{InternalCoHomOnObjects}(a, b)$.

1.14.86 AddInternalCoHomTensorProductCompatibilityMorphism (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomTensorProductCompatibilityMorphism(C, F) (operation)
- ▷ AddInternalCoHomTensorProductCompatibilityMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomTensorProductCompatibilityMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (list) \mapsto \text{InternalCoHomTensorProductCompatibilityMorphism}(list)$.

1.14.87 AddInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects(C , F) (operation)
- ▷ AddInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomTensorProductCompatibilityMorphismWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (source, list, range) \mapsto \text{InternalCoHomTensorProductCompatibilityMorphismWithGivenObjects}(source, list, range)$.

1.14.88 AddInternalCoHomToTensorProductLeftAdjunctMorphism (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomToTensorProductLeftAdjunctMorphism(C , F) (operation)
- ▷ AddInternalCoHomToTensorProductLeftAdjunctMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomToTensorProductLeftAdjunctMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, c, f) \mapsto \text{InternalCoHomToTensorProductLeftAdjunctMorphism}(a, c, f)$.

1.14.89 AddInternalCoHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct(C , F) (operation)
- ▷ AddInternalCoHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, c, f, t) \mapsto \text{InternalCoHomToTensorProductLeftAdjunctMorphismWithGivenTensorProduct}(a, c, f, t)$.

1.14.90 AddInternalCoHomToTensorProductRightAdjunctMorphism (for IsCapCategory, IsFunction)

- ▷ AddInternalCoHomToTensorProductRightAdjunctMorphism(C , F) (operation)
- ▷ AddInternalCoHomToTensorProductRightAdjunctMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalCoHomToTensorProductRightAdjunctMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f) \mapsto \text{InternalCoHomToTensorProductRightAdjunctMorphism}(a, b, f)$.

1.14.91 AddInternalCoHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ `AddInternalCoHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct(C, F)` (operation)
- ▷ `AddInternalCoHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `InternalCoHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f, t) \mapsto \text{InternalCoHomToTensorProductRightAdjunctMorphismWithGivenTensorProduct}(a, b, f, t)$.

1.14.92 AddIsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit(C, F)` (operation)
- ▷ `AddIsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromCoDualObjectToInternalCoHomFromTensorUnit}(a)$.

1.14.93 AddIsomorphismFromInternalCoHomFromTensorUnitToCoDualObject (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromInternalCoHomFromTensorUnitToCoDualObject(C, F)` (operation)
- ▷ `AddIsomorphismFromInternalCoHomFromTensorUnitToCoDualObject(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromInternalCoHomFromTensorUnitToCoDualObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the compu-

tational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromInternalCoHomFromTensorUnitToCoDualObject}(a)$.

1.14.94 AddIsomorphismFromInternalCoHomToObject (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromInternalCoHomToObject(C, F)` (operation)
- ▷ `AddIsomorphismFromInternalCoHomToObject(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromInternalCoHomToObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromInternalCoHomToObject}(a)$.

1.14.95 AddIsomorphismFromInternalCoHomToObjectWithGivenInternalCoHom (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromInternalCoHomToObjectWithGivenInternalCoHom(C, F)` (operation)
- ▷ `AddIsomorphismFromInternalCoHomToObjectWithGivenInternalCoHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromInternalCoHomToObjectWithGivenInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, s) \mapsto \text{IsomorphismFromInternalCoHomToObjectWithGivenInternalCoHom}(a, s)$.

1.14.96 AddIsomorphismFromObjectToInternalCoHom (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromObjectToInternalCoHom(C, F)` (operation)
- ▷ `AddIsomorphismFromObjectToInternalCoHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromObjectToInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromObjectToInternalCoHom}(a)$.

1.14.97 AddIsomorphismFromObjectToInternalCoHomWithGivenInternalCoHom (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromObjectToInternalCoHomWithGivenInternalCoHom(C, F)` (operation)
- ▷ `AddIsomorphismFromObjectToInternalCoHomWithGivenInternalCoHom(C, F, weight)` (operation)

(operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromObjectToInternalCoHomWithGivenInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{IsomorphismFromObjectToInternalCoHomWithGivenInternalCoHom}(a, r)$.

1.14.98 AddMonoidalPostCoComposeMorphism (for IsCapCategory, IsFunction)

▷ `AddMonoidalPostCoComposeMorphism(C, F)` (operation)

▷ `AddMonoidalPostCoComposeMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPostCoComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{MonoidalPostCoComposeMorphism}(a, b, c)$.

1.14.99 AddMonoidalPostCoComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddMonoidalPostCoComposeMorphismWithGivenObjects(C, F)` (operation)

▷ `AddMonoidalPostCoComposeMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPostCoComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{MonoidalPostCoComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.100 AddMonoidalPreCoComposeMorphism (for IsCapCategory, IsFunction)

▷ `AddMonoidalPreCoComposeMorphism(C, F)` (operation)

▷ `AddMonoidalPreCoComposeMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPreCoComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{MonoidalPreCoComposeMorphism}(a, b, c)$.

1.14.101 AddMonoidalPreCoComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddMonoidalPreCoComposeMorphismWithGivenObjects(C, F)` (operation)

▷ `AddMonoidalPreCoComposeMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MonoidalPreCoComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{MonoidalPreCoComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.102 AddMorphismFromCoBidual (for IsCapCategory, IsFunction)

▷ `AddMorphismFromCoBidual(C, F)` (operation)

▷ `AddMorphismFromCoBidual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromCoBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{MorphismFromCoBidual}(a)$.

1.14.103 AddMorphismFromCoBidualWithGivenCoBidual (for IsCapCategory, IsFunction)

▷ `AddMorphismFromCoBidualWithGivenCoBidual(C, F)` (operation)

▷ `AddMorphismFromCoBidualWithGivenCoBidual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromCoBidualWithGivenCoBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, s) \mapsto \text{MorphismFromCoBidualWithGivenCoBidual}(a, s)$.

1.14.104 AddMorphismFromInternalCoHomToTensorProduct (for IsCapCategory, IsFunction)

▷ `AddMorphismFromInternalCoHomToTensorProduct(C, F)` (operation)

▷ `AddMorphismFromInternalCoHomToTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromInternalCoHomToTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{MorphismFromInternalCoHomToTensorProduct}(a, b)$.

1.14.105 AddMorphismFromInternalCoHomToTensorProductWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddMorphismFromInternalCoHomToTensorProductWithGivenObjects(C, F)` (operation)

▷ `AddMorphismFromInternalCoHomToTensorProductWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromInternalCoHomToTensorProductWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{MorphismFromInternalCoHomToTensorProductWithGivenObjects}(s, a, b, r)$.

1.14.106 **AddTensorProductToInternalCoHomLeftAdjunctMorphism (for IsCapCategory, IsFunction)**

- ▷ `AddTensorProductToInternalCoHomLeftAdjunctMorphism(C, F)` (operation)
- ▷ `AddTensorProductToInternalCoHomLeftAdjunctMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductToInternalCoHomLeftAdjunctMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g) \mapsto \text{TensorProductToInternalCoHomLeftAdjunctMorphism}(b, c, g)$.

1.14.107 **AddTensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom (for IsCapCategory, IsFunction)**

- ▷ `AddTensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom(C, F)` (operation)
- ▷ `AddTensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g, i) \mapsto \text{TensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom}(b, c, g, i)$.

1.14.108 **AddTensorProductToInternalCoHomRightAdjunctMorphism (for IsCapCategory, IsFunction)**

- ▷ `AddTensorProductToInternalCoHomRightAdjunctMorphism(C, F)` (operation)
- ▷ `AddTensorProductToInternalCoHomRightAdjunctMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductToInternalCoHomRightAdjunctMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g) \mapsto \text{TensorProductToInternalCoHomRightAdjunctMorphism}(b, c, g)$.

1.14.109 AddTensorProductToInternalCoHomRightAdjunctMorphismWithGivenInternalCoHom (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToInternalCoHomRightAdjunctMorphismWithGivenInternalCoHom(C , F) (operation)
- ▷ AddTensorProductToInternalCoHomRightAdjunctMorphismWithGivenInternalCoHom(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToInternalCoHomRightAdjunctMorphismWithGivenInternalCoHom. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g, i) \mapsto \text{TensorProductToInternalCoHomRightAdjunctMorphismWithGivenInternalCoHom}(b, c, g, i)$.

1.14.110 AddUniversalPropertyOfCoDual (for IsCapCategory, IsFunction)

- ▷ AddUniversalPropertyOfCoDual(C , F) (operation)
- ▷ AddUniversalPropertyOfCoDual(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation UniversalPropertyOfCoDual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (t, a, alpha) \mapsto \text{UniversalPropertyOfCoDual}(t, a, alpha)$.

1.14.111 AddIsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit (for IsCapCategory, IsFunction)

- ▷ AddIsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit(C , F) (operation)
- ▷ AddIsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation IsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromLeftDualObjectToLeftInternalHomIntoTensorUnit}(a)$.

1.14.112 AddIsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject (for IsCapCategory, IsFunction)

- ▷ AddIsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject(C , F) (operation)
- ▷ AddIsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromLeftInternalHomIntoTensorUnitToLeftDualObject}(a)$.

1.14.113 AddIsomorphismFromLeftInternalHomToObject (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromLeftInternalHomToObject(C, F)` (operation)
- ▷ `AddIsomorphismFromLeftInternalHomToObject(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromLeftInternalHomToObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromLeftInternalHomToObject}(a)$.

1.14.114 AddIsomorphismFromLeftInternalHomToObjectWithGivenLeftInternalHom (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromLeftInternalHomToObjectWithGivenLeftInternalHom(C, F)` (operation)
- ▷ `AddIsomorphismFromLeftInternalHomToObjectWithGivenLeftInternalHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromLeftInternalHomToObjectWithGivenLeftInternalHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, s) \mapsto \text{IsomorphismFromLeftInternalHomToObjectWithGivenLeftInternalHom}(a, s)$.

1.14.115 AddIsomorphismFromObjectToLeftInternalHom (for IsCapCategory, IsFunction)

- ▷ `AddIsomorphismFromObjectToLeftInternalHom(C, F)` (operation)
- ▷ `AddIsomorphismFromObjectToLeftInternalHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromObjectToLeftInternalHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromObjectToLeftInternalHom}(a)$.

1.14.116 AddIsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom (for IsCapCategory, IsFunction)

- ▷ AddIsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom(C, F) (operation)
- ▷ AddIsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{IsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom}(a, r)$.

1.14.117 AddLeftClosedMonoidalCoevaluationMorphism (for IsCapCategory, IsFunction)

- ▷ AddLeftClosedMonoidalCoevaluationMorphism(C, F) (operation)
- ▷ AddLeftClosedMonoidalCoevaluationMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalCoevaluationMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{LeftClosedMonoidalCoevaluationMorphism}(a, b)$.

1.14.118 AddLeftClosedMonoidalCoevaluationMorphismWithGivenRange (for IsCapCategory, IsFunction)

- ▷ AddLeftClosedMonoidalCoevaluationMorphismWithGivenRange(C, F) (operation)
- ▷ AddLeftClosedMonoidalCoevaluationMorphismWithGivenRange($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalCoevaluationMorphismWithGivenRange`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, r) \mapsto \text{LeftClosedMonoidalCoevaluationMorphismWithGivenRange}(a, b, r)$.

1.14.119 AddLeftClosedMonoidalEvaluationForLeftDual (for IsCapCategory, IsFunction)

- ▷ AddLeftClosedMonoidalEvaluationForLeftDual(C, F) (operation)
- ▷ AddLeftClosedMonoidalEvaluationForLeftDual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalEvaluationForLeftDual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computa-

tional complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{LeftClosedMonoidalEvaluationForLeftDual}(a)$.

1.14.120 AddLeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct(C, F)` (operation)

▷ `AddLeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, r) \mapsto \text{LeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct}(s, a, r)$.

1.14.121 AddLeftClosedMonoidalEvaluationMorphism (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalEvaluationMorphism(C, F)` (operation)

▷ `AddLeftClosedMonoidalEvaluationMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalEvaluationMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{LeftClosedMonoidalEvaluationMorphism}(a, b)$.

1.14.122 AddLeftClosedMonoidalEvaluationMorphismWithGivenSource (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalEvaluationMorphismWithGivenSource(C, F)` (operation)

▷ `AddLeftClosedMonoidalEvaluationMorphismWithGivenSource(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalEvaluationMorphismWithGivenSource`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, s) \mapsto \text{LeftClosedMonoidalEvaluationMorphismWithGivenSource}(a, b, s)$.

1.14.123 AddLeftClosedMonoidalLambdaElimination (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalLambdaElimination(C, F)` (operation)

▷ `AddLeftClosedMonoidalLambdaElimination(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalLambdaElimination`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, \alpha) \mapsto \text{LeftClosedMonoidalLambdaElimination}(a, b, \alpha)$.

1.14.124 AddLeftClosedMonoidalLambdaIntroduction (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalLambdaIntroduction(C, F)` (operation)

▷ `AddLeftClosedMonoidalLambdaIntroduction(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalLambdaIntroduction`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (\alpha) \mapsto \text{LeftClosedMonoidalLambdaIntroduction}(\alpha)$.

1.14.125 AddLeftClosedMonoidalPostComposeMorphism (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalPostComposeMorphism(C, F)` (operation)

▷ `AddLeftClosedMonoidalPostComposeMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalPostComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{LeftClosedMonoidalPostComposeMorphism}(a, b, c)$.

1.14.126 AddLeftClosedMonoidalPostComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalPostComposeMorphismWithGivenObjects(C, F)` (operation)

▷ `AddLeftClosedMonoidalPostComposeMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalPostComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{LeftClosedMonoidalPostComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.127 AddLeftClosedMonoidalPreComposeMorphism (for IsCapCategory, IsFunction)

▷ `AddLeftClosedMonoidalPreComposeMorphism(C, F)` (operation)

▷ `AddLeftClosedMonoidalPreComposeMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalPreComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{LeftClosedMonoidalPreComposeMorphism}(a, b, c)$.

1.14.128 AddLeftClosedMonoidalPreComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ `AddLeftClosedMonoidalPreComposeMorphismWithGivenObjects(C, F)` (operation)
- ▷ `AddLeftClosedMonoidalPreComposeMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftClosedMonoidalPreComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{LeftClosedMonoidalPreComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.129 AddLeftDualOnMorphisms (for IsCapCategory, IsFunction)

- ▷ `AddLeftDualOnMorphisms(C, F)` (operation)
- ▷ `AddLeftDualOnMorphisms(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftDualOnMorphisms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha) \mapsto \text{LeftDualOnMorphisms}(alpha)$.

1.14.130 AddLeftDualOnMorphismsWithGivenLeftDuals (for IsCapCategory, IsFunction)

- ▷ `AddLeftDualOnMorphismsWithGivenLeftDuals(C, F)` (operation)
- ▷ `AddLeftDualOnMorphismsWithGivenLeftDuals(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftDualOnMorphismsWithGivenLeftDuals`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, alpha, r) \mapsto \text{LeftDualOnMorphismsWithGivenLeftDuals}(s, alpha, r)$.

1.14.131 AddLeftDualOnObjects (for IsCapCategory, IsFunction)

- ▷ `AddLeftDualOnObjects(C, F)` (operation)
- ▷ `AddLeftDualOnObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftDualOnObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{LeftDualOnObjects}(a)$.

1.14.132 AddLeftInternalHomOnMorphisms (for IsCapCategory, IsFunction)

- ▷ `AddLeftInternalHomOnMorphisms(C, F)` (operation)
- ▷ `AddLeftInternalHomOnMorphisms(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalHomOnMorphisms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (\alpha, \beta) \mapsto \text{LeftInternalHomOnMorphisms}(\alpha, \beta)$.

1.14.133 AddLeftInternalHomOnMorphismsWithGivenLeftInternalHoms (for IsCapCategory, IsFunction)

- ▷ `AddLeftInternalHomOnMorphismsWithGivenLeftInternalHoms(C, F)` (operation)
- ▷ `AddLeftInternalHomOnMorphismsWithGivenLeftInternalHoms(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalHomOnMorphismsWithGivenLeftInternalHoms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, \alpha, \beta, r) \mapsto \text{LeftInternalHomOnMorphismsWithGivenLeftInternalHoms}(s, \alpha, \beta, r)$.

1.14.134 AddLeftInternalHomOnObjects (for IsCapCategory, IsFunction)

- ▷ `AddLeftInternalHomOnObjects(C, F)` (operation)
- ▷ `AddLeftInternalHomOnObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalHomOnObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{LeftInternalHomOnObjects}(a, b)$.

1.14.135 AddLeftInternalHomToTensorProductAdjunctMorphism (for IsCapCategory, IsFunction)

- ▷ `AddLeftInternalHomToTensorProductAdjunctMorphism(C, F)` (operation)
- ▷ `AddLeftInternalHomToTensorProductAdjunctMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalHomToTensorProductAdjunctMorphism`. Op-

tionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g) \mapsto \text{LeftInternalHomToTensorProductAdjunctMorphism}(b, c, g)$.

1.14.136 AddLeftInternalHomToTensorProductAdjunctMorphismWithGivenTensorProduct (for IsCapCategory, IsFunction)

▷ `AddLeftInternalHomToTensorProductAdjunctMorphismWithGivenTensorProduct(C, F)` (operation)

▷ `AddLeftInternalHomToTensorProductAdjunctMorphismWithGivenTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalHomToTensorProductAdjunctMorphismWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g, t) \mapsto \text{LeftInternalHomToTensorProductAdjunctMorphismWithGivenTensorProduct}(b, c, g, t)$.

1.14.137 AddMorphismFromTensorProductToLeftInternalHom (for IsCapCategory, IsFunction)

▷ `AddMorphismFromTensorProductToLeftInternalHom(C, F)` (operation)

▷ `AddMorphismFromTensorProductToLeftInternalHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromTensorProductToLeftInternalHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{MorphismFromTensorProductToLeftInternalHom}(a, b)$.

1.14.138 AddMorphismFromTensorProductToLeftInternalHomWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddMorphismFromTensorProductToLeftInternalHomWithGivenObjects(C, F)` (operation)

▷ `AddMorphismFromTensorProductToLeftInternalHomWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromTensorProductToLeftInternalHomWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{MorphismFromTensorProductToLeftInternalHomWithGivenObjects}(s, a, b, r)$.

1.14.139 AddMorphismToLeftBidual (for IsCapCategory, IsFunction)

- ▷ AddMorphismToLeftBidual(C, F) (operation)
- ▷ AddMorphismToLeftBidual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismToLeftBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{MorphismToLeftBidual}(a)$.

1.14.140 AddMorphismToLeftBidualWithGivenLeftBidual (for IsCapCategory, IsFunction)

- ▷ AddMorphismToLeftBidualWithGivenLeftBidual(C, F) (operation)
- ▷ AddMorphismToLeftBidualWithGivenLeftBidual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismToLeftBidualWithGivenLeftBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{MorphismToLeftBidualWithGivenLeftBidual}(a, r)$.

1.14.141 AddTensorProductLeftDualityCompatibilityMorphism (for IsCapCategory, IsFunction)

- ▷ AddTensorProductLeftDualityCompatibilityMorphism(C, F) (operation)
- ▷ AddTensorProductLeftDualityCompatibilityMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductLeftDualityCompatibilityMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{TensorProductLeftDualityCompatibilityMorphism}(a, b)$.

1.14.142 AddTensorProductLeftDualityCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddTensorProductLeftDualityCompatibilityMorphismWithGivenObjects(C, F) (operation)
- ▷ AddTensorProductLeftDualityCompatibilityMorphismWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductLeftDualityCompatibilityMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{TensorProductLeftDualityCompatibilityMorphismWithGivenObjects}(s, a, b, r)$.

1.14.143 AddTensorProductLeftInternalHomCompatibilityMorphism (for IsCapCategory, IsFunction)

▷ AddTensorProductLeftInternalHomCompatibilityMorphism(C , F) (operation)

▷ AddTensorProductLeftInternalHomCompatibilityMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductLeftInternalHomCompatibilityMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (list) \mapsto \text{TensorProductLeftInternalHomCompatibilityMorphism}(list)$.

1.14.144 AddTensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

▷ AddTensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects(C , F) (operation)

▷ AddTensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (source, list, range) \mapsto \text{TensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects}(source, list, range)$.

1.14.145 AddTensorProductToLeftInternalHomAdjunctMorphism (for IsCapCategory, IsFunction)

▷ AddTensorProductToLeftInternalHomAdjunctMorphism(C , F) (operation)

▷ AddTensorProductToLeftInternalHomAdjunctMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToLeftInternalHomAdjunctMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f) \mapsto \text{TensorProductToLeftInternalHomAdjunctMorphism}(a, b, f)$.

1.14.146 AddTensorProductToLeftInternalHomAdjunctMorphismWithGivenLeftInternalHom (for IsCapCategory, IsFunction)

▷ AddTensorProductToLeftInternalHomAdjunctMorphismWithGivenLeftInternalHom(C , F) (operation)

▷ AddTensorProductToLeftInternalHomAdjunctMorphismWithGivenLeftInternalHom(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductToLeftInternalHomAdjunctMorphismWithGivenLeftInternalHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, f, i) \mapsto \text{TensorProductToLeftInternalHomAdjunctMorphismWithGivenLeftInternalHom}(a, b, f, i)$.

1.14.147 AddUniversalPropertyOfLeftDual (for IsCapCategory, IsFunction)

▷ `AddUniversalPropertyOfLeftDual(C, F)` (operation)

▷ `AddUniversalPropertyOfLeftDual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `UniversalPropertyOfLeftDual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (t, a, \alpha) \mapsto \text{UniversalPropertyOfLeftDual}(t, a, \alpha)$.

1.14.148 AddIsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit (for IsCapCategory, IsFunction)

▷ `AddIsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit(C, F)` (operation)

▷ `AddIsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromLeftCoDualObjectToLeftInternalCoHomFromTensorUnit}(a)$.

1.14.149 AddIsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject (for IsCapCategory, IsFunction)

▷ `AddIsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject(C, F)` (operation)

▷ `AddIsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromLeftInternalCoHomFromTensorUnitToLeftCoDualObject}(a)$.

1.14.150 AddIsomorphismFromLeftInternalCoHomToObject (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromLeftInternalCoHomToObject(C, F) (operation)

▷ AddIsomorphismFromLeftInternalCoHomToObject($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromLeftInternalCoHomToObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromLeftInternalCoHomToObject}(a)$.

1.14.151 AddIsomorphismFromLeftInternalCoHomToObjectWithGivenLeftInternalCoHom (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromLeftInternalCoHomToObjectWithGivenLeftInternalCoHom(C, F) (operation)

▷ AddIsomorphismFromLeftInternalCoHomToObjectWithGivenLeftInternalCoHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromLeftInternalCoHomToObjectWithGivenLeftInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, s) \mapsto \text{IsomorphismFromLeftInternalCoHomToObjectWithGivenLeftInternalCoHom}(a, s)$.

1.14.152 AddIsomorphismFromObjectToLeftInternalCoHom (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromObjectToLeftInternalCoHom(C, F) (operation)

▷ AddIsomorphismFromObjectToLeftInternalCoHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromObjectToLeftInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{IsomorphismFromObjectToLeftInternalCoHom}(a)$.

1.14.153 AddIsomorphismFromObjectToLeftInternalCoHomWithGivenLeftInternalCoHom (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromObjectToLeftInternalCoHomWithGivenLeftInternalCoHom(C, F) (operation)

▷ AddIsomorphismFromObjectToLeftInternalCoHomWithGivenLeftInternalCoHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromObjectToLeftInternalCoHomWithGivenLeftInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{IsomorphismFromObjectToLeftInternalCoHomWithGivenLeftInternalCoHom}(a, r)$.

1.14.154 AddLeftCoDualOnMorphisms (for IsCapCategory, IsFunction)

▷ `AddLeftCoDualOnMorphisms(C, F)` (operation)

▷ `AddLeftCoDualOnMorphisms(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoDualOnMorphisms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha) \mapsto \text{LeftCoDualOnMorphisms}(alpha)$.

1.14.155 AddLeftCoDualOnMorphismsWithGivenLeftCoDuals (for IsCapCategory, IsFunction)

▷ `AddLeftCoDualOnMorphismsWithGivenLeftCoDuals(C, F)` (operation)

▷ `AddLeftCoDualOnMorphismsWithGivenLeftCoDuals(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoDualOnMorphismsWithGivenLeftCoDuals`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, alpha, r) \mapsto \text{LeftCoDualOnMorphismsWithGivenLeftCoDuals}(s, alpha, r)$.

1.14.156 AddLeftCoDualOnObjects (for IsCapCategory, IsFunction)

▷ `AddLeftCoDualOnObjects(C, F)` (operation)

▷ `AddLeftCoDualOnObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoDualOnObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{LeftCoDualOnObjects}(a)$.

1.14.157 AddLeftCoDualityTensorProductCompatibilityMorphism (for IsCapCategory, IsFunction)

▷ `AddLeftCoDualityTensorProductCompatibilityMorphism(C, F)` (operation)

▷ `AddLeftCoDualityTensorProductCompatibilityMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoDualityTensorProductCompatibilityMorphism`.

Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{LeftCoDualityTensorProductCompatibilityMorphism}(a, b)$.

1.14.158 AddLeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

▷ `AddLeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects(C, F)` (operation)

▷ `AddLeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{LeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects}(s, a, b, r)$.

1.14.159 AddLeftCoclosedMonoidalCoevaluationMorphism (for IsCapCategory, IsFunction)

▷ `AddLeftCoclosedMonoidalCoevaluationMorphism(C, F)` (operation)

▷ `AddLeftCoclosedMonoidalCoevaluationMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalCoevaluationMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{LeftCoclosedMonoidalCoevaluationMorphism}(a, b)$.

1.14.160 AddLeftCoclosedMonoidalCoevaluationMorphismWithGivenSource (for IsCapCategory, IsFunction)

▷ `AddLeftCoclosedMonoidalCoevaluationMorphismWithGivenSource(C, F)` (operation)

▷ `AddLeftCoclosedMonoidalCoevaluationMorphismWithGivenSource(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalCoevaluationMorphismWithGivenSource`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, s) \mapsto \text{LeftCoclosedMonoidalCoevaluationMorphismWithGivenSource}(a, b, s)$.

1.14.161 AddLeftCoclosedMonoidalEvaluationForLeftCoDual (for IsCapCategory, IsFunction)

▷ AddLeftCoclosedMonoidalEvaluationForLeftCoDual(C , F) (operation)

▷ AddLeftCoclosedMonoidalEvaluationForLeftCoDual(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftCoclosedMonoidalEvaluationForLeftCoDual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{LeftCoclosedMonoidalEvaluationForLeftCoDual}(a)$.

1.14.162 AddLeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct (for IsCapCategory, IsFunction)

▷ AddLeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct(C , F) (operation)

▷ AddLeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, r) \mapsto \text{LeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct}(s, a, r)$.

1.14.163 AddLeftCoclosedMonoidalEvaluationMorphism (for IsCapCategory, IsFunction)

▷ AddLeftCoclosedMonoidalEvaluationMorphism(C , F) (operation)

▷ AddLeftCoclosedMonoidalEvaluationMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftCoclosedMonoidalEvaluationMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{LeftCoclosedMonoidalEvaluationMorphism}(a, b)$.

1.14.164 AddLeftCoclosedMonoidalEvaluationMorphismWithGivenRange (for IsCapCategory, IsFunction)

▷ AddLeftCoclosedMonoidalEvaluationMorphismWithGivenRange(C , F) (operation)

▷ AddLeftCoclosedMonoidalEvaluationMorphismWithGivenRange(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftCoclosedMonoidalEvaluationMorphismWithGivenRange.

Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, r) \mapsto \text{LeftCoclosedMonoidalEvaluationMorphismWithGivenRange}(a, b, r)$.

1.14.165 AddLeftCoclosedMonoidalLambdaElimination (for IsCapCategory, IsFunction)

- ▷ `AddLeftCoclosedMonoidalLambdaElimination(C, F)` (operation)
- ▷ `AddLeftCoclosedMonoidalLambdaElimination(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalLambdaElimination`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, \alpha) \mapsto \text{LeftCoclosedMonoidalLambdaElimination}(a, b, \alpha)$.

1.14.166 AddLeftCoclosedMonoidalLambdaIntroduction (for IsCapCategory, IsFunction)

- ▷ `AddLeftCoclosedMonoidalLambdaIntroduction(C, F)` (operation)
- ▷ `AddLeftCoclosedMonoidalLambdaIntroduction(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalLambdaIntroduction`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (\alpha) \mapsto \text{LeftCoclosedMonoidalLambdaIntroduction}(\alpha)$.

1.14.167 AddLeftCoclosedMonoidalPostCoComposeMorphism (for IsCapCategory, IsFunction)

- ▷ `AddLeftCoclosedMonoidalPostCoComposeMorphism(C, F)` (operation)
- ▷ `AddLeftCoclosedMonoidalPostCoComposeMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalPostCoComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{LeftCoclosedMonoidalPostCoComposeMorphism}(a, b, c)$.

1.14.168 AddLeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ `AddLeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects(C, F)` (operation)
- ▷ `AddLeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{LeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.169 AddLeftCoclosedMonoidalPreCoComposeMorphism (for IsCapCategory, IsFunction)

- ▷ `AddLeftCoclosedMonoidalPreCoComposeMorphism(C, F)` (operation)
- ▷ `AddLeftCoclosedMonoidalPreCoComposeMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalPreCoComposeMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{LeftCoclosedMonoidalPreCoComposeMorphism}(a, b, c)$.

1.14.170 AddLeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ `AddLeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects(C, F)` (operation)
- ▷ `AddLeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{LeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects}(s, a, b, c, r)$.

1.14.171 AddLeftInternalCoHomOnMorphisms (for IsCapCategory, IsFunction)

- ▷ `AddLeftInternalCoHomOnMorphisms(C, F)` (operation)
- ▷ `AddLeftInternalCoHomOnMorphisms(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalCoHomOnMorphisms`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (\alpha, \beta) \mapsto \text{LeftInternalCoHomOnMorphisms}(\alpha, \beta)$.

1.14.172 AddLeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHoms (for IsCapCategory, IsFunction)

- ▷ AddLeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHoms(C, F) (operation)
- ▷ AddLeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHoms($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHoms. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, alpha, beta, r) \mapsto \text{LeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHoms}(s, alpha, beta, r)$.

1.14.173 AddLeftInternalCoHomOnObjects (for IsCapCategory, IsFunction)

- ▷ AddLeftInternalCoHomOnObjects(C, F) (operation)
- ▷ AddLeftInternalCoHomOnObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftInternalCoHomOnObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{LeftInternalCoHomOnObjects}(a, b)$.

1.14.174 AddLeftInternalCoHomTensorProductCompatibilityMorphism (for IsCapCategory, IsFunction)

- ▷ AddLeftInternalCoHomTensorProductCompatibilityMorphism(C, F) (operation)
- ▷ AddLeftInternalCoHomTensorProductCompatibilityMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftInternalCoHomTensorProductCompatibilityMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (list) \mapsto \text{LeftInternalCoHomTensorProductCompatibilityMorphism}(list)$.

1.14.175 AddLeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddLeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects(C, F) (operation)
- ▷ AddLeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation LeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects. Optionally, a

weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (source, list, range) \mapsto \text{LeftInternalCoHomTensorProductCompatibilityMorphismWithGivenObjects}(source, list, range)$.

1.14.176 AddLeftInternalCoHomToTensorProductAdjunctMorphism (for IsCapCategory, IsFunction)

- ▷ `AddLeftInternalCoHomToTensorProductAdjunctMorphism(C, F)` (operation)
- ▷ `AddLeftInternalCoHomToTensorProductAdjunctMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalCoHomToTensorProductAdjunctMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, c, f) \mapsto \text{LeftInternalCoHomToTensorProductAdjunctMorphism}(a, c, f)$.

1.14.177 AddLeftInternalCoHomToTensorProductAdjunctMorphismWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ `AddLeftInternalCoHomToTensorProductAdjunctMorphismWithGivenTensorProduct(C, F)` (operation)
- ▷ `AddLeftInternalCoHomToTensorProductAdjunctMorphismWithGivenTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftInternalCoHomToTensorProductAdjunctMorphismWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, c, f, t) \mapsto \text{LeftInternalCoHomToTensorProductAdjunctMorphismWithGivenTensorProduct}(a, c, f, t)$.

1.14.178 AddMorphismFromLeftCoBidual (for IsCapCategory, IsFunction)

- ▷ `AddMorphismFromLeftCoBidual(C, F)` (operation)
- ▷ `AddMorphismFromLeftCoBidual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromLeftCoBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{MorphismFromLeftCoBidual}(a)$.

1.14.179 AddMorphismFromLeftCoBidualWithGivenLeftCoBidual (for IsCapCategory, IsFunction)

- ▷ `AddMorphismFromLeftCoBidualWithGivenLeftCoBidual(C, F)` (operation)
- ▷ `AddMorphismFromLeftCoBidualWithGivenLeftCoBidual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromLeftCoBidualWithGivenLeftCoBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, s) \mapsto \text{MorphismFromLeftCoBidualWithGivenLeftCoBidual}(a, s)$.

1.14.180 **AddMorphismFromLeftInternalCoHomToTensorProduct (for IsCapCategory, IsFunction)**

▷ `AddMorphismFromLeftInternalCoHomToTensorProduct(C, F)` (operation)

▷ `AddMorphismFromLeftInternalCoHomToTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromLeftInternalCoHomToTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{MorphismFromLeftInternalCoHomToTensorProduct}(a, b)$.

1.14.181 **AddMorphismFromLeftInternalCoHomToTensorProductWithGivenObjects (for IsCapCategory, IsFunction)**

▷ `AddMorphismFromLeftInternalCoHomToTensorProductWithGivenObjects(C, F)` (operation)

▷ `AddMorphismFromLeftInternalCoHomToTensorProductWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromLeftInternalCoHomToTensorProductWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{MorphismFromLeftInternalCoHomToTensorProductWithGivenObjects}(s, a, b, r)$.

1.14.182 **AddTensorProductToLeftInternalCoHomAdjunctMorphism (for IsCapCategory, IsFunction)**

▷ `AddTensorProductToLeftInternalCoHomAdjunctMorphism(C, F)` (operation)

▷ `AddTensorProductToLeftInternalCoHomAdjunctMorphism(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `TensorProductToLeftInternalCoHomAdjunctMorphism`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g) \mapsto \text{TensorProductToLeftInternalCoHomAdjunctMorphism}(b, c, g)$.

1.14.183 AddTensorProductToLeftInternalCoHomAdjunctMorphismWithGivenLeftInternalCoHom (for IsCapCategory, IsFunction)

- ▷ AddTensorProductToLeftInternalCoHomAdjunctMorphismWithGivenLeftInternalCoHom(C, F) (operation)
- ▷ AddTensorProductToLeftInternalCoHomAdjunctMorphismWithGivenLeftInternalCoHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductToLeftInternalCoHomAdjunctMorphismWithGivenLeftInternalCoHom.

Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (b, c, g, i) \mapsto \text{TensorProductToLeftInternalCoHomAdjunctMorphismWithGivenLeftInternalCoHom}(b, c, g, i)$.

1.14.184 AddUniversalPropertyOfLeftCoDual (for IsCapCategory, IsFunction)

- ▷ AddUniversalPropertyOfLeftCoDual(C, F) (operation)
- ▷ AddUniversalPropertyOfLeftCoDual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation UniversalPropertyOfLeftCoDual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (t, a, alpha) \mapsto \text{UniversalPropertyOfLeftCoDual}(t, a, alpha)$.

1.14.185 AddAssociatorLeftToRight (for IsCapCategory, IsFunction)

- ▷ AddAssociatorLeftToRight(C, F) (operation)
- ▷ AddAssociatorLeftToRight($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation AssociatorLeftToRight. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{AssociatorLeftToRight}(a, b, c)$.

1.14.186 AddAssociatorLeftToRightWithGivenTensorProducts (for IsCapCategory, IsFunction)

- ▷ AddAssociatorLeftToRightWithGivenTensorProducts(C, F) (operation)
- ▷ AddAssociatorLeftToRightWithGivenTensorProducts($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation AssociatorLeftToRightWithGivenTensorProducts. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{AssociatorLeftToRightWithGivenTensorProducts}(s, a, b, c, r)$.

1.14.187 AddAssociatorRightToLeft (for IsCapCategory, IsFunction)

- ▷ AddAssociatorRightToLeft(C, F) (operation)
- ▷ AddAssociatorRightToLeft($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `AssociatorRightToLeft`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b, c) \mapsto \text{AssociatorRightToLeft}(a, b, c)$.

1.14.188 AddAssociatorRightToLeftWithGivenTensorProducts (for IsCapCategory, IsFunction)

- ▷ AddAssociatorRightToLeftWithGivenTensorProducts(C, F) (operation)
- ▷ AddAssociatorRightToLeftWithGivenTensorProducts($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `AssociatorRightToLeftWithGivenTensorProducts`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, c, r) \mapsto \text{AssociatorRightToLeftWithGivenTensorProducts}(s, a, b, c, r)$.

1.14.189 AddLeftUnitor (for IsCapCategory, IsFunction)

- ▷ AddLeftUnitor(C, F) (operation)
- ▷ AddLeftUnitor($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftUnitor`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{LeftUnitor}(a)$.

1.14.190 AddLeftUnitorInverse (for IsCapCategory, IsFunction)

- ▷ AddLeftUnitorInverse(C, F) (operation)
- ▷ AddLeftUnitorInverse($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftUnitorInverse`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{LeftUnitorInverse}(a)$.

1.14.191 AddLeftUnitorInverseWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ AddLeftUnitorInverseWithGivenTensorProduct(C, F) (operation)
- ▷ AddLeftUnitorInverseWithGivenTensorProduct($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftUnitorInverseWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{LeftUnitorInverseWithGivenTensorProduct}(a, r)$.

1.14.192 AddLeftUnitorWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ `AddLeftUnitorWithGivenTensorProduct(C, F)` (operation)
- ▷ `AddLeftUnitorWithGivenTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `LeftUnitorWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, s) \mapsto \text{LeftUnitorWithGivenTensorProduct}(a, s)$.

1.14.193 AddRightUnitor (for IsCapCategory, IsFunction)

- ▷ `AddRightUnitor(C, F)` (operation)
- ▷ `AddRightUnitor(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `RightUnitor`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{RightUnitor}(a)$.

1.14.194 AddRightUnitorInverse (for IsCapCategory, IsFunction)

- ▷ `AddRightUnitorInverse(C, F)` (operation)
- ▷ `AddRightUnitorInverse(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `RightUnitorInverse`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{RightUnitorInverse}(a)$.

1.14.195 AddRightUnitorInverseWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ `AddRightUnitorInverseWithGivenTensorProduct(C, F)` (operation)
- ▷ `AddRightUnitorInverseWithGivenTensorProduct(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `RightUnitorInverseWithGivenTensorProduct`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{RightUnitorInverseWithGivenTensorProduct}(a, r)$.

1.14.196 AddRightUnitorWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ AddRightUnitorWithGivenTensorProduct(C, F) (operation)
- ▷ AddRightUnitorWithGivenTensorProduct($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation RightUnitorWithGivenTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, s) \mapsto \text{RightUnitorWithGivenTensorProduct}(a, s)$.

1.14.197 AddTensorProductOnMorphismAndObject (for IsCapCategory, IsFunction)

- ▷ AddTensorProductOnMorphismAndObject(C, F) (operation)
- ▷ AddTensorProductOnMorphismAndObject($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductOnMorphismAndObject. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha, b) \mapsto \text{TensorProductOnMorphismAndObject}(alpha, b)$.

1.14.198 AddTensorProductOnMorphismAndObjectWithGivenTensorProducts (for IsCapCategory, IsFunction)

- ▷ AddTensorProductOnMorphismAndObjectWithGivenTensorProducts(C, F) (operation)
- ▷ AddTensorProductOnMorphismAndObjectWithGivenTensorProducts($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductOnMorphismAndObjectWithGivenTensorProducts. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, alpha, b, r) \mapsto \text{TensorProductOnMorphismAndObjectWithGivenTensorProducts}(s, alpha, b, r)$.

1.14.199 AddTensorProductOnMorphisms (for IsCapCategory, IsFunction)

- ▷ AddTensorProductOnMorphisms(C, F) (operation)
- ▷ AddTensorProductOnMorphisms($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductOnMorphisms. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha, beta) \mapsto \text{TensorProductOnMorphisms}(alpha, beta)$.

1.14.200 AddTensorProductOnMorphismsWithGivenTensorProducts (for IsCapCategory, IsFunction)

▷ AddTensorProductOnMorphismsWithGivenTensorProducts(C , F) (operation)

▷ AddTensorProductOnMorphismsWithGivenTensorProducts(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductOnMorphismsWithGivenTensorProducts. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, alpha, beta, r) \mapsto \text{TensorProductOnMorphismsWithGivenTensorProducts}(s, alpha, beta, r)$.

1.14.201 AddTensorProductOnObjectAndMorphism (for IsCapCategory, IsFunction)

▷ AddTensorProductOnObjectAndMorphism(C , F) (operation)

▷ AddTensorProductOnObjectAndMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductOnObjectAndMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, beta) \mapsto \text{TensorProductOnObjectAndMorphism}(a, beta)$.

1.14.202 AddTensorProductOnObjectAndMorphismWithGivenTensorProducts (for IsCapCategory, IsFunction)

▷ AddTensorProductOnObjectAndMorphismWithGivenTensorProducts(C , F) (operation)

▷ AddTensorProductOnObjectAndMorphismWithGivenTensorProducts(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductOnObjectAndMorphismWithGivenTensorProducts. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, beta, r) \mapsto \text{TensorProductOnObjectAndMorphismWithGivenTensorProducts}(s, a, beta, r)$.

1.14.203 AddTensorProductOnObjects (for IsCapCategory, IsFunction)

▷ AddTensorProductOnObjects(C , F) (operation)

▷ AddTensorProductOnObjects(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductOnObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (arg2, arg3) \mapsto \text{TensorProductOnObjects}(arg2, arg3)$.

1.14.204 AddTensorUnit (for IsCapCategory, IsFunction)

- ▷ AddTensorUnit(C, F) (operation)
- ▷ AddTensorUnit($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorUnit. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : () \mapsto \text{TensorUnit}()$.

1.14.205 AddCoevaluationForDual (for IsCapCategory, IsFunction)

- ▷ AddCoevaluationForDual(C, F) (operation)
- ▷ AddCoevaluationForDual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation CoevaluationForDual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{CoevaluationForDual}(a)$.

1.14.206 AddCoevaluationForDualWithGivenTensorProduct (for IsCapCategory, IsFunction)

- ▷ AddCoevaluationForDualWithGivenTensorProduct(C, F) (operation)
- ▷ AddCoevaluationForDualWithGivenTensorProduct($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation CoevaluationForDualWithGivenTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, r) \mapsto \text{CoevaluationForDualWithGivenTensorProduct}(s, a, r)$.

1.14.207 AddIsomorphismFromInternalHomToTensorProductWithDualObject (for IsCapCategory, IsFunction)

- ▷ AddIsomorphismFromInternalHomToTensorProductWithDualObject(C, F) (operation)
- ▷ AddIsomorphismFromInternalHomToTensorProductWithDualObject($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation IsomorphismFromInternalHomToTensorProductWithDualObject. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, b) \mapsto \text{IsomorphismFromInternalHomToTensorProductWithDualObject}(a, b)$.

1.14.208 AddIsomorphismFromTensorProductWithDualObjectToInternalHom (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromTensorProductWithDualObjectToInternalHom(C, F) (operation)
 ▷ AddIsomorphismFromTensorProductWithDualObjectToInternalHom($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation IsomorphismFromTensorProductWithDualObjectToInternalHom. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b) \mapsto \text{IsomorphismFromTensorProductWithDualObjectToInternalHom}(a,b)$.

1.14.209 AddMorphismFromBidual (for IsCapCategory, IsFunction)

▷ AddMorphismFromBidual(C, F) (operation)
 ▷ AddMorphismFromBidual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismFromBidual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{MorphismFromBidual}(a)$.

1.14.210 AddMorphismFromBidualWithGivenBidual (for IsCapCategory, IsFunction)

▷ AddMorphismFromBidualWithGivenBidual(C, F) (operation)
 ▷ AddMorphismFromBidualWithGivenBidual($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismFromBidualWithGivenBidual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,s) \mapsto \text{MorphismFromBidualWithGivenBidual}(a,s)$.

1.14.211 AddMorphismFromInternalHomToTensorProduct (for IsCapCategory, IsFunction)

▷ AddMorphismFromInternalHomToTensorProduct(C, F) (operation)
 ▷ AddMorphismFromInternalHomToTensorProduct($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismFromInternalHomToTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b) \mapsto \text{MorphismFromInternalHomToTensorProduct}(a,b)$.

1.14.212 AddMorphismFromInternalHomToTensorProductWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddMorphismFromInternalHomToTensorProductWithGivenObjects(C, F) (operation)
- ▷ AddMorphismFromInternalHomToTensorProductWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation MorphismFromInternalHomToTensorProductWithGivenObjects. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{MorphismFromInternalHomToTensorProductWithGivenObjects}(s, a, b, r)$.

1.14.213 AddRankMorphism (for IsCapCategory, IsFunction)

- ▷ AddRankMorphism(C, F) (operation)
- ▷ AddRankMorphism($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation RankMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{RankMorphism}(a)$.

1.14.214 AddTensorProductInternalHomCompatibilityMorphismInverse (for IsCapCategory, IsFunction)

- ▷ AddTensorProductInternalHomCompatibilityMorphismInverse(C, F) (operation)
- ▷ AddTensorProductInternalHomCompatibilityMorphismInverse($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductInternalHomCompatibilityMorphismInverse. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (list) \mapsto \text{TensorProductInternalHomCompatibilityMorphismInverse}(list)$.

1.14.215 AddTensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects (for IsCapCategory, IsFunction)

- ▷ AddTensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects(C, F) (operation)
- ▷ AddTensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects($C, F, weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects. Optionally,

a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (source, list, range) \mapsto \text{TensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects}(source, list, range)$.

1.14.216 AddTraceMap (for IsCapCategory, IsFunction)

- ▷ AddTraceMap(C , F) (operation)
- ▷ AddTraceMap(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation TraceMap. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha) \mapsto \text{TraceMap}(alpha)$.

1.14.217 AddCoRankMorphism (for IsCapCategory, IsFunction)

- ▷ AddCoRankMorphism(C , F) (operation)
- ▷ AddCoRankMorphism(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation CoRankMorphism. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{CoRankMorphism}(a)$.

1.14.218 AddCoTraceMap (for IsCapCategory, IsFunction)

- ▷ AddCoTraceMap(C , F) (operation)
- ▷ AddCoTraceMap(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation CoTraceMap. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (alpha) \mapsto \text{CoTraceMap}(alpha)$.

1.14.219 AddCoclosedCoevaluationForCoDual (for IsCapCategory, IsFunction)

- ▷ AddCoclosedCoevaluationForCoDual(C , F) (operation)
- ▷ AddCoclosedCoevaluationForCoDual(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation CoclosedCoevaluationForCoDual. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{CoclosedCoevaluationForCoDual}(a)$.

1.14.220 AddCoclosedCoevaluationForCoDualWithGivenTensorProduct (for IsCapCategory, IsFunction)

▷ AddCoclosedCoevaluationForCoDualWithGivenTensorProduct(C , F) (operation)

▷ AddCoclosedCoevaluationForCoDualWithGivenTensorProduct(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation CoclosedCoevaluationForCoDualWithGivenTensorProduct. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (s, a, r) \mapsto \text{CoclosedCoevaluationForCoDualWithGivenTensorProduct}(s, a, r)$.

1.14.221 AddInternalCoHomTensorProductCompatibilityMorphismInverse (for IsCapCategory, IsFunction)

▷ AddInternalCoHomTensorProductCompatibilityMorphismInverse(C , F) (operation)

▷ AddInternalCoHomTensorProductCompatibilityMorphismInverse(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomTensorProductCompatibilityMorphismInverse. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (list) \mapsto \text{InternalCoHomTensorProductCompatibilityMorphismInverse}(list)$.

1.14.222 AddInternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects (for IsCapCategory, IsFunction)

▷ AddInternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects(C , F) (operation)

▷ AddInternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects(C , F , $weight$) (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation InternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects.

Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (source, list, range) \mapsto \text{InternalCoHomTensorProductCompatibilityMorphismInverseWithGivenObjects}(source, list, range)$.

1.14.223 AddIsomorphismFromInternalCoHomToTensorProductWithCoDualObject (for IsCapCategory, IsFunction)

▷ AddIsomorphismFromInternalCoHomToTensorProductWithCoDualObject(C , F) (operation)

▷ AddIsomorphismFromInternalCoHomToTensorProductWithCoDualObject(C , F , $weight$) (operation)

(operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromInternalCoHomToTensorProductWithCoDualObject`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b) \mapsto \text{IsomorphismFromInternalCoHomToTensorProductWithCoDualObject}(a,b)$.

1.14.224 `AddIsomorphismFromTensorProductWithCoDualObjectToInternalCoHom` (for `IsCapCategory`, `IsFunction`)

- ▷ `AddIsomorphismFromTensorProductWithCoDualObjectToInternalCoHom(C, F)` (operation)
- ▷ `AddIsomorphismFromTensorProductWithCoDualObjectToInternalCoHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `IsomorphismFromTensorProductWithCoDualObjectToInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b) \mapsto \text{IsomorphismFromTensorProductWithCoDualObjectToInternalCoHom}(a,b)$.

1.14.225 `AddMorphismFromTensorProductToInternalCoHom` (for `IsCapCategory`, `IsFunction`)

- ▷ `AddMorphismFromTensorProductToInternalCoHom(C, F)` (operation)
- ▷ `AddMorphismFromTensorProductToInternalCoHom(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromTensorProductToInternalCoHom`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a,b) \mapsto \text{MorphismFromTensorProductToInternalCoHom}(a,b)$.

1.14.226 `AddMorphismFromTensorProductToInternalCoHomWithGivenObjects` (for `IsCapCategory`, `IsFunction`)

- ▷ `AddMorphismFromTensorProductToInternalCoHomWithGivenObjects(C, F)` (operation)
- ▷ `AddMorphismFromTensorProductToInternalCoHomWithGivenObjects(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismFromTensorProductToInternalCoHomWithGivenObjects`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational com-

plexity of the function (lower weight = less complex = faster execution). $F : (s, a, b, r) \mapsto \text{MorphismFromTensorProductToInternalCoHomWithGivenObjects}(s, a, b, r)$.

1.14.227 AddMorphismToCoBidual (for IsCapCategory, IsFunction)

- ▷ `AddMorphismToCoBidual(C, F)` (operation)
- ▷ `AddMorphismToCoBidual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismToCoBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a) \mapsto \text{MorphismToCoBidual}(a)$.

1.14.228 AddMorphismToCoBidualWithGivenCoBidual (for IsCapCategory, IsFunction)

- ▷ `AddMorphismToCoBidualWithGivenCoBidual(C, F)` (operation)
- ▷ `AddMorphismToCoBidualWithGivenCoBidual(C, F, weight)` (operation)

Returns: nothing

The arguments are a category C and a function F . This operation adds the given function F to the category for the basic operation `MorphismToCoBidualWithGivenCoBidual`. Optionally, a weight (default: 100) can be specified which should roughly correspond to the computational complexity of the function (lower weight = less complex = faster execution). $F : (a, r) \mapsto \text{MorphismToCoBidualWithGivenCoBidual}(a, r)$.

Chapter 2

Examples and Tests

2.1 Test functions

2.1.1 AdditiveMonoidalCategoriesTest

▷ `AdditiveMonoidalCategoriesTest(cat, a, L)` (function)

The arguments are

- a CAP category *cat*
- an object *a*
- a list *L* of objects

This function checks for every operation declared in `AdditiveMonoidalCategories.gd` if it is computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.2 TestBraidingForInvertibility (for IsCapCategory, IsCapCategoryObject, IsCapCategoryObject)

▷ `TestBraidingForInvertibility(cat, obj_1, obj_2)` (operation)

Returns: a boolean

This is a debug operation. The arguments are three objects *obj₁*, *obj₂* in a braided monoidal category *cat*. The output is true if the braiding is invertible, false otherwise.

2.1.3 TestBraidingCompatibility (for IsCapCategory, IsCapCategoryObject, IsCapCategoryObject)

▷ TestBraidingCompatibility(*cat*, *obj_1*, *obj_2*, *obj_3*) (operation)

Returns: a boolean

This is a debug operation. The arguments are three objects *obj₁*, *obj₂*, *obj₃* in a braided monoidal category *cat*. The output is true if the braiding compatibilities with the associator hold, false otherwise.

2.1.4 TestBraidingCompatibilityForAllTriplesInList (for IsCapCategory, IsList)

▷ TestBraidingCompatibilityForAllTriplesInList(*cat*, *L*) (operation)

Returns: a boolean

This is a debug operation. The argument is a list *L* of objects in a braided monoidal category *cat*. The output is true if the braiding compatibilities with the associator hold for all triples of objects in *L*, otherwise false.

2.1.5 BraidedMonoidalCategoriesTest

▷ BraidedMonoidalCategoriesTest(*cat*, *a*, *b*) (function)

The arguments are

- a CAP category *cat*
- objects *a*, *b*

This function checks for every operation declared in BraidedMonoidalCategories.gd if it is computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to Opposite(), to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer dual_pre/postprocessor_funcs.

2.1.6 ClosedMonoidalCategoriesTest

▷ ClosedMonoidalCategoriesTest(*cat*, *a*, *b*, *c*, *d*, *alpha*, *beta*, *gamma*, *delta*, *epsilon*, *zeta*) (function)

The arguments are

- a CAP category *cat*
- objects *a*, *b*, *c*, *d*
- a morphism $\alpha : a \rightarrow b$

- a morphism $\beta : c \rightarrow d$
- a morphism $\gamma : a \otimes b \rightarrow 1$
- a morphism $\delta : c \otimes d \rightarrow 1$
- a morphism $\varepsilon : 1 \rightarrow \text{Hom}(a, b)$
- a morphism $\zeta : 1 \rightarrow \text{Hom}(c, d)$

This function checks for every operation declared in `ClosedMonoidalCategories.gd` if it is computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.7 ClosedMonoidalCategoriesTestWithGiven

▷ `ClosedMonoidalCategoriesTestWithGiven(cat, a, b, c, d, alpha, beta)` (function)

The arguments are

- a CAP category *cat*
- objects a, b, c, d
- a morphism $\alpha : a \rightarrow b$
- a morphism $\beta : c \rightarrow d$

This function checks for some `*WithGiven` operations declared in `ClosedMonoidalCategories.gd` if they are computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.8 CoclosedMonoidalCategoriesTest

▷ `CoclosedMonoidalCategoriesTest(cat, a, b, c, d, alpha, beta, gamma, delta, epsilon, zeta)` (function)

The arguments are

- a CAP category *cat*
- objects a, b, c, d
- a morphism $\alpha : a \rightarrow b$
- a morphism $\beta : c \rightarrow d$
- a morphism $\gamma : 1 \rightarrow a \otimes b$
- a morphism $\delta : 1 \rightarrow c \otimes d$
- a morphism $\varepsilon : \text{coHom}(a, b) \rightarrow 1$
- a morphism $\zeta : \text{coHom}(c, d) \rightarrow 1$

This function checks for every operation declared in `CoclosedMonoidalCategories.gd` if it is computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.9 CoclosedMonoidalCategoriesTestWithGiven

▷ `CoclosedMonoidalCategoriesTestWithGiven(cat, a, b, c, d, alpha, beta, gamma, delta, epsilon, zeta)` (function)

The arguments are

- a CAP category *cat*
- objects a, b, c, d
- a morphism $\alpha : a \rightarrow b$
- a morphism $\beta : c \rightarrow d$

This function checks for some `*WithGiven` operations declared in `CoclosedMonoidalCategories.gd` if they are computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in `cat`. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.10 LeftClosedMonoidalCategoriesTest

▷ `LeftClosedMonoidalCategoriesTest(cat, a, b, c, d, alpha, beta, gamma, delta, epsilon, zeta)` (function)

The arguments are

- a CAP category `cat`
- objects a, b, c, d
- a morphism $\alpha : a \rightarrow b$
- a morphism $\beta : c \rightarrow d$
- a morphism $\gamma : a \otimes b \rightarrow 1$
- a morphism $\delta : c \otimes d \rightarrow 1$
- a morphism $\varepsilon : 1 \rightarrow \text{Hom}(a, b)$
- a morphism $\zeta : 1 \rightarrow \text{Hom}(c, d)$

This function checks for every operation declared in `LeftClosedMonoidalCategories.gd` if it is computable in the CAP category `cat`. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of `cat`. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in `cat`. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.11 LeftClosedMonoidalCategoriesTestWithGiven

▷ `LeftClosedMonoidalCategoriesTestWithGiven(cat, a, b, c, d, alpha, beta, gamma, delta, epsilon, zeta)` (function)

The arguments are

- a CAP category `cat`
- objects a, b, c, d
- a morphism $\alpha : a \rightarrow b$

- a morphism $\beta : c \rightarrow d$

This function checks for some `*WithGiven operationS` declared in `LeftClosedMonoidalCategories.gd` if they are computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.12 LeftCoclosedMonoidalCategoriesTest

▷ `LeftCoclosedMonoidalCategoriesTest(cat, a, b, c, d, alpha, beta, gamma, delta, epsilon, zeta)` (function)

The arguments are

- a CAP category *cat*
- objects a, b, c, d
- a morphism $\alpha : a \rightarrow b$
- a morphism $\beta : c \rightarrow d$
- a morphism $\gamma : 1 \rightarrow a \otimes b$
- a morphism $\delta : 1 \rightarrow c \otimes d$
- a morphism $\varepsilon : \text{coHom}(a, b) \rightarrow 1$
- a morphism $\zeta : \text{coHom}(c, d) \rightarrow 1$

This function checks for every operation declared in `LeftCoclosedMonoidalCategories.gd` if it is computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.13 LeftCoclosedMonoidalCategoriesTestWithGiven

▷ `LeftCoclosedMonoidalCategoriesTestWithGiven(cat, a, b, c, d, alpha, beta, gamma, delta, epsilon, zeta)` (function)

The arguments are

- a CAP category *cat*
- objects a, b, c, d
- a morphism $\alpha : a \rightarrow b$
- a morphism $\beta : c \rightarrow d$

This function checks for some `*WithGiven` operations declared in `LeftCoclosedMonoidalCategories.gd` if they are computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.14 MonoidalCategoriesTensorProductOnObjectsAndTensorUnitTest

▷ `MonoidalCategoriesTensorProductOnObjectsAndTensorUnitTest(cat, a, b)` (function)

The arguments are

- a CAP category *cat*
- objects a, b

This function checks for every operation declared in `MonoidalCategoriesTensorProductOnObjectsAndTensorUnit.gd` if it is computable in the CAP category *cat*. If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of *cat*. Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in *cat*. The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.15 TestMonoidalUnitorsForInvertibility (for IsCapCategory, IsCapCategoryObject)

▷ TestMonoidalUnitorsForInvertibility(*cat*, *obj*) (operation)

Returns: a boolean

This is a debug operation. The arguments are two objects *obj* in a monoidal category *cat*. The output is true if the left and right unitors are invertible for *obj*.

2.1.16 TestAssociatorForInvertibility (for IsCapCategory, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ TestAssociatorForInvertibility(*cat*, *obj_1*, *obj_2*, *obj_3*) (operation)

Returns: a boolean

This is a debug operation. The arguments are two objects *obj₁*, *obj₂*, *obj₃* in a monoidal category *cat*. The output is true if the associator are invertible for these 3 objects, false otherwise.

2.1.17 TestMonoidalTriangleIdentity (for IsCapCategory, IsCapCategoryObject, IsCapCategoryObject)

▷ TestMonoidalTriangleIdentity(*cat*, *obj_1*, *obj_2*) (operation)

Returns: a boolean

This is a debug operation. The arguments are two objects *obj₁*, *obj₂* in a monoidal category *cat*. The output is true if the triangle identity holds for these 2 objects, false otherwise.

2.1.18 TestMonoidalTriangleIdentityForAllPairsInList (for IsCapCategory, IsList)

▷ TestMonoidalTriangleIdentityForAllPairsInList(*cat*, *L*) (operation)

Returns: a boolean

This is a debug operation. The argument is a list *L* of objects in a monoidal category *cat*. The output is true if the triangle identity holds for all pairs of objects in *L*, otherwise false.

2.1.19 TestMonoidalPentagonIdentity (for IsCapCategory, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ TestMonoidalPentagonIdentity(*cat*, *obj_1*, *obj_2*, *obj_3*, *obj_4*) (operation)

Returns: a boolean

This is a debug operation. The arguments are 4 objects *obj₁*, *obj₂*, *obj₃*, *obj₄* in a monoidal category *cat*. The output is true if the pentagon identity holds for these 4 objects, false otherwise.

2.1.20 TestMonoidalPentagonIdentityUsingWithGivenOperations (for IsCapCategory, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ TestMonoidalPentagonIdentityUsingWithGivenOperations(*cat*, *obj_1*, *obj_2*, *obj_3*, *obj_4*) (operation)

Returns: a boolean

This is a debug operation. The arguments are 4 objects $obj_1, obj_2, obj_3, obj_4$ in a monoidal category cat . The output is true if the pentagon identity holds for these 4 objects, false otherwise. This test uses the WithGiven-operations.

2.1.21 TestMonoidalPentagonIdentityForAllQuadruplesInList (for IsCapCategory, IsList)

▷ TestMonoidalPentagonIdentityForAllQuadruplesInList(cat, L) (operation)

Returns: a boolean

This is a debug operation. The argument is a list L of objects in a monoidal category cat . The output is true if the pentagon identity holds for all quadruples of objects in L , otherwise false.

2.1.22 MonoidalCategoriesTest

▷ MonoidalCategoriesTest($cat, a, b, c, alpha, beta$) (function)

The arguments are

- a CAP category cat
- objects a, b, c
- a morphism $\alpha : a \rightarrow b$
- a morphism $\beta : c \rightarrow d$

This function checks for every operation declared in MonoidalCategories.gd if it is computable in the CAP category cat . If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of cat . Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to Opposite(), to only primitively install dual operations for primitively installed operations in cat . The advantage is, that more derivations might be tested. On the downside, this might test fewer dual_pre/postprocessor_funcs.

2.1.23 TestZigZagIdentitiesForDual (for IsCapCategory, IsCapCategoryObject)

▷ TestZigZagIdentitiesForDual(cat, obj) (operation)

Returns: a boolean

This is a debug operation. The argument is an object obj in a rigid symmetric monoidal category cat . The output is true if the zig zag identity for duals hold, false otherwise.

2.1.24 RigidSymmetricClosedMonoidalCategoriesTest

▷ RigidSymmetricClosedMonoidalCategoriesTest($cat, a, b, c, d, alpha$) (function)

The arguments are

- a CAP category cat

- objects a, b, c, d
- an endomorphism $\alpha : a \rightarrow a$

This function checks for every object and morphism declared in `RigidSymmetricClosedMonoidalCategories.gd` if it is computable in the CAP category cat . If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of cat . Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in cat . The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

2.1.25 RigidSymmetricCoclosedMonoidalCategoriesTest

▷ `RigidSymmetricCoclosedMonoidalCategoriesTest(cat, a, b, c, d, alpha)` (function)

The arguments are

- a CAP category cat
- objects a, b, c, d
- an endomorphism $\alpha : a \rightarrow a$

This function checks for every object and morphism declared in `RigidSymmetricCoclosedMonoidalCategories.gd` if it is computable in the CAP category cat . If yes, then the operation is executed with the parameters given above and compared to the equivalent computation in the opposite category of cat . Pass the options

- `verbose := true` to output more information.
- `only_primitive_operations := true`, which is passed on to `Opposite()`, to only primitively install dual operations for primitively installed operations in cat . The advantage is, that more derivations might be tested. On the downside, this might test fewer `dual_pre/postprocessor_funcs`.

Chapter 3

Code Generation for Monoidal Categories

3.1 Monoidal Categories

3.1.1 WriteFileForMonoidalStructure

▷ WriteFileForMonoidalStructure(*key_val_rec*, *package_name*, *files_rec*) (function)

Returns: nothing

This functions uses the dictionary *key_val_rec* to create a new monoidal structure. It generates the necessary files in the package *package_name* using the file-correspondence table *files_rec*. See the implementation for details.

3.2 Closed Monoidal Categories

3.2.1 WriteFileForClosedMonoidalStructure

▷ WriteFileForClosedMonoidalStructure(*key_val_rec*, *package_name*, *files_rec*) (function)

Returns: nothing

This functions uses the dictionary *key_val_rec* to create a new closed monoidal structure. It generates the necessary files in the package *package_name* using the file-correspondence table *files_rec*. See the implementation for details.

3.2.2 WriteFileForLeftClosedMonoidalStructure

▷ WriteFileForLeftClosedMonoidalStructure(*key_val_rec*, *package_name*, *files_rec*) (function)

Returns: nothing

This functions uses the dictionary *key_val_rec* to create a new left closed monoidal structure. It generates the necessary files in the package *package_name* using the file-correspondence table *files_rec*. See the implementation for details.

3.3 Coclosed Monoidal Categories

3.3.1 WriteFileForCoclosedMonoidalStructure

▷ `WriteFileForCoclosedMonoidalStructure(key_val_rec, package_name, files_rec)`
(function)

Returns: nothing

This functions uses the dictionary `key_val_rec` to create a new coclosed monoidal structure. It generates the necessary files in the package `package_name` using the file-correspondence table `files_rec`. See the implementation for details.

3.3.2 WriteFileForLeftCoclosedMonoidalStructure

▷ `WriteFileForLeftCoclosedMonoidalStructure(key_val_rec, package_name, files_rec)`
(function)

Returns: nothing

This functions uses the dictionary `key_val_rec` to create a new left coclosed monoidal structure. It generates the necessary files in the package `package_name` using the file-correspondence table `files_rec`. See the implementation for details.

Chapter 4

The terminal category with multiple objects

This is an example of a category which is created using `CategoryConstructor` out of no input.

This category “lies” in all doctrines and can hence be used (in conjunction with `LazyCategory`) in order to check the type-correctness of the various derived methods provided by CAP or any CAP-based package.

4.1 Constructors

4.2 GAP Categories

Chapter 5

Legacy Operations and Synonyms

5.1 Legacy operations

5.1.1 `CoclosedCoevaluationMorphism` (for `IsCapCategoryObject`, `IsCapCategoryObject`)

▷ `CoclosedCoevaluationMorphism(a, b)` (operation)

This is a legacy operation for `CoclosedMonoidalLeftCoevaluationMorphism(b, a)`, i.e., with the first and second argument interchanged.

5.1.2 `CoclosedCoevaluationMorphismWithGivenSource` (for `IsCapCategoryObject`, `IsCapCategoryObject`, `IsCapCategoryObject`)

▷ `CoclosedCoevaluationMorphismWithGivenSource(a, b, s)` (operation)

This is a legacy operation for `CoclosedMonoidalLeftCoevaluationMorphismWithGivenSource(b, a, s)`, i.e., with the first and second argument interchanged.

5.1.3 `CoclosedEvaluationMorphism` (for `IsCapCategoryObject`, `IsCapCategoryObject`)

▷ `CoclosedEvaluationMorphism(a, b)` (operation)

This is a legacy operation for `CoclosedMonoidalLeftEvaluationMorphism(b, a)`, i.e., with the first and second argument interchanged.

5.1.4 `CoclosedEvaluationMorphismWithGivenRange` (for `IsCapCategoryObject`, `IsCapCategoryObject`, `IsCapCategoryObject`)

▷ `CoclosedEvaluationMorphismWithGivenRange(a, b, r)` (operation)

This is a legacy operation for `CoclosedMonoidalLeftEvaluationMorphismWithGivenRange(b, a, r)`, i.e., with the first and second argument interchanged.

5.1.5 CoevaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ `CoevaluationMorphism(a, b)` (operation)

This is a legacy operation for `ClosedMonoidalLeftCoevaluationMorphism(b, a)`, i.e., with the first and second argument interchanged.

5.1.6 CoevaluationMorphismWithGivenRange (for IsCapCategoryObject, IsCapCategoryObject)

▷ `CoevaluationMorphismWithGivenRange(a, b, r)` (operation)

This is a legacy operation for `ClosedMonoidalLeftCoevaluationMorphismWithGivenRange(b, a, r)`, i.e., with the first and second argument interchanged.

5.2 Synonyms for legacy operations

5.2.1 EvaluationMorphism (for IsCapCategoryObject, IsCapCategoryObject)

▷ `EvaluationMorphism(arg1, arg2)` (operation)

This is a synonym for `ClosedMonoidalLeftEvaluationMorphism`.

5.2.2 EvaluationMorphismWithGivenSource (for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject)

▷ `EvaluationMorphismWithGivenSource(arg1, arg2, arg3)` (operation)

This is a synonym for `ClosedMonoidalLeftEvaluationMorphismWithGivenSource`.

5.2.3 InternalCoHomToTensorProductAdjunctionMap (for IsObject)

▷ `InternalCoHomToTensorProductAdjunctionMap(arg)` (operation)

This is a synonym for `InternalCoHomToTensorProductLeftAdjunctMorphism`.

5.2.4 InternalCoHomToTensorProductAdjunctionMapWithGivenTensorProduct (for IsObject)

▷ `InternalCoHomToTensorProductAdjunctionMapWithGivenTensorProduct(arg)` (operation)

This is a synonym for `InternalCoHomToTensorProductLeftAdjunctionMapWithGivenTensorProduct`.

5.2.5 InternalHomToTensorProductAdjunctionMap (for IsObject)

▷ InternalHomToTensorProductAdjunctionMap(*arg*) (operation)

This is a synonym for InternalHomToTensorProductLeftAdjunctMorphism.

5.2.6 InternalHomToTensorProductAdjunctionMapWithGivenTensorProduct (for IsObject)

▷ InternalHomToTensorProductAdjunctionMapWithGivenTensorProduct(*arg*) (operation)

This is a synonym for InternalHomToTensorProductLeftAdjunctionMapWithGivenTensorProduct.

5.2.7 TensorProductToInternalCoHomAdjunctionMap (for IsObject)

▷ TensorProductToInternalCoHomAdjunctionMap(*arg*) (operation)

This is a synonym for TensorProductToInternalCoHomLeftAdjunctMorphism.

5.2.8 TensorProductToInternalCoHomAdjunctionMapWithGivenInternalCoHom (for IsObject)

▷ TensorProductToInternalCoHomAdjunctionMapWithGivenInternalCoHom(*arg*) (operation)

This is a synonym for TensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom.

5.2.9 TensorProductToInternalHomAdjunctionMap (for IsObject)

▷ TensorProductToInternalHomAdjunctionMap(*arg*) (operation)

This is a synonym for TensorProductToInternalHomLeftAdjunctMorphism.

5.2.10 TensorProductToInternalHomAdjunctionMapWithGivenInternalHom (for IsObject)

▷ TensorProductToInternalHomAdjunctionMapWithGivenInternalHom(*arg*) (operation)

This is a synonym for TensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom.

5.2.11 InternalCoHomToTensorProductLeftAdjunctionMap (for IsObject)

▷ InternalCoHomToTensorProductLeftAdjunctionMap(*arg*) (operation)

This is a synonym for InternalCoHomToTensorProductLeftAdjunctMorphism.

5.2.12 InternalHomToTensorProductLeftAdjunctionMap (for IsObject)

▷ `InternalHomToTensorProductLeftAdjunctionMap(arg)` (operation)

This is a synonym for `InternalHomToTensorProductLeftAdjunctMorphism`.

5.2.13 TensorProductToInternalCoHomLeftAdjunctionMap (for IsObject)

▷ `TensorProductToInternalCoHomLeftAdjunctionMap(arg)` (operation)

This is a synonym for `TensorProductToInternalCoHomLeftAdjunctMorphism`.

5.2.14 TensorProductToInternalCoHomLeftAdjunctionMapWithGivenInternalCoHom (for IsObject)

▷ `TensorProductToInternalCoHomLeftAdjunctionMapWithGivenInternalCoHom(arg)` (operation)

This is a synonym for `TensorProductToInternalCoHomLeftAdjunctMorphismWithGivenInternalCoHom`.

5.2.15 TensorProductToInternalHomLeftAdjunctionMap (for IsObject)

▷ `TensorProductToInternalHomLeftAdjunctionMap(arg)` (operation)

This is a synonym for `TensorProductToInternalHomLeftAdjunctMorphism`.

5.2.16 TensorProductToInternalHomLeftAdjunctionMapWithGivenInternalHom (for IsObject)

▷ `TensorProductToInternalHomLeftAdjunctionMapWithGivenInternalHom(arg)` (operation)

This is a synonym for `TensorProductToInternalHomLeftAdjunctMorphismWithGivenInternalHom`.

Chapter 6

MonoidalCategories automatic generated documentation

6.1 MonoidalCategories automatic generated documentation of properties

6.1.1 IsAdditiveMonoidalCategory (for IsCapCategory)

▷ `IsAdditiveMonoidalCategory(C)` (property)

Returns: true or false

The property of the category C being additive monoidal, i.e., where the tensor product preserves finite direct sums.

6.1.2 IsBraidedMonoidalCategory (for IsCapCategory)

▷ `IsBraidedMonoidalCategory(C)` (property)

Returns: true or false

The property of the category C being braided monoidal.

6.1.3 IsClosedMonoidalCategory (for IsCapCategory)

▷ `IsClosedMonoidalCategory(C)` (property)

Returns: true or false

The property of the category C being (bi)closed monoidal.

6.1.4 IsCoclosedMonoidalCategory (for IsCapCategory)

▷ `IsCoclosedMonoidalCategory(C)` (property)

Returns: true or false

The property of the category C being (bi)coclosed monoidal.

6.1.5 IsLeftClosedMonoidalCategory (for IsCapCategory)

▷ `IsLeftClosedMonoidalCategory(C)` (property)

Returns: true or false

The property of the category \mathcal{C} being left closed monoidal.

6.1.6 IsLeftCoclosedMonoidalCategory (for IsCapCategory)

- ▷ IsLeftCoclosedMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being coclosed monoidal.

6.1.7 IsMonoidalCategory (for IsCapCategory)

- ▷ IsMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being monoidal.

6.1.8 IsStrictMonoidalCategory (for IsCapCategory)

- ▷ IsStrictMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being strict monoidal.

6.1.9 IsRigidSymmetricClosedMonoidalCategory (for IsCapCategory)

- ▷ IsRigidSymmetricClosedMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being rigid symmetric closed monoidal.

6.1.10 IsRigidSymmetricCoclosedMonoidalCategory (for IsCapCategory)

- ▷ IsRigidSymmetricCoclosedMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being rigid symmetric coclosed monoidal.

6.1.11 IsSymmetricClosedMonoidalCategory (for IsCapCategory)

- ▷ IsSymmetricClosedMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being symmetric closed monoidal.

6.1.12 IsSymmetricCoclosedMonoidalCategory (for IsCapCategory)

- ▷ IsSymmetricCoclosedMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being symmetric coclosed monoidal.

6.1.13 IsSymmetricMonoidalCategory (for IsCapCategory)

- ▷ IsSymmetricMonoidalCategory(\mathcal{C}) (property)
Returns: true or false
 The property of the category \mathcal{C} being symmetric monoidal.

Index

- AddAssociatorLeftToRight
 - for IsCapCategory, IsFunction, [95](#)
 - for IsCapCategory, IsFunction, IsInt, [95](#)
- AddAssociatorLeftToRightWithGivenTensorProducts
 - for IsCapCategory, IsFunction, [95](#)
 - for IsCapCategory, IsFunction, IsInt, [95](#)
- AddAssociatorRightToLeft
 - for IsCapCategory, IsFunction, [96](#)
 - for IsCapCategory, IsFunction, IsInt, [96](#)
- AddAssociatorRightToLeftWithGivenTensorProducts
 - for IsCapCategory, IsFunction, [96](#)
 - for IsCapCategory, IsFunction, IsInt, [96](#)
- AddBraiding
 - for IsCapCategory, IsFunction, [48](#)
 - for IsCapCategory, IsFunction, IsInt, [48](#)
- AddBraidingInverse
 - for IsCapCategory, IsFunction, [49](#)
 - for IsCapCategory, IsFunction, IsInt, [49](#)
- AddBraidingInverseWithGivenTensorProducts
 - for IsCapCategory, IsFunction, [49](#)
 - for IsCapCategory, IsFunction, IsInt, [49](#)
- AddBraidingWithGivenTensorProducts
 - for IsCapCategory, IsFunction, [49](#)
 - for IsCapCategory, IsFunction, IsInt, [49](#)
- AddClosedMonoidalLeftCoevaluationMorphism
 - for IsCapCategory, IsFunction, [49](#)
 - for IsCapCategory, IsFunction, IsInt, [49](#)
- AddClosedMonoidalLeftCoevaluationMorphismWithGivenRange
 - for IsCapCategory, IsFunction, [50](#)
 - for IsCapCategory, IsFunction, IsInt, [50](#)
- AddClosedMonoidalLeftEvaluationMorphism
 - for IsCapCategory, IsFunction, IsInt, [50](#)
- AddClosedMonoidalLeftEvaluationMorphismWithGivenSource
 - for IsCapCategory, IsFunction, [50](#)
 - for IsCapCategory, IsFunction, IsInt, [50](#)
- AddClosedMonoidalRightCoevaluationMorphism
 - for IsCapCategory, IsFunction, [50](#)
 - for IsCapCategory, IsFunction, IsInt, [50](#)
- AddClosedMonoidalRightCoevaluationMorphismWithGivenRange
 - for IsCapCategory, IsFunction, [51](#)
 - for IsCapCategory, IsFunction, IsInt, [51](#)
- AddClosedMonoidalRightEvaluationMorphism
 - for IsCapCategory, IsFunction, [51](#)
 - for IsCapCategory, IsFunction, IsInt, [51](#)
- AddClosedMonoidalRightEvaluationMorphismWithGivenSource
 - for IsCapCategory, IsFunction, [51](#)
 - for IsCapCategory, IsFunction, IsInt, [51](#)
- AddCoclosedCoevaluationForCoDual
 - for IsCapCategory, IsFunction, [103](#)
 - for IsCapCategory, IsFunction, IsInt, [103](#)
- AddCoclosedCoevaluationForCoDualWithGivenTensorProduct
 - for IsCapCategory, IsFunction, [104](#)
 - for IsCapCategory, IsFunction, IsInt, [104](#)
- AddCoclosedEvaluationForCoDual
 - for IsCapCategory, IsFunction, [65](#)
 - for IsCapCategory, IsFunction, IsInt, [65](#)
- AddCoclosedEvaluationForCoDualWithGivenTensorProduct
 - for IsCapCategory, IsFunction, [65](#)
 - for IsCapCategory, IsFunction, IsInt, [65](#)
- AddCoclosedMonoidalLeftCoevaluationMorphism
 - for IsCapCategory, IsFunction, [65](#)

- for IsCapCategory, IsFunction, IsInt, [65](#)
- AddCoclosedMonoidalLeftCoevaluation-
MorphismWithGivenSource
 - for IsCapCategory, IsFunction, [66](#)
 - for IsCapCategory, IsFunction, IsInt, [66](#)
- AddCoclosedMonoidalLeftEvaluation-
Morphism
 - for IsCapCategory, IsFunction, [66](#)
 - for IsCapCategory, IsFunction, IsInt, [66](#)
- AddCoclosedMonoidalLeftEvaluation-
MorphismWithGivenRange
 - for IsCapCategory, IsFunction, [66](#)
 - for IsCapCategory, IsFunction, IsInt, [66](#)
- AddCoclosedMonoidalRightCoevaluation-
Morphism
 - for IsCapCategory, IsFunction, [66](#)
 - for IsCapCategory, IsFunction, IsInt, [66](#)
- AddCoclosedMonoidalRightCoevaluation-
MorphismWithGivenSource
 - for IsCapCategory, IsFunction, [67](#)
 - for IsCapCategory, IsFunction, IsInt, [67](#)
- AddCoclosedMonoidalRightEvaluation-
Morphism
 - for IsCapCategory, IsFunction, [67](#)
 - for IsCapCategory, IsFunction, IsInt, [67](#)
- AddCoclosedMonoidalRightEvaluation-
MorphismWithGivenRange
 - for IsCapCategory, IsFunction, [67](#)
 - for IsCapCategory, IsFunction, IsInt, [67](#)
- AddCoDualityTensorProduct-
CompatibilityMorphism
 - for IsCapCategory, IsFunction, [64](#)
 - for IsCapCategory, IsFunction, IsInt, [64](#)
- AddCoDualityTensorProduct-
CompatibilityMorphismWith-
GivenObjects
 - for IsCapCategory, IsFunction, [64](#)
 - for IsCapCategory, IsFunction, IsInt, [64](#)
- AddCoDualOnMorphisms
 - for IsCapCategory, IsFunction, [63](#)
 - for IsCapCategory, IsFunction, IsInt, [63](#)
- AddCoDualOnMorphismsWithGivenCoDuals
 - for IsCapCategory, IsFunction, [63](#)
 - for IsCapCategory, IsFunction, IsInt, [63](#)
- AddCoDualOnObjects
 - for IsCapCategory, IsFunction, [64](#)
- for IsCapCategory, IsFunction, IsInt, [64](#)
- AddCoevaluationForDual
 - for IsCapCategory, IsFunction, [100](#)
 - for IsCapCategory, IsFunction, IsInt, [100](#)
- AddCoevaluationForDualWithGivenTensor-
Product
 - for IsCapCategory, IsFunction, [100](#)
 - for IsCapCategory, IsFunction, IsInt, [100](#)
- AddCoLambdaElimination
 - for IsCapCategory, IsFunction, [64](#)
 - for IsCapCategory, IsFunction, IsInt, [64](#)
- AddCoLambdaIntroduction
 - for IsCapCategory, IsFunction, [65](#)
 - for IsCapCategory, IsFunction, IsInt, [65](#)
- AddCoRankMorphism
 - for IsCapCategory, IsFunction, [103](#)
 - for IsCapCategory, IsFunction, IsInt, [103](#)
- AddCoTraceMap
 - for IsCapCategory, IsFunction, [103](#)
 - for IsCapCategory, IsFunction, IsInt, [103](#)
- AddDualOnMorphisms
 - for IsCapCategory, IsFunction, [51](#)
 - for IsCapCategory, IsFunction, IsInt, [51](#)
- AddDualOnMorphismsWithGivenDuals
 - for IsCapCategory, IsFunction, [52](#)
 - for IsCapCategory, IsFunction, IsInt, [52](#)
- AddDualOnObjects
 - for IsCapCategory, IsFunction, [52](#)
 - for IsCapCategory, IsFunction, IsInt, [52](#)
- AddEvaluationForDual
 - for IsCapCategory, IsFunction, [52](#)
 - for IsCapCategory, IsFunction, IsInt, [52](#)
- AddEvaluationForDualWithGivenTensor-
Product
 - for IsCapCategory, IsFunction, [52](#)
 - for IsCapCategory, IsFunction, IsInt, [52](#)
- AddInternalCoHomOnMorphisms
 - for IsCapCategory, IsFunction, [68](#)
 - for IsCapCategory, IsFunction, IsInt, [68](#)
- AddInternalCoHomOnMorphismsWithGiven-
InternalCoHoms
 - for IsCapCategory, IsFunction, [68](#)
 - for IsCapCategory, IsFunction, IsInt, [68](#)
- AddInternalCoHomOnObjects
 - for IsCapCategory, IsFunction, [68](#)
 - for IsCapCategory, IsFunction, IsInt, [68](#)

AddInternalCoHomTensorProduct-
 CompatibilityMorphism
 for IsCapCategory, IsFunction, 68
 for IsCapCategory, IsFunction, IsInt, 68
 AddInternalCoHomTensorProduct-
 CompatibilityMorphismInverse
 for IsCapCategory, IsFunction, 104
 for IsCapCategory, IsFunction, IsInt, 104
 AddInternalCoHomTensorProduct-
 CompatibilityMorphismInverse-
 WithGivenObjects
 for IsCapCategory, IsFunction, 104
 for IsCapCategory, IsFunction, IsInt, 104
 AddInternalCoHomTensorProduct-
 CompatibilityMorphismWith-
 GivenObjects
 for IsCapCategory, IsFunction, 69
 for IsCapCategory, IsFunction, IsInt, 69
 AddInternalCoHomToTensorProductLeft-
 AdjunctMorphism
 for IsCapCategory, IsFunction, 69
 for IsCapCategory, IsFunction, IsInt, 69
 AddInternalCoHomToTensorProductLeft-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategory, IsFunction, 69
 for IsCapCategory, IsFunction, IsInt, 69
 AddInternalCoHomToTensorProductRight-
 AdjunctMorphism
 for IsCapCategory, IsFunction, 69
 for IsCapCategory, IsFunction, IsInt, 69
 AddInternalCoHomToTensorProductRight-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategory, IsFunction, 70
 for IsCapCategory, IsFunction, IsInt, 70
 AddInternalHomOnMorphisms
 for IsCapCategory, IsFunction, 52
 for IsCapCategory, IsFunction, IsInt, 52
 AddInternalHomOnMorphismsWithGiven-
 InternalHoms
 for IsCapCategory, IsFunction, 53
 for IsCapCategory, IsFunction, IsInt, 53
 AddInternalHomOnObjects
 for IsCapCategory, IsFunction, 53
 for IsCapCategory, IsFunction, IsInt, 53
 AddInternalHomToTensorProductLeft-
 AdjunctionIsomorphism
 for IsCapCategory, IsFunction, 54
 for IsCapCategory, IsFunction, IsInt, 54
 AddInternalHomToTensorProductLeft-
 AdjunctionIsomorphismWithGiven-
 Objects
 for IsCapCategory, IsFunction, 54
 for IsCapCategory, IsFunction, IsInt, 54
 AddInternalHomToTensorProductLeft-
 AdjunctMorphism
 for IsCapCategory, IsFunction, 53
 for IsCapCategory, IsFunction, IsInt, 53
 AddInternalHomToTensorProductLeft-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategory, IsFunction, 53
 for IsCapCategory, IsFunction, IsInt, 54
 AddInternalHomToTensorProductRight-
 AdjunctionIsomorphism
 for IsCapCategory, IsFunction, 55
 for IsCapCategory, IsFunction, IsInt, 55
 AddInternalHomToTensorProductRight-
 AdjunctionIsomorphismWithGiven-
 Objects
 for IsCapCategory, IsFunction, 55
 for IsCapCategory, IsFunction, IsInt, 55
 AddInternalHomToTensorProductRight-
 AdjunctMorphism
 for IsCapCategory, IsFunction, 54
 for IsCapCategory, IsFunction, IsInt, 54
 AddInternalHomToTensorProductRight-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategory, IsFunction, 55
 for IsCapCategory, IsFunction, IsInt, 55
 AddIsomorphismFromCoDualObjectTo-
 InternalCoHomFromTensorUnit
 for IsCapCategory, IsFunction, 70
 for IsCapCategory, IsFunction, IsInt, 70
 AddIsomorphismFromDualObjectTo-
 InternalHomIntoTensorUnit
 for IsCapCategory, IsFunction, 55
 for IsCapCategory, IsFunction, IsInt, 55
 AddIsomorphismFromInternalCoHomFrom-
 TensorUnitToCoDualObject

for IsCapCategory, IsFunction, [70](#)
 for IsCapCategory, IsFunction, IsInt, [70](#)
 AddIsomorphismFromInternalCoHomTo-
 Object
 for IsCapCategory, IsFunction, [71](#)
 for IsCapCategory, IsFunction, IsInt, [71](#)
 AddIsomorphismFromInternalCoHomTo-
 ObjectWithGivenInternalCoHom
 for IsCapCategory, IsFunction, [71](#)
 for IsCapCategory, IsFunction, IsInt, [71](#)
 AddIsomorphismFromInternalCoHomTo-
 TensorProductWithCoDualObject
 for IsCapCategory, IsFunction, [104](#)
 for IsCapCategory, IsFunction, IsInt, [104](#)
 AddIsomorphismFromInternalHomInto-
 TensorUnitToDualObject
 for IsCapCategory, IsFunction, [56](#)
 for IsCapCategory, IsFunction, IsInt, [56](#)
 AddIsomorphismFromInternalHomToObject
 for IsCapCategory, IsFunction, [56](#)
 for IsCapCategory, IsFunction, IsInt, [56](#)
 AddIsomorphismFromInternalHomToObject-
 WithGivenInternalHom
 for IsCapCategory, IsFunction, [56](#)
 for IsCapCategory, IsFunction, IsInt, [56](#)
 AddIsomorphismFromInternalHomToTensor-
 ProductWithDualObject
 for IsCapCategory, IsFunction, [100](#)
 for IsCapCategory, IsFunction, IsInt, [100](#)
 AddIsomorphismFromLeftCoDualObjectTo-
 LeftInternalCoHomFromTensorUnit
 for IsCapCategory, IsFunction, [85](#)
 for IsCapCategory, IsFunction, IsInt, [85](#)
 AddIsomorphismFromLeftDualObjectTo-
 LeftInternalHomIntoTensorUnit
 for IsCapCategory, IsFunction, [75](#)
 for IsCapCategory, IsFunction, IsInt, [75](#)
 AddIsomorphismFromLeftInternalCoHom-
 FromTensorUnitToLeftCoDual-
 Object
 for IsCapCategory, IsFunction, [85](#)
 for IsCapCategory, IsFunction, IsInt, [85](#)
 AddIsomorphismFromLeftInternalCoHomTo-
 Object
 for IsCapCategory, IsFunction, [86](#)
 for IsCapCategory, IsFunction, IsInt, [86](#)
 AddIsomorphismFromLeftInternalCoHom-
 ToObjectWithGivenLeftInternal-
 CoHom
 for IsCapCategory, IsFunction, [86](#)
 for IsCapCategory, IsFunction, IsInt, [86](#)
 AddIsomorphismFromLeftInternalHomInto-
 TensorUnitToLeftDualObject
 for IsCapCategory, IsFunction, [75](#)
 for IsCapCategory, IsFunction, IsInt, [75](#)
 AddIsomorphismFromLeftInternalHomTo-
 Object
 for IsCapCategory, IsFunction, [76](#)
 for IsCapCategory, IsFunction, IsInt, [76](#)
 AddIsomorphismFromLeftInternalHomTo-
 ObjectWithGivenLeftInternalHom
 for IsCapCategory, IsFunction, [76](#)
 for IsCapCategory, IsFunction, IsInt, [76](#)
 AddIsomorphismFromObjectToInternal-
 CoHom
 for IsCapCategory, IsFunction, [71](#)
 for IsCapCategory, IsFunction, IsInt, [71](#)
 AddIsomorphismFromObjectToInternalCo-
 HomWithGivenInternalCoHom
 for IsCapCategory, IsFunction, [71](#)
 for IsCapCategory, IsFunction, IsInt, [71](#)
 AddIsomorphismFromObjectToInternalHom
 for IsCapCategory, IsFunction, [57](#)
 for IsCapCategory, IsFunction, IsInt, [57](#)
 AddIsomorphismFromObjectToInternalHom-
 WithGivenInternalHom
 for IsCapCategory, IsFunction, [57](#)
 for IsCapCategory, IsFunction, IsInt, [57](#)
 AddIsomorphismFromObjectToLeft-
 InternalCoHom
 for IsCapCategory, IsFunction, [86](#)
 for IsCapCategory, IsFunction, IsInt, [86](#)
 AddIsomorphismFromObjectToLeft-
 InternalCoHomWithGivenLeft-
 InternalCoHom
 for IsCapCategory, IsFunction, [86](#)
 for IsCapCategory, IsFunction, IsInt, [86](#)
 AddIsomorphismFromObjectToLeft-
 InternalHom
 for IsCapCategory, IsFunction, [76](#)
 for IsCapCategory, IsFunction, IsInt, [76](#)
 AddIsomorphismFromObjectToLeft-

InternalHomWithGivenLeft- InternalHom	AddLeftClosedMonoidalLambda- Introduction
for IsCapCategory, IsFunction, 77	for IsCapCategory, IsFunction, 79
for IsCapCategory, IsFunction, IsInt, 77	for IsCapCategory, IsFunction, IsInt, 79
AddIsomorphismFromTensorProductWithCo- DualObjectToInternalCoHom	AddLeftClosedMonoidalPostCompose- Morphism
for IsCapCategory, IsFunction, 105	for IsCapCategory, IsFunction, 79
for IsCapCategory, IsFunction, IsInt, 105	for IsCapCategory, IsFunction, IsInt, 79
AddIsomorphismFromTensorProductWith- DualObjectToInternalHom	AddLeftClosedMonoidalPostCompose- MorphismWithGivenObjects
for IsCapCategory, IsFunction, 101	for IsCapCategory, IsFunction, 79
for IsCapCategory, IsFunction, IsInt, 101	for IsCapCategory, IsFunction, IsInt, 79
AdditiveMonoidalCategoriesTest, 107	AddLeftClosedMonoidalPreCompose- Morphism
AddLambdaElimination	for IsCapCategory, IsFunction, 79
for IsCapCategory, IsFunction, 57	for IsCapCategory, IsFunction, IsInt, 79
for IsCapCategory, IsFunction, IsInt, 57	AddLeftClosedMonoidalPreCompose- MorphismWithGivenObjects
AddLambdaIntroduction	for IsCapCategory, IsFunction, 80
for IsCapCategory, IsFunction, 57	for IsCapCategory, IsFunction, IsInt, 80
for IsCapCategory, IsFunction, IsInt, 57	AddLeftCoclosedMonoidalCoevaluation- Morphism
AddLeftClosedMonoidalCoevaluation- Morphism	for IsCapCategory, IsFunction, 88
for IsCapCategory, IsFunction, 77	for IsCapCategory, IsFunction, IsInt, 88
for IsCapCategory, IsFunction, IsInt, 77	AddLeftCoclosedMonoidalCoevaluation- MorphismWithGivenSource
AddLeftClosedMonoidalCoevaluation- MorphismWithGivenRange	for IsCapCategory, IsFunction, 88
for IsCapCategory, IsFunction, 77	for IsCapCategory, IsFunction, IsInt, 88
for IsCapCategory, IsFunction, IsInt, 77	AddLeftCoclosedMonoidalEvaluationFor- LeftCoDual
AddLeftClosedMonoidalEvaluationFor- LeftDual	for IsCapCategory, IsFunction, 89
for IsCapCategory, IsFunction, 77	for IsCapCategory, IsFunction, IsInt, 89
for IsCapCategory, IsFunction, IsInt, 77	AddLeftCoclosedMonoidalEvaluation- ForLeftCoDualWithGivenTensor- Product
AddLeftClosedMonoidalEvaluationFor- LeftDualWithGivenTensorProduct	for IsCapCategory, IsFunction, 89
for IsCapCategory, IsFunction, 78	for IsCapCategory, IsFunction, IsInt, 89
for IsCapCategory, IsFunction, IsInt, 78	AddLeftCoclosedMonoidalEvaluation- Morphism
AddLeftClosedMonoidalEvaluation- Morphism	for IsCapCategory, IsFunction, 89
for IsCapCategory, IsFunction, 78	for IsCapCategory, IsFunction, IsInt, 89
for IsCapCategory, IsFunction, IsInt, 78	AddLeftCoclosedMonoidalEvaluation- MorphismWithGivenRange
AddLeftClosedMonoidalEvaluation- MorphismWithGivenSource	for IsCapCategory, IsFunction, 89
for IsCapCategory, IsFunction, 78	for IsCapCategory, IsFunction, IsInt, 89
for IsCapCategory, IsFunction, IsInt, 78	AddLeftCoclosedMonoidalLambda-
AddLeftClosedMonoidalLambdaElimination	
for IsCapCategory, IsFunction, 78	
for IsCapCategory, IsFunction, IsInt, 78	

- Elimination
 - for IsCapCategory, IsFunction, 90
 - for IsCapCategory, IsFunction, IsInt, 90
- AddLeftCoclosedMonoidalLambda-Introduction
 - for IsCapCategory, IsFunction, 90
 - for IsCapCategory, IsFunction, IsInt, 90
- AddLeftCoclosedMonoidalPostCoCompose-Morphism
 - for IsCapCategory, IsFunction, 90
 - for IsCapCategory, IsFunction, IsInt, 90
- AddLeftCoclosedMonoidalPostCoCompose-MorphismWithGivenObjects
 - for IsCapCategory, IsFunction, 90
 - for IsCapCategory, IsFunction, IsInt, 90
- AddLeftCoclosedMonoidalPreCoCompose-Morphism
 - for IsCapCategory, IsFunction, 91
 - for IsCapCategory, IsFunction, IsInt, 91
- AddLeftCoclosedMonoidalPreCoCompose-MorphismWithGivenObjects
 - for IsCapCategory, IsFunction, 91
 - for IsCapCategory, IsFunction, IsInt, 91
- AddLeftCoDualityTensorProduct-CompatibilityMorphism
 - for IsCapCategory, IsFunction, 87
 - for IsCapCategory, IsFunction, IsInt, 87
- AddLeftCoDualityTensorProduct-CompatibilityMorphismWithGivenObjects
 - for IsCapCategory, IsFunction, 88
 - for IsCapCategory, IsFunction, IsInt, 88
- AddLeftCoDualOnMorphisms
 - for IsCapCategory, IsFunction, 87
 - for IsCapCategory, IsFunction, IsInt, 87
- AddLeftCoDualOnMorphismsWithGivenLeft-CoDuals
 - for IsCapCategory, IsFunction, 87
 - for IsCapCategory, IsFunction, IsInt, 87
- AddLeftCoDualOnObjects
 - for IsCapCategory, IsFunction, 87
 - for IsCapCategory, IsFunction, IsInt, 87
- AddLeftDistributivityExpanding
 - for IsCapCategory, IsFunction, 46
 - for IsCapCategory, IsFunction, IsInt, 46
- AddLeftDistributivityExpandingWith-
 - GivenObjects
 - for IsCapCategory, IsFunction, 47
 - for IsCapCategory, IsFunction, IsInt, 47
- AddLeftDistributivityFactoring
 - for IsCapCategory, IsFunction, 47
 - for IsCapCategory, IsFunction, IsInt, 47
- AddLeftDistributivityFactoringWith-GivenObjects
 - for IsCapCategory, IsFunction, 47
 - for IsCapCategory, IsFunction, IsInt, 47
- AddLeftDualOnMorphisms
 - for IsCapCategory, IsFunction, 80
 - for IsCapCategory, IsFunction, IsInt, 80
- AddLeftDualOnMorphismsWithGivenLeft-Duals
 - for IsCapCategory, IsFunction, 80
 - for IsCapCategory, IsFunction, IsInt, 80
- AddLeftDualOnObjects
 - for IsCapCategory, IsFunction, 80
 - for IsCapCategory, IsFunction, IsInt, 80
- AddLeftInternalCoHomOnMorphisms
 - for IsCapCategory, IsFunction, 91
 - for IsCapCategory, IsFunction, IsInt, 91
- AddLeftInternalCoHomOnMorphismsWith-GivenLeftInternalCoHoms
 - for IsCapCategory, IsFunction, 92
 - for IsCapCategory, IsFunction, IsInt, 92
- AddLeftInternalCoHomOnObjects
 - for IsCapCategory, IsFunction, 92
 - for IsCapCategory, IsFunction, IsInt, 92
- AddLeftInternalCoHomTensorProduct-CompatibilityMorphism
 - for IsCapCategory, IsFunction, 92
 - for IsCapCategory, IsFunction, IsInt, 92
- AddLeftInternalCoHomTensorProduct-CompatibilityMorphismWithGiven-Objects
 - for IsCapCategory, IsFunction, 92
 - for IsCapCategory, IsFunction, IsInt, 92
- AddLeftInternalCoHomToTensorProduct-AdjunctMorphism
 - for IsCapCategory, IsFunction, 93
 - for IsCapCategory, IsFunction, IsInt, 93
- AddLeftInternalCoHomToTensorProduct-AdjunctMorphismWithGivenTensor-Product
 - for IsCapCategory, IsFunction, 93
 - for IsCapCategory, IsFunction, IsInt, 93

- for IsCapCategory, IsFunction, 93
- for IsCapCategory, IsFunction, IsInt, 93
- AddLeftInternalHomOnMorphisms
 - for IsCapCategory, IsFunction, 81
 - for IsCapCategory, IsFunction, IsInt, 81
- AddLeftInternalHomOnMorphismsWith-
 - GivenLeftInternalHoms
 - for IsCapCategory, IsFunction, 81
 - for IsCapCategory, IsFunction, IsInt, 81
- AddLeftInternalHomOnObjects
 - for IsCapCategory, IsFunction, 81
 - for IsCapCategory, IsFunction, IsInt, 81
- AddLeftInternalHomToTensorProduct-
 - AdjunctMorphism
 - for IsCapCategory, IsFunction, 81
 - for IsCapCategory, IsFunction, IsInt, 81
- AddLeftInternalHomToTensorProduct-
 - AdjunctMorphismWithGivenTensor-
Product
 - for IsCapCategory, IsFunction, 82
 - for IsCapCategory, IsFunction, IsInt, 82
- AddLeftUnitor
 - for IsCapCategory, IsFunction, 96
 - for IsCapCategory, IsFunction, IsInt, 96
- AddLeftUnitorInverse
 - for IsCapCategory, IsFunction, 96
 - for IsCapCategory, IsFunction, IsInt, 96
- AddLeftUnitorInverseWithGivenTensor-
 - Product
 - for IsCapCategory, IsFunction, 96
 - for IsCapCategory, IsFunction, IsInt, 96
- AddLeftUnitorWithGivenTensorProduct
 - for IsCapCategory, IsFunction, 97
 - for IsCapCategory, IsFunction, IsInt, 97
- AddMonoidalPostCoComposeMorphism
 - for IsCapCategory, IsFunction, 72
 - for IsCapCategory, IsFunction, IsInt, 72
- AddMonoidalPostCoComposeMorphismWith-
 - GivenObjects
 - for IsCapCategory, IsFunction, 72
 - for IsCapCategory, IsFunction, IsInt, 72
- AddMonoidalPostComposeMorphism
 - for IsCapCategory, IsFunction, 58
 - for IsCapCategory, IsFunction, IsInt, 58
- AddMonoidalPostComposeMorphismWith-
 - GivenObjects
 - for IsCapCategory, IsFunction, 58
 - for IsCapCategory, IsFunction, IsInt, 58
- AddMonoidalPreCoComposeMorphism
 - for IsCapCategory, IsFunction, 72
 - for IsCapCategory, IsFunction, IsInt, 72
- AddMonoidalPreCoComposeMorphismWith-
 - GivenObjects
 - for IsCapCategory, IsFunction, 72
 - for IsCapCategory, IsFunction, IsInt, 72
- AddMonoidalPreComposeMorphism
 - for IsCapCategory, IsFunction, 58
 - for IsCapCategory, IsFunction, IsInt, 58
- AddMonoidalPreComposeMorphismWith-
 - GivenObjects
 - for IsCapCategory, IsFunction, 58
 - for IsCapCategory, IsFunction, IsInt, 58
- AddMorphismFromBidual
 - for IsCapCategory, IsFunction, 101
 - for IsCapCategory, IsFunction, IsInt, 101
- AddMorphismFromBidualWithGivenBidual
 - for IsCapCategory, IsFunction, 101
 - for IsCapCategory, IsFunction, IsInt, 101
- AddMorphismFromCoBidual
 - for IsCapCategory, IsFunction, 73
 - for IsCapCategory, IsFunction, IsInt, 73
- AddMorphismFromCoBidualWithGivenCo-
 - Bidual
 - for IsCapCategory, IsFunction, 73
 - for IsCapCategory, IsFunction, IsInt, 73
- AddMorphismFromInternalCoHomToTensor-
 - Product
 - for IsCapCategory, IsFunction, 73
 - for IsCapCategory, IsFunction, IsInt, 73
- AddMorphismFromInternalCoHomToTensor-
 - ProductWithGivenObjects
 - for IsCapCategory, IsFunction, 73
 - for IsCapCategory, IsFunction, IsInt, 73
- AddMorphismFromInternalHomToTensor-
 - Product
 - for IsCapCategory, IsFunction, 101
 - for IsCapCategory, IsFunction, IsInt, 101
- AddMorphismFromInternalHomToTensor-
 - ProductWithGivenObjects
 - for IsCapCategory, IsFunction, 102
 - for IsCapCategory, IsFunction, IsInt, 102
- AddMorphismFromLeftCoBidual

- for IsCapCategory, IsFunction, 93
- for IsCapCategory, IsFunction, IsInt, 93
- AddMorphismFromLeftCoBidualWithGiven-
LeftCoBidual
 - for IsCapCategory, IsFunction, 93
 - for IsCapCategory, IsFunction, IsInt, 93
- AddMorphismFromLeftInternalCoHomTo-
TensorProduct
 - for IsCapCategory, IsFunction, 94
 - for IsCapCategory, IsFunction, IsInt, 94
- AddMorphismFromLeftInternalCoHomTo-
TensorProductWithGivenObjects
 - for IsCapCategory, IsFunction, 94
 - for IsCapCategory, IsFunction, IsInt, 94
- AddMorphismFromTensorProductTo-
InternalCoHom
 - for IsCapCategory, IsFunction, 105
 - for IsCapCategory, IsFunction, IsInt, 105
- AddMorphismFromTensorProductTo-
InternalCoHomWithGivenObjects
 - for IsCapCategory, IsFunction, 105
 - for IsCapCategory, IsFunction, IsInt, 105
- AddMorphismFromTensorProductTo-
InternalHom
 - for IsCapCategory, IsFunction, 59
 - for IsCapCategory, IsFunction, IsInt, 59
- AddMorphismFromTensorProductTo-
InternalHomWithGivenObjects
 - for IsCapCategory, IsFunction, 59
 - for IsCapCategory, IsFunction, IsInt, 59
- AddMorphismFromTensorProductToLeft-
InternalHom
 - for IsCapCategory, IsFunction, 82
 - for IsCapCategory, IsFunction, IsInt, 82
- AddMorphismFromTensorProductToLeft-
InternalHomWithGivenObjects
 - for IsCapCategory, IsFunction, 82
 - for IsCapCategory, IsFunction, IsInt, 82
- AddMorphismToBidual
 - for IsCapCategory, IsFunction, 59
 - for IsCapCategory, IsFunction, IsInt, 59
- AddMorphismToBidualWithGivenBidual
 - for IsCapCategory, IsFunction, 59
 - for IsCapCategory, IsFunction, IsInt, 59
- AddMorphismToCoBidual
 - for IsCapCategory, IsFunction, 106
- for IsCapCategory, IsFunction, IsInt, 106
- AddMorphismToCoBidualWithGivenCoBidual
 - for IsCapCategory, IsFunction, 106
 - for IsCapCategory, IsFunction, IsInt, 106
- AddMorphismToLeftBidual
 - for IsCapCategory, IsFunction, 83
 - for IsCapCategory, IsFunction, IsInt, 83
- AddMorphismToLeftBidualWithGivenLeft-
Bidual
 - for IsCapCategory, IsFunction, 83
 - for IsCapCategory, IsFunction, IsInt, 83
- AddRankMorphism
 - for IsCapCategory, IsFunction, 102
 - for IsCapCategory, IsFunction, IsInt, 102
- AddRightDistributivityExpanding
 - for IsCapCategory, IsFunction, 47
 - for IsCapCategory, IsFunction, IsInt, 47
- AddRightDistributivityExpandingWith-
GivenObjects
 - for IsCapCategory, IsFunction, 48
 - for IsCapCategory, IsFunction, IsInt, 48
- AddRightDistributivityFactoring
 - for IsCapCategory, IsFunction, 48
 - for IsCapCategory, IsFunction, IsInt, 48
- AddRightDistributivityFactoringWith-
GivenObjects
 - for IsCapCategory, IsFunction, 48
 - for IsCapCategory, IsFunction, IsInt, 48
- AddRightUnitor
 - for IsCapCategory, IsFunction, 97
 - for IsCapCategory, IsFunction, IsInt, 97
- AddRightUnitorInverse
 - for IsCapCategory, IsFunction, 97
 - for IsCapCategory, IsFunction, IsInt, 97
- AddRightUnitorInverseWithGivenTensor-
Product
 - for IsCapCategory, IsFunction, 97
 - for IsCapCategory, IsFunction, IsInt, 97
- AddRightUnitorWithGivenTensorProduct
 - for IsCapCategory, IsFunction, 98
 - for IsCapCategory, IsFunction, IsInt, 98
- AddTensorProductDualityCompatibility-
Morphism
 - for IsCapCategory, IsFunction, 60
 - for IsCapCategory, IsFunction, IsInt, 60
- AddTensorProductDualityCompatibility-

- MorphismWithGivenObjects
 - for IsCapCategory, IsFunction, 60
 - for IsCapCategory, IsFunction, IsInt, 60
- AddTensorProductInternalHom-CompatibilityMorphism
 - for IsCapCategory, IsFunction, 60
 - for IsCapCategory, IsFunction, IsInt, 60
- AddTensorProductInternalHom-CompatibilityMorphismInverse
 - for IsCapCategory, IsFunction, 102
 - for IsCapCategory, IsFunction, IsInt, 102
- AddTensorProductInternalHom-CompatibilityMorphismInverse-WithGivenObjects
 - for IsCapCategory, IsFunction, 102
 - for IsCapCategory, IsFunction, IsInt, 102
- AddTensorProductInternalHom-CompatibilityMorphismWith-GivenObjects
 - for IsCapCategory, IsFunction, 60
 - for IsCapCategory, IsFunction, IsInt, 60
- AddTensorProductLeftDuality-CompatibilityMorphism
 - for IsCapCategory, IsFunction, 83
 - for IsCapCategory, IsFunction, IsInt, 83
- AddTensorProductLeftDuality-CompatibilityMorphismWith-GivenObjects
 - for IsCapCategory, IsFunction, 83
 - for IsCapCategory, IsFunction, IsInt, 83
- AddTensorProductLeftInternalHom-CompatibilityMorphism
 - for IsCapCategory, IsFunction, 84
 - for IsCapCategory, IsFunction, IsInt, 84
- AddTensorProductLeftInternalHom-CompatibilityMorphismWith-GivenObjects
 - for IsCapCategory, IsFunction, 84
 - for IsCapCategory, IsFunction, IsInt, 84
- AddTensorProductOnMorphismAndObject
 - for IsCapCategory, IsFunction, 98
 - for IsCapCategory, IsFunction, IsInt, 98
- AddTensorProductOnMorphismAndObject-WithGivenTensorProducts
 - for IsCapCategory, IsFunction, 98
 - for IsCapCategory, IsFunction, IsInt, 98
- AddTensorProductOnMorphisms
 - for IsCapCategory, IsFunction, 98
 - for IsCapCategory, IsFunction, IsInt, 98
- AddTensorProductOnMorphismsWithGiven-TensorProducts
 - for IsCapCategory, IsFunction, 99
 - for IsCapCategory, IsFunction, IsInt, 99
- AddTensorProductOnObjectAndMorphism
 - for IsCapCategory, IsFunction, 99
 - for IsCapCategory, IsFunction, IsInt, 99
- AddTensorProductOnObjectAndMorphism-WithGivenTensorProducts
 - for IsCapCategory, IsFunction, 99
 - for IsCapCategory, IsFunction, IsInt, 99
- AddTensorProductOnObjects
 - for IsCapCategory, IsFunction, 99
 - for IsCapCategory, IsFunction, IsInt, 99
- AddTensorProductToInternalCoHomLeft-AdjunctMorphism
 - for IsCapCategory, IsFunction, 74
 - for IsCapCategory, IsFunction, IsInt, 74
- AddTensorProductToInternalCoHomLeft-AdjunctMorphismWithGiven-InternalCoHom
 - for IsCapCategory, IsFunction, 74
 - for IsCapCategory, IsFunction, IsInt, 74
- AddTensorProductToInternalCoHomRight-AdjunctMorphism
 - for IsCapCategory, IsFunction, 74
 - for IsCapCategory, IsFunction, IsInt, 74
- AddTensorProductToInternalCoHomRight-AdjunctMorphismWithGiven-InternalCoHom
 - for IsCapCategory, IsFunction, 75
 - for IsCapCategory, IsFunction, IsInt, 75
- AddTensorProductToInternalHomLeft-AdjunctionIsomorphism
 - for IsCapCategory, IsFunction, 61
 - for IsCapCategory, IsFunction, IsInt, 61
- AddTensorProductToInternalHomLeft-AdjunctionIsomorphismWithGiven-Objects
 - for IsCapCategory, IsFunction, 62
 - for IsCapCategory, IsFunction, IsInt, 62
- AddTensorProductToInternalHomLeft-AdjunctMorphism

- for IsCapCategory, IsFunction, 61
- for IsCapCategory, IsFunction, IsInt, 61
- AddTensorProductToInternalHomLeft-
 - AdjunctMorphismWithGiven-InternalHom
 - for IsCapCategory, IsFunction, 61
 - for IsCapCategory, IsFunction, IsInt, 61
- AddTensorProductToInternalHomRight-
 - AdjunctionIsomorphism
 - for IsCapCategory, IsFunction, 62
 - for IsCapCategory, IsFunction, IsInt, 62
- AddTensorProductToInternalHomRight-
 - AdjunctionIsomorphismWithGiven-Objects
 - for IsCapCategory, IsFunction, 63
 - for IsCapCategory, IsFunction, IsInt, 63
- AddTensorProductToInternalHomRight-
 - AdjunctMorphism
 - for IsCapCategory, IsFunction, 62
 - for IsCapCategory, IsFunction, IsInt, 62
- AddTensorProductToInternalHomRight-
 - AdjunctMorphismWithGiven-InternalHom
 - for IsCapCategory, IsFunction, 62
 - for IsCapCategory, IsFunction, IsInt, 62
- AddTensorProductToLeftInternalCoHom-
 - AdjunctMorphism
 - for IsCapCategory, IsFunction, 94
 - for IsCapCategory, IsFunction, IsInt, 94
- AddTensorProductToLeftInternalCoHom-
 - AdjunctMorphismWithGivenLeft-InternalCoHom
 - for IsCapCategory, IsFunction, 95
 - for IsCapCategory, IsFunction, IsInt, 95
- AddTensorProductToLeftInternalHom-
 - AdjunctMorphism
 - for IsCapCategory, IsFunction, 84
 - for IsCapCategory, IsFunction, IsInt, 84
- AddTensorProductToLeftInternalHom-
 - AdjunctMorphismWithGivenLeft-InternalHom
 - for IsCapCategory, IsFunction, 84
 - for IsCapCategory, IsFunction, IsInt, 84
- AddTensorUnit
 - for IsCapCategory, IsFunction, 100
 - for IsCapCategory, IsFunction, IsInt, 100
- AddTraceMap
 - for IsCapCategory, IsFunction, 103
 - for IsCapCategory, IsFunction, IsInt, 103
- AddUniversalPropertyOfCoDual
 - for IsCapCategory, IsFunction, 75
 - for IsCapCategory, IsFunction, IsInt, 75
- AddUniversalPropertyOfDual
 - for IsCapCategory, IsFunction, 63
 - for IsCapCategory, IsFunction, IsInt, 63
- AddUniversalPropertyOfLeftCoDual
 - for IsCapCategory, IsFunction, 95
 - for IsCapCategory, IsFunction, IsInt, 95
- AddUniversalPropertyOfLeftDual
 - for IsCapCategory, IsFunction, 85
 - for IsCapCategory, IsFunction, IsInt, 85
- AssociatorLeftToRight
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 5
- AssociatorLeftToRightWithGivenTensor-Products
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, IsCapCategoryObject, 5
- AssociatorRightToLeft
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 5
- AssociatorRightToLeftWithGivenTensor-Products
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, IsCapCategoryObject, 5
- BraidedMonoidalCategoriesTest, 108
- Braiding
 - for IsCapCategoryObject, IsCapCategory-Object, 8
- BraidingInverse
 - for IsCapCategoryObject, IsCapCategory-Object, 9
- BraidingInverseWithGivenTensorProducts
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, 9
- BraidingWithGivenTensorProducts

- for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, 8
- ClosedMonoidalCategoriesTest, 108
- ClosedMonoidalCategoriesTestWithGiven, 109
- ClosedMonoidalLeftCoevaluationMorphism
 - for IsCapCategoryObject, IsCapCategory-Object, 19
- ClosedMonoidalLeftCoevaluation-
MorphismWithGivenRange
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 19
- ClosedMonoidalLeftEvaluationMorphism
 - for IsCapCategoryObject, IsCapCategory-Object, 19
- ClosedMonoidalLeftEvaluationMorphism-
WithGivenSource
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 19
- ClosedMonoidalRightCoevaluation-
Morphism
 - for IsCapCategoryObject, IsCapCategory-Object, 17
- ClosedMonoidalRightCoevaluation-
MorphismWithGivenRange
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 17
- ClosedMonoidalRightEvaluationMorphism
 - for IsCapCategoryObject, IsCapCategory-Object, 16
- ClosedMonoidalRightEvaluationMorphism-
WithGivenSource
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 16
- CoclosedCoevaluationForCoDual
 - for IsCapCategoryObject, 45
- CoclosedCoevaluationForCoDualWith-
GivenTensorProduct
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 45
- CoclosedCoevaluationMorphism
 - for IsCapCategoryObject, IsCapCategory-Object, 120
- CoclosedCoevaluationMorphismWithGiven-
Source
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 120
- CoclosedEvaluationForCoDual
 - for IsCapCategoryObject, 37
- CoclosedEvaluationForCoDualWithGiven-
TensorProduct
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 37
- CoclosedEvaluationMorphism
 - for IsCapCategoryObject, IsCapCategory-Object, 120
- CoclosedEvaluationMorphismWithGiven-
Range
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 120
- CoclosedMonoidalCategoriesTest, 110
- CoclosedMonoidalCategoriesTestWith-
Given, 110
- CoclosedMonoidalLeftCoevaluation-
Morphism
 - for IsCapCategoryObject, IsCapCategory-Object, 35
- CoclosedMonoidalLeftCoevaluation-
MorphismWithGivenSource
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 35
- CoclosedMonoidalLeftEvaluationMorphism
 - for IsCapCategoryObject, IsCapCategory-Object, 34
- CoclosedMonoidalLeftEvaluation-
MorphismWithGivenRange
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 35
- CoclosedMonoidalRightCoevaluation-
Morphism
 - for IsCapCategoryObject, IsCapCategory-Object, 33
- CoclosedMonoidalRightCoevaluation-
MorphismWithGivenSource
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, 33
- CoclosedMonoidalRightEvaluation-
Morphism
 - for IsCapCategoryObject, IsCapCategory-Object, 33
- CoclosedMonoidalRightEvaluation-

MorphismWithGivenRange
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 33
 CoDualityTensorProductCompatibility-
 Morphism
 for IsCapCategoryObject, IsCapCategory-
 Object, 38
 CoDualityTensorProductCompatibility-
 MorphismWithGivenObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, 38
 CoDualOnMorphisms
 for IsCapCategoryMorphism, 37
 CoDualOnMorphismsWithGivenCoDUALS
 for IsCapCategoryObject, IsCapCategory-
 Morphism, IsCapCategoryObject, 37
 CoDualOnObjects
 for IsCapCategoryObject, 37
 CoevaluationForDual
 for IsCapCategoryObject, 42
 CoevaluationForDualWithGivenTensor-
 Product
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 42
 CoevaluationMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, 121
 CoevaluationMorphismWithGivenRange
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 121
 CoLambdaElimination
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 40
 CoLambdaIntroduction
 for IsCapCategoryMorphism, 40
 CoRankMorphism
 for IsCapCategoryObject, 45
 CoTraceMap
 for IsCapCategoryMorphism, 45
 DualOnMorphisms
 for IsCapCategoryMorphism, 22
 DualOnMorphismsWithGivenDuals
 for IsCapCategoryObject, IsCapCategory-
 Morphism, IsCapCategoryObject, 22
 DualOnObjects
 for IsCapCategoryObject, 22
 EvaluationForDual
 for IsCapCategoryObject, 22
 EvaluationForDualWithGivenTensor-
 Product
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 22
 EvaluationMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, 121
 EvaluationMorphismWithGivenSource
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 121
 InternalCoHom
 for IsCapCategoryCell, IsCapCategoryCell,
 46
 InternalCoHomOnMorphisms
 for IsCapCategoryMorphism, IsCapCatego-
 ryMorphism, 32
 InternalCoHomOnMorphismsWithGiven-
 InternalCoHoms
 for IsCapCategoryObject, IsCapCategory-
 Morphism, IsCapCategoryMorphism,
 IsCapCategoryObject, 33
 InternalCoHomOnObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, 32
 InternalCoHomTensorProduct-
 CompatibilityMorphism
 for IsList, 38
 InternalCoHomTensorProduct-
 CompatibilityMorphismInverse
 for IsList, 44
 InternalCoHomTensorProduct-
 CompatibilityMorphismInverse-
 WithGivenObjects
 for IsCapCategoryObject, IsList, IsCapCate-
 goryObject, 44
 InternalCoHomTensorProduct-
 CompatibilityMorphismWith-
 GivenObjects
 for IsCapCategoryObject, IsList, IsCapCate-
 goryObject, 38
 InternalCoHomToTensorProduct-
 AdjunctionMap

for IsObject, 121
 InternalCoHomToTensorProduct-
 AdjunctionMapWithGivenTensor-
 Product
 for IsObject, 121
 InternalCoHomToTensorProductLeft-
 AdjunctionMap
 for IsObject, 122
 InternalCoHomToTensorProductLeft-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 36
 InternalCoHomToTensorProductLeft-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, 36
 InternalCoHomToTensorProductRight-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 34
 InternalCoHomToTensorProductRight-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, 34
 InternalHom
 for IsCapCategoryCell, IsCapCategoryCell,
 46
 InternalHomOnMorphisms
 for IsCapCategoryMorphism, IsCapCatego-
 ryMorphism, 16
 InternalHomOnMorphismsWithGiven-
 InternalHoms
 for IsCapCategoryObject, IsCapCategory-
 Morphism, IsCapCategoryMorphism,
 IsCapCategoryObject, 16
 InternalHomOnObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, 16
 InternalHomToTensorProduct-
 AdjunctionMap
 for IsObject, 122
 InternalHomToTensorProductAdjunction-
 MapWithGivenTensorProduct
 for IsObject, 122
 InternalHomToTensorProductLeft-
 AdjunctionIsomorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 20
 InternalHomToTensorProductLeft-
 AdjunctionIsomorphismWith-
 GivenObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject,
 21
 InternalHomToTensorProductLeft-
 AdjunctionMap
 for IsObject, 123
 InternalHomToTensorProductLeftAdjunct-
 Morphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 20
 InternalHomToTensorProductLeftAdjunct-
 MorphismWithGivenTensorProduct
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, 20
 InternalHomToTensorProductRight-
 AdjunctionIsomorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 18
 InternalHomToTensorProductRight-
 AdjunctionIsomorphismWith-
 GivenObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject,
 18
 InternalHomToTensorProductRight-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 18
 InternalHomToTensorProductRight-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, 18

IsAdditiveMonoidalCategory
 for IsCapCategory, [124](#)
 IsBraidedMonoidalCategory
 for IsCapCategory, [124](#)
 IsClosedMonoidalCategory
 for IsCapCategory, [124](#)
 IsCoclosedMonoidalCategory
 for IsCapCategory, [124](#)
 IsLeftClosedMonoidalCategory
 for IsCapCategory, [124](#)
 IsLeftCoclosedMonoidalCategory
 for IsCapCategory, [125](#)
 IsMonoidalCategory
 for IsCapCategory, [125](#)
 IsomorphismFromCoDualObjectToInternal-
 CoHomFromTensorUnit
 for IsCapCategoryObject, [39](#)
 IsomorphismFromDualObjectToInternal-
 HomIntoTensorUnit
 for IsCapCategoryObject, [24](#)
 IsomorphismFromInternalCoHomFrom-
 TensorUnitToCoDualObject
 for IsCapCategoryObject, [39](#)
 IsomorphismFromInternalCoHomToObject
 for IsCapCategoryObject, [40](#)
 IsomorphismFromInternalCoHomToObject-
 WithGivenInternalCoHom
 for IsCapCategoryObject, IsCapCategory-
 Object, [40](#)
 IsomorphismFromInternalCoHomToTensor-
 ProductWithCoDualObject
 for IsCapCategoryObject, IsCapCategory-
 Object, [43](#)
 IsomorphismFromInternalHomIntoTensor-
 UnitToDualObject
 for IsCapCategoryObject, [24](#)
 IsomorphismFromInternalHomToObject
 for IsCapCategoryObject, [25](#)
 IsomorphismFromInternalHomToObject-
 WithGivenInternalHom
 for IsCapCategoryObject, IsCapCategory-
 Object, [25](#)
 IsomorphismFromInternalHomToTensor-
 ProductWithDualObject
 for IsCapCategoryObject, IsCapCategory-
 Object, [41](#)
 IsomorphismFromLeftCoDualObjectToLeft-
 InternalCoHomFromTensorUnit
 for IsCapCategoryObject, [31](#)
 IsomorphismFromLeftDualObjectToLeft-
 InternalHomIntoTensorUnit
 for IsCapCategoryObject, [14](#)
 IsomorphismFromLeftInternalCoHomFrom-
 TensorUnitToLeftCoDualObject
 for IsCapCategoryObject, [31](#)
 IsomorphismFromLeftInternalCoHomTo-
 Object
 for IsCapCategoryObject, [32](#)
 IsomorphismFromLeftInternalCoHomTo-
 ObjectWithGivenLeftInternal-
 CoHom
 for IsCapCategoryObject, IsCapCategory-
 Object, [32](#)
 IsomorphismFromLeftInternalHomInto-
 TensorUnitToLeftDualObject
 for IsCapCategoryObject, [14](#)
 IsomorphismFromLeftInternalHomToObject
 for IsCapCategoryObject, [15](#)
 IsomorphismFromLeftInternalHomTo-
 ObjectWithGivenLeftInternalHom
 for IsCapCategoryObject, IsCapCategory-
 Object, [15](#)
 IsomorphismFromObjectToInternalCoHom
 for IsCapCategoryObject, [40](#)
 IsomorphismFromObjectToInternalCoHom-
 WithGivenInternalCoHom
 for IsCapCategoryObject, IsCapCategory-
 Object, [40](#)
 IsomorphismFromObjectToInternalHom
 for IsCapCategoryObject, [25](#)
 IsomorphismFromObjectToInternalHom-
 WithGivenInternalHom
 for IsCapCategoryObject, IsCapCategory-
 Object, [25](#)
 IsomorphismFromObjectToLeftInternal-
 CoHom
 for IsCapCategoryObject, [31](#)
 IsomorphismFromObjectToLeftInternalCo-
 HomWithGivenLeftInternalCoHom
 for IsCapCategoryObject, IsCapCategory-
 Object, [32](#)
 IsomorphismFromObjectToLeftInternalHom

- for IsCapCategoryObject, [15](#)
- IsomorphismFromObjectToLeftInternalHomWithGivenLeftInternalHom
 - for IsCapCategoryObject, IsCapCategoryObject, [15](#)
- IsomorphismFromTensorProductWithCoDualObjectToInternalCoHom
 - for IsCapCategoryObject, IsCapCategoryObject, [44](#)
- IsomorphismFromTensorProductWithDualObjectToInternalHom
 - for IsCapCategoryObject, IsCapCategoryObject, [41](#)
- IsRigidSymmetricClosedMonoidalCategory
 - for IsCapCategory, [125](#)
- IsRigidSymmetricCoclosedMonoidalCategory
 - for IsCapCategory, [125](#)
- IsStrictMonoidalCategory
 - for IsCapCategory, [125](#)
- IsSymmetricClosedMonoidalCategory
 - for IsCapCategory, [125](#)
- IsSymmetricCoclosedMonoidalCategory
 - for IsCapCategory, [125](#)
- IsSymmetricMonoidalCategory
 - for IsCapCategory, [125](#)
- LambdaElimination
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, [24](#)
- LambdaIntroduction
 - for IsCapCategoryMorphism, [24](#)
- LeftClosedMonoidalCategoriesTest, [111](#)
- LeftClosedMonoidalCategoriesTestWithGiven, [111](#)
- LeftClosedMonoidalCoevaluationMorphism
 - for IsCapCategoryObject, IsCapCategoryObject, [10](#)
- LeftClosedMonoidalCoevaluationMorphismWithGivenRange
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [10](#)
- LeftClosedMonoidalEvaluationForLeftDual
 - for IsCapCategoryObject, [12](#)
- LeftClosedMonoidalEvaluationForLeftDualWithGivenTensorProduct
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [13](#)
- LeftClosedMonoidalEvaluationMorphism
 - for IsCapCategoryObject, IsCapCategoryObject, [10](#)
- LeftClosedMonoidalEvaluationMorphismWithGivenSource
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [10](#)
- LeftClosedMonoidalLambdaElimination
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, [15](#)
- LeftClosedMonoidalLambdaIntroduction
 - for IsCapCategoryMorphism, [15](#)
- LeftClosedMonoidalPostComposeMorphism
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [12](#)
- LeftClosedMonoidalPostComposeMorphismWithGivenObjects
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [12](#)
- LeftClosedMonoidalPreComposeMorphism
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [11](#)
- LeftClosedMonoidalPreComposeMorphismWithGivenObjects
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [11](#)
- LeftCoclosedMonoidalCategoriesTest, [112](#)
- LeftCoclosedMonoidalCategoriesTestWithGiven, [113](#)
- LeftCoclosedMonoidalCoevaluationMorphism
 - for IsCapCategoryObject, IsCapCategoryObject, [26](#)
- LeftCoclosedMonoidalCoevaluationMorphismWithGivenSource
 - for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [26](#)
- LeftCoclosedMonoidalEvaluationForLeftCoDual
 - for IsCapCategoryObject, [29](#)

LeftCoclosedMonoidalEvaluationForLeftCoDualWithGivenTensorProduct
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [29](#)
 LeftCoclosedMonoidalEvaluationMorphism
 for IsCapCategoryObject, IsCapCategoryObject, [26](#)
 LeftCoclosedMonoidalEvaluationMorphismWithGivenRange
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [26](#)
 LeftCoclosedMonoidalLambdaElimination
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, [31](#)
 LeftCoclosedMonoidalLambdaIntroduction
 for IsCapCategoryMorphism, [31](#)
 LeftCoclosedMonoidalPostCoComposeMorphism
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [28](#)
 LeftCoclosedMonoidalPostCoComposeMorphismWithGivenObjects
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [28](#)
 LeftCoclosedMonoidalPreCoComposeMorphism
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [27](#)
 LeftCoclosedMonoidalPreCoComposeMorphismWithGivenObjects
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [28](#)
 LeftCoDualityTensorProductCompatibilityMorphism
 for IsCapCategoryObject, IsCapCategoryObject, [30](#)
 LeftCoDualityTensorProductCompatibilityMorphismWithGivenObjects
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, [30](#)
 LeftCoDualOnMorphisms
 for IsCapCategoryMorphism, [28](#)
 LeftCoDualOnMorphismsWithGivenLeftCoDUALS
 for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject, [29](#)
 LeftCoDualOnObjects
 for IsCapCategoryObject, [28](#)
 LeftDistributivityExpanding
 for IsCapCategoryObject, IsList, [7](#)
 LeftDistributivityExpandingWithGivenObjects
 for IsCapCategoryObject, IsCapCategoryObject, IsList, IsCapCategoryObject, [7](#)
 LeftDistributivityFactoring
 for IsCapCategoryObject, IsList, [7](#)
 LeftDistributivityFactoringWithGivenObjects
 for IsCapCategoryObject, IsCapCategoryObject, IsList, IsCapCategoryObject, [7](#)
 LeftDualOnMorphisms
 for IsCapCategoryMorphism, [12](#)
 LeftDualOnMorphismsWithGivenLeftDualS
 for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject, [12](#)
 LeftDualOnObjects
 for IsCapCategoryObject, [12](#)
 LeftInternalCoHom
 for IsCapCategoryCell, IsCapCategoryCell, [46](#)
 LeftInternalCoHomOnMorphisms
 for IsCapCategoryMorphism, IsCapCategoryMorphism, [25](#)
 LeftInternalCoHomOnMorphismsWithGivenLeftInternalCoHOMs
 for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryMorphism, IsCapCategoryObject, [26](#)
 LeftInternalCoHomOnObjects
 for IsCapCategoryObject, IsCapCategoryObject, [25](#)
 LeftInternalCoHomTensorProductCompatibilityMorphism
 for IsList, [29](#)
 LeftInternalCoHomTensorProductCompatibilityMorphismWith-

GivenObjects
 for IsCapCategoryObject, IsList, IsCapCategoryObject, 30
 LeftInternalCoHomToTensorProduct-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 27
 LeftInternalCoHomToTensorProduct-
 AdjunctMorphismWithGivenTensor-
 Product
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, 27
 LeftInternalHom
 for IsCapCategoryCell, IsCapCategoryCell, 46
 LeftInternalHomOnMorphisms
 for IsCapCategoryMorphism, IsCapCategory-
 Morphism, 9
 LeftInternalHomOnMorphismsWithGiven-
 LeftInternalHoms
 for IsCapCategoryObject, IsCapCategory-
 Morphism, IsCapCategoryMorphism, IsCapCategoryObject, 9
 LeftInternalHomOnObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, 9
 LeftInternalHomToTensorProductAdjunct-
 Morphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 11
 LeftInternalHomToTensorProductAdjunct-
 MorphismWithGivenTensorProduct
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, 11
 LeftUnitor
 for IsCapCategoryObject, 5
 LeftUnitorInverse
 for IsCapCategoryObject, 6
 LeftUnitorInverseWithGivenTensor-
 Product
 for IsCapCategoryObject, IsCapCategory-
 Object, 6
 LeftUnitorWithGivenTensorProduct
 for IsCapCategoryObject, IsCapCategory-
 Object, 5
 MonoidalCategoriesTensorProductOn-
 ObjectsAndTensorUnitTest, 113
 MonoidalCategoriesTest, 115
 MonoidalPostCoComposeMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 36
 MonoidalPostCoComposeMorphismWith-
 GivenObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject, 37
 MonoidalPostComposeMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 21
 MonoidalPostComposeMorphismWithGiven-
 Objects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject, 21
 MonoidalPreCoComposeMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 36
 MonoidalPreCoComposeMorphismWithGiven-
 Objects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject, 36
 MonoidalPreComposeMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, 21
 MonoidalPreComposeMorphismWithGiven-
 Objects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject, 21
 MorphismFromBidual
 for IsCapCategoryObject, 43
 MorphismFromBidualWithGivenBidual
 for IsCapCategoryObject, IsCapCategory-
 Object, 43
 MorphismFromCoBidual

- for IsCapCategoryObject, [38](#)
- MorphismFromCoBidualWithGivenCoBidual
 - for IsCapCategoryObject, IsCapCategory-Object, [38](#)
- MorphismFromInternalCoHomToTensor-Product
 - for IsCapCategoryObject, IsCapCategory-Object, [39](#)
- MorphismFromInternalCoHomToTensor-ProductWithGivenObjects
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, [39](#)
- MorphismFromInternalHomToTensorProduct
 - for IsCapCategoryObject, IsCapCategory-Object, [41](#)
- MorphismFromInternalHomToTensor-ProductWithGivenObjects
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, [42](#)
- MorphismFromLeftCoBidual
 - for IsCapCategoryObject, [29](#)
- MorphismFromLeftCoBidualWithGivenLeft-CoBidual
 - for IsCapCategoryObject, IsCapCategory-Object, [29](#)
- MorphismFromLeftInternalCoHomToTensor-Product
 - for IsCapCategoryObject, IsCapCategory-Object, [30](#)
- MorphismFromLeftInternalCoHomToTensor-ProductWithGivenObjects
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, [30](#)
- MorphismFromTensorProductToInternal-CoHom
 - for IsCapCategoryObject, IsCapCategory-Object, [44](#)
- MorphismFromTensorProductToInternalCo-HomWithGivenObjects
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, [44](#)
- MorphismFromTensorProductToInternalHom
 - for IsCapCategoryObject, IsCapCategory-Object, [23](#)
- MorphismFromTensorProductToInternal-HomWithGivenObjects
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, [24](#)
- MorphismFromTensorProductToLeft-InternalHom
 - for IsCapCategoryObject, IsCapCategory-Object, [14](#)
- MorphismFromTensorProductToLeft-InternalHomWithGivenObjects
 - for IsCapCategoryObject, IsCapCategory-Object, IsCapCategoryObject, IsCap-CategoryObject, [14](#)
- MorphismToBidual
 - for IsCapCategoryObject, [22](#)
- MorphismToBidualWithGivenBidual
 - for IsCapCategoryObject, IsCapCategory-Object, [22](#)
- MorphismToCoBidual
 - for IsCapCategoryObject, [45](#)
- MorphismToCoBidualWithGivenCoBidual
 - for IsCapCategoryObject, IsCapCategory-Object, [45](#)
- MorphismToLeftBidual
 - for IsCapCategoryObject, [13](#)
- MorphismToLeftBidualWithGivenLeft-Bidual
 - for IsCapCategoryObject, IsCapCategory-Object, [13](#)
- RankMorphism
 - for IsCapCategoryObject, [43](#)
- RightDistributivityExpanding
 - for IsList, IsCapCategoryObject, [7](#)
- RightDistributivityExpandingWithGiven-Objects
 - for IsCapCategoryObject, IsList, IsCapCate-goryObject, IsCapCategoryObject, [8](#)
- RightDistributivityFactoring
 - for IsList, IsCapCategoryObject, [8](#)
- RightDistributivityFactoringWithGiven-Objects
 - for IsCapCategoryObject, IsList, IsCapCate-goryObject, IsCapCategoryObject, [8](#)

RightUnitor
 for IsCapCategoryObject, 6
 RightUnitorInverse
 for IsCapCategoryObject, 6
 RightUnitorInverseWithGivenTensorProduct
 for IsCapCategoryObject, IsCapCategoryObject, 6
 RightUnitorWithGivenTensorProduct
 for IsCapCategoryObject, IsCapCategoryObject, 6
 RigidSymmetricClosedMonoidalCategoriesTest, 115
 RigidSymmetricCoclosedMonoidalCategoriesTest, 116
 TensorProductDualityCompatibilityMorphism
 for IsCapCategoryObject, IsCapCategoryObject, 23
 TensorProductDualityCompatibilityMorphismWithGivenObjects
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, 23
 TensorProductInternalHomCompatibilityMorphism
 for IsList, 23
 TensorProductInternalHomCompatibilityMorphismInverse
 for IsList, 42
 TensorProductInternalHomCompatibilityMorphismInverseWithGivenObjects
 for IsCapCategoryObject, IsList, IsCapCategoryObject, 42
 TensorProductInternalHomCompatibilityMorphismWithGivenObjects
 for IsCapCategoryObject, IsList, IsCapCategoryObject, 23
 TensorProductLeftDualityCompatibilityMorphism
 for IsCapCategoryObject, IsCapCategoryObject, 13
 TensorProductLeftDualityCompatibilityMorphismWithGivenObjects
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryObject, 14
 TensorProductLeftInternalHomCompatibilityMorphism
 for IsList, 13
 TensorProductLeftInternalHomCompatibilityMorphismWithGivenObjects
 for IsCapCategoryObject, IsList, IsCapCategoryObject, 13
 TensorProductOnMorphismAndObject
 for IsCapCategoryMorphism, IsCapCategoryObject, 4
 TensorProductOnMorphismAndObjectWithGivenTensorProducts
 for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject, IsCapCategoryObject, 4
 TensorProductOnMorphisms
 for IsCapCategoryMorphism, IsCapCategoryMorphism, 3
 TensorProductOnMorphismsWithGivenTensorProducts
 for IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryMorphism, IsCapCategoryObject, 4
 TensorProductOnObjectAndMorphism
 for IsCapCategoryObject, IsCapCategoryMorphism, 4
 TensorProductOnObjectAndMorphismWithGivenTensorProducts
 for IsCapCategoryObject, IsCapCategoryObject, IsCapCategoryMorphism, IsCapCategoryObject, 4
 TensorProductOnObjects
 for IsCapCategoryObject, IsCapCategoryObject, 6
 TensorProductToInternalCoHomAdjunctionMap
 for IsObject, 122
 TensorProductToInternalCoHomAdjunctionMapWithGivenInternalCoHom
 for IsObject, 122
 TensorProductToInternalCoHomLeftAdjunctionMap
 for IsObject, 123

TensorProductToInternalCoHomLeft-
 AdjunctionMapWithGivenInternal-
 CoHom
 for IsObject, [123](#)
 TensorProductToInternalCoHomLeft-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, [35](#)
 TensorProductToInternalCoHomLeft-
 AdjunctMorphismWithGiven-
 InternalCoHom
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, [35](#)
 TensorProductToInternalCoHomRight-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, [34](#)
 TensorProductToInternalCoHomRight-
 AdjunctMorphismWithGiven-
 InternalCoHom
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, [34](#)
 TensorProductToInternalHom-
 AdjunctionMap
 for IsObject, [122](#)
 TensorProductToInternalHomAdjunction-
 MapWithGivenInternalHom
 for IsObject, [122](#)
 TensorProductToInternalHomLeft-
 AdjunctionIsomorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, [20](#)
 TensorProductToInternalHomLeft-
 AdjunctionIsomorphismWith-
 GivenObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject,
 [20](#)
 TensorProductToInternalHomLeft-
 AdjunctionMap
 for IsObject, [123](#)
 TensorProductToInternalHomLeft-
 AdjunctionMapWithGiven-
 InternalHom
 for IsObject, [123](#)
 TensorProductToInternalHomLeftAdjunct-
 Morphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, [19](#)
 TensorProductToInternalHomLeftAdjunct-
 MorphismWithGivenInternalHom
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, [19](#)
 TensorProductToInternalHomRight-
 AdjunctionIsomorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, [17](#)
 TensorProductToInternalHomRight-
 AdjunctionIsomorphismWith-
 GivenObjects
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryObject, IsCap-
 CategoryObject, IsCapCategoryObject,
 [18](#)
 TensorProductToInternalHomRight-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, [17](#)
 TensorProductToInternalHomRight-
 AdjunctMorphismWithGiven-
 InternalHom
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, [17](#)
 TensorProductToLeftInternalCoHom-
 AdjunctMorphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, [27](#)
 TensorProductToLeftInternalCoHom-
 AdjunctMorphismWithGivenLeft-
 InternalCoHom
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, [27](#)
 TensorProductToLeftInternalHomAdjunct-
 Morphism
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, [10](#)

TensorProductToLeftInternalHom-
 AdjunctMorphismWithGivenLeft-
 InternalHom
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, Is-
 CapCategoryObject, 11
 TensorUnit
 for IsCapCategory, 7
 TestAssociatorForInvertibility
 for IsCapCategory, IsCapCategoryObject, Is-
 CapCategoryObject, IsCapCategoryOb-
 ject, 114
 TestBraidingCompatibility
 for IsCapCategory, IsCapCategoryObject, Is-
 CapCategoryObject, IsCapCategoryOb-
 ject, 108
 TestBraidingCompatibilityForAll-
 TriplesInList
 for IsCapCategory, IsList, 108
 TestBraidingForInvertibility
 for IsCapCategory, IsCapCategoryObject, Is-
 CapCategoryObject, 107
 TestMonoidalPentagonIdentity
 for IsCapCategory, IsCapCategoryObject, Is-
 CapCategoryObject, IsCapCategoryOb-
 ject, IsCapCategoryObject, 114
 TestMonoidalPentagonIdentityForAll-
 QuadruplesInList
 for IsCapCategory, IsList, 115
 TestMonoidalPentagonIdentityUsingWith-
 GivenOperations
 for IsCapCategory, IsCapCategoryObject, Is-
 CapCategoryObject, IsCapCategoryOb-
 ject, IsCapCategoryObject, 114
 TestMonoidalTriangleIdentity
 for IsCapCategory, IsCapCategoryObject, Is-
 CapCategoryObject, 114
 TestMonoidalTriangleIdentityForAll-
 PairsInList
 for IsCapCategory, IsList, 114
 TestMonoidalUnitorsForInvertibility
 for IsCapCategory, IsCapCategoryObject,
 114
 TestZigZagIdentitiesForDual
 for IsCapCategory, IsCapCategoryObject,
 115
 TraceMap
 for IsCapCategoryMorphism, 42
 UniversalPropertyOfCoDual
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 39
 UniversalPropertyOfDual
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 24
 UniversalPropertyOfLeftCoDual
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 31
 UniversalPropertyOfLeftDual
 for IsCapCategoryObject, IsCapCategory-
 Object, IsCapCategoryMorphism, 15
 WriteFileForClosedMonoidalStructure,
 117
 WriteFileForCoclosedMonoidalStructure,
 118
 WriteFileForLeftClosedMonoidal-
 Structure, 117
 WriteFileForLeftCoclosedMonoidal-
 Structure, 118
 WriteFileForMonoidalStructure, 117