Relational Association Paradigm

C++ Generic Programming

Trajection Example

	Trajection		Page
Association	**Namespace**	trajection	▨
	Conformation Template	Referential	3
	Classification Template	Vectorial	3
	Classification Template	Locational	3
	Classification Template	Traversal	4
	Classification Template	Lineal	4
	Classification Template	Transpositional	5
	Classification Template	Directional	5
	Classification Template	Axial	5

Association which classifies template vectorial and lineal trajection through spaces to subspaces.

Conformation Template	trajection::Referential	
Template Parameters	Subjective	Type
Member	at	Subjective

Conformation template used for syntactual clarity when returning a reference from a function.

Classification Template	trajection::Vectorial	
Template Parameters	Spatial	Type
	Positional	Type
	Endemical	Type
Member	to	
	Return	Referential< Endemical >
	Parameters	Spatial&
		const Positional&
	from	
	Return	Referential< const Endemical >
	Parameters	const Spatial&
		const Positional&

Classification template which models vectorial trajection through a space to a subspace by position.

Classification Template	trajection::Locational	
Template Parameters	Spatial	Type
	Positional	Type
Member	contains	
	Return	bool
	Parameters	const Spatial&
		const Positional&

Classification template which models querying the existence of subspace in a space by position.

Classification Template	trajection::Traversal	
Template Parameters	Spatial	Type
	Positional	Type
	Endemical	Type
Member	vector	Vectorial< Spatial, Positional, Endemical >&
	locator	Locational< Spatial, Positional, Endemical >&

Classification which models vectorial and locational objectification for combined use.

Classification Template	trajection::Lineal	
Template Parameters	Spatial	Type
	Positional	Type
	Endemical	Type
Member	begin	
	Return	▨
	Parameters	Spatial&
		Positional&
	traverse	
	Return	▨
	Parameters	Positional&
	to	
	Return	Referential< Endemical >
	Parameters	const Positional&
	from	
	Return	Referential< const Endemical >
	Parameters	const Positional&

Classification template which models lineal trajection through a space to a subspace by positional traversals.

Classification Template	trajection::Transpositional	
Template Parameters	Spatial	Type
	Positional	Type
Member	begins	
	Return	bool
	Parameters	const Spatial&
	traversable	
	Return	bool
	Parameters	const Positional&

Classification template which models querying if a lineal trajection begins for a space or is traversable from a position.

Classification Template	trajection::Directional	
Template Parameters	Spatial	Type
	Positional	Type
	Endemical	Type
Member	liner	Lineal< Spatial, Positional, Endemical >&
	transposer	Transpositional< Spatial, Positional, Endemical >&

Classification template which models lineal and transpositional objectification for combined use.

Classification Template	trajection::Axial	
Template Parameters	Spatial	Type
	Positional	Type
	Endemical	Type
Member	increment	Directional< Spatial, Positional, Endemical >&
	decrement	Directional< Spatial, Positional, Endemical >&

Classification template which models incremental and decremental directional objectification for combined use.

	Numeration		Page
Association	**Namespace**	numeration	
	Classification Template	Lineal	7
	Classification Template	Transpositional	7
	Classification Template	Directional	7
	Classification Template	Axial	7
	Function Template	template <typename Integral> bool Begins(const Integral &)	7
	Function Template	template <typename Integral, Integral beginning, Integral ending> bool FiniteBegins(const Integral &)	8
	Function Template	template <typename Integral> void Begin(const Integral &, Integral &)	8
	Function Template	template <typename Integral, Integral beginning, Integral ending> void FiniteBegin(const Integral &, Integral &)	8
	Function Template	template <typename Integral> trajection::Referential< const Integral > To(const Integral &)	8
	Function Template	template <typename Integral, Integral beginning, Integral ending> trajection::Referential< const Integral > FiniteTo(const Integral &)	9
	Inner Association	Increment	9 to 12
	Inner Association	Decrement	13 to 16
	Objectification Template	Axis	17
	Objectification Template	FiniteAxis	17

Association which implements axial trajection through an integral space.

Classification Template	numeration::Lineal	
Template Parameters	Integral	Type
Alias	trajection::Lineal< const Integral, Integral, const Integral >	

Classification template alias which models lineal trajection through an integral space.

Classification Template	numeration::Transpositional	
Template Parameters	Integral	Type
Alias	trajection::Transpositional< const Integral, Integral >	

Classification template alias which models querying if a lineal trajection begins for an integral space or is traversable from an integral position.

Classification Template	numeration::Directional	
Template Parameters	Integral	Type
Alias	trajection::Directional< const Integral, Integral, const Integral >	

Classification template alias which models integral lineal and transpositional objectification for combined use.

Classification Template	numeration::Axial	
Template Parameters	Integral	Type
Alias	trajection::Axial< const Integral, Integral, const Integral >	

Classification template alias which models integral incremental and decremental directional objectification for combined use.

Function Template	numeration::Begins	
Template Parameters	Integral	Type
Return	bool	
Parameters	integer	const Integral&

This function template returns true if integer is non-zero and false if it is zero.

Function Template	numeration::FiniteBegins	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Return	bool	
Parameters	integer	const Integral&

This function template returns true if integer is between beginning and ending inclusively.

Function Template	numeration::Begin	
Template Parameters	Integral	Type
Return		
Parameters	value	const Integral&
	integer	Integral&

This function template assigns value to integer.

Function Template	numeration::FiniteBegin	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Return		
Parameters	value	const Integral&
	integer	Integral&

- Throws value if it is not within the sequence

This function template assigns value to integer.

Function Template	numeration::To	
Template Parameters	Integral	Type
Return	trajection::Referential< const Integral >	
Parameters	integer	const Integral&

This function template returns an explicit reference to integer as the subspace is the position as a constant.

Function Template	numeration::FiniteTo	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Return	trajection::Referential< const Integral >	
Parameters	integer	const Integral&

- Throws integer if it is not within the sequence

This function template returns an explicit reference to integer as the subspace is the position as a constant.

	Increment		**Page**
Inner Association	**Namespace**	::numeration::increment	▨
	Function Template	template <typename Integral> bool Traversable(const Integral &)	10
	Function Template	template <typename Integral, Integral beginning, Integral ending> bool FiniteTraversable(const Integral &)	10
	Function Template	template <typename Integral> void Traverse(Integral &)	10
	Function Template	template <typename Integral, Integral beginning, Integral ending> void FiniteTraverse(Integral &)	10
	Objectification Template	Liner	11
	Objectification Template	FiniteLiner	11
	Objectification Template	Transposer	11
	Objectification Template	FiniteTransposer	12
	Objectification Template	Direction	12
	Objectification Template	FiniteDirection	12

Association containing incremental numeration function and objectification.

Function Template	numeration::increment::Traversable	
Template Parameters	Integral	Type
Return	bool	
Parameters	integer	const Integral&

This function template returns true if integer is smaller than integer plus one.

Function Template	numeration::increment::FiniteTraversable	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Return	bool	
Parameters	integer	const Integral&

This function template returns true if integer is equal to or greater than beginning and smaller than ending.

Function Template	numeration::increment::Traverse	
Template Parameters	Integral	Type
Return		
Parameters	integer	Integral&

This function template traverses the integer position by increment.

Function Template	numeration::increment::FiniteTraverse	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Return		
Parameters	integer	const Integral&

- Throws integer if is in not within the sequence

This function template traverses the integer position by increment.

Objectification Template	numeration::increment::Liner	
Template Parameters	Integral	Type
Classification	Lineal< Integral >	
Initialization	Begin< Integral >	
	Traverse< Integral >	
	To< Integral >	
	To< Integral >	

This objectification template is used to reference template specified instances of the incremental lineal over integral domain function template definitions.

Objectification Template	numeration::increment::FiniteLiner	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Classification	Lineal< Integral >	
Initialization	FiniteBegin< Integral, beginning, ending >	
	FiniteTraverse< Integral, beginning, ending >	
	FiniteTo< Integral, beginning, ending >	
	FiniteTo< Integral, beginning, ending >	

This objectification template is used to reference template specified instances of the incremental lineal over finite integral sequences function template definitions.

Objectification Template	numeration::increment::Transposer	
Template Parameters	Integral	Type
Classification	Transpositional< Integral >	
Initialization	Begins< Integral >	
	Traversable< Integral >	

This objectification template is used to reference template specified instances of the incremental transpositional over integral domain function template definitions.

Objectification Template	numeration::increment::FiniteTransposer	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Classification	Transpositional< Integral >	
Initialization	FiniteBegins< Integral, beginning, ending >	
	FiniteTraversable< Integral, beginning, ending >	

This objectification template is used to reference template specified instances of the incremental transpositional over integral sequences function template definitions.

Objectification Template	numeration::increment::Direction	
Template Parameters	Integral	Type
Classification	Directional< Integral >	
Initialization	Liner< Integral >	
	Transposer< Integral >	

This objectification template is used to reference template specified instances of the incremental directional over integral domain function template definitions.

Objectification Template	numeration::increment::FiniteDirection	
Template Parameters	Integral	Type
Classification	Directional< Integral >	
Initialization	FiniteLiner< Integral >	
	FiniteTransposer< Integral >	

This objectification template is used to reference template specified instances of the incremental directional over integral sequences function template definitions.

	Decrement		Page
Inner Association	**Namespace**	numeration::decrement	
	Function Template	template <typename Integral> bool Traversable(const Integral &)	13
	Function Template	template <typename Integral, Integral beginning, Integral ending> bool FiniteTraversable(const Integral &)	14
	Function Template	template <typename Integral> void Traverse(Integral &)	14
	Function Template	template <typename Integral, Integral beginning, Integral ending> void FiniteTraverse(Integral &)	14
	Objectification Template	Liner	15
	Objectification Template	FiniteLiner	15
	Objectification Template	Transposer	15
	Objectification Template	FiniteTransposer	16
	Objectification Template	Direction	16
	Objectification Template	FiniteDirection	16

Association containing decremental numeration function and objectification.

Function Template	numeration::decrement::Traversable	
Template Parameters	Integral	Type
Return	bool	
Parameters	integer	const Integral&

This function template returns true if integer is greater than integer minus one.

Function Template	numeration::decrement::FiniteTraversable	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Return	bool	
Parameters	integer	const Integral&

This function template returns true if integer is greater than beginning and smaller than or equal to ending.

Function Template	numeration::decrement::Traverse	
Template Parameters	Integral	Type
Return	▨	
Parameters	integer	Integral&

This function template traverses the integer position by decrement.

Function Template	numeration::decrement::FiniteTraverse	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Return	▨	
Parameters	integer	const Integral&

- Throws integer if it is not within the sequence

This function template traverses the integer position by decrement.

Objectification Template	numeration::decrement::Liner	
Template Parameters	Integral	Type
Classification	const Lineal< Integral >	
Initialization	Begin< Integral >	
	Traverse< Integral >	
	To< Integral >	
	To< Integral >	

This objectification template is used to reference template specified instances of the decremental lineal over integral domain function template definitions.

Objectification Template	numeration::decrement::FiniteLiner	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Classification	const Lineal< Integral >	
Initialization	FiniteBegin< Integral, beginning, ending >	
	FiniteTraverse< Integral, beginning, ending >	
	FiniteTo< Integral, beginning, ending >	
	FiniteTo< Integral, beginning, ending >	

This objectification template is used to reference template specified instances of the decremental lineal over finite integral sequences function template definitions.

Objectification Template	numeration::decrement::Transposer	
Template Parameters	Integral	Type
Classification	const Transpositional< Integral >	
Initialization	Begins< Integral >	
	Traversable< Integral >	

This objectification template is used to reference template specified instances of the decremental transpositional over integral domain function template definitions.

Objectification Template	numeration::decrement::FiniteTransposer	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Classification	const Transpositional< Integral >	
Initialization	FiniteBegins< Integral, beginning, ending >	
	FiniteTraversable< Integral, beginning, ending >	

This objectification template is used to reference template specified instances of the decremental transpositional over integral sequences function template definitions.

Objectification Template	numeration::decrement::Direction	
Template Parameters	Integral	Type
Classification	const Directional< Integral >	
Initialization	Liner< Integral >	
	Transposer< Integral >	

This objectification template is used to reference template specified instances of the decremental directional over integral domain function template definitions.

Objectification Template	numeration::decrement::FiniteDirection	
Template Parameters	Integral	Type
Classification	const Directional< Integral >	
Initialization	FiniteLiner< Integral >	
	FiniteTransposer< Integral >	

This objectification template is used to reference template specified instances of the decremental directional over integral sequences function template definitions.

Objectification Template	numeration::Axis	
Template Parameters	Integral	Type
Classification	const Axial< Integral >	
Initialization	increment::Direction< Integral >	
	decrement::Direction< Integral >	

This objectification template is used to reference template specified instances of the axial over integral domain function template definitions.

Objectification Template	numeration::FiniteAxis	
Template Parameters	Integral	Type
	beginning	Integral
	ending	Integral
Classification	const Axial< Integral >	
Initialization	increment::FiniteDirection< Integral, beginning, ending >	
	decrement::FiniteDirection< Integral, beginning, ending >	

This objectification template is used to reference template specified instances of the axial over integral sequences function template definitions.

	Localization		Page
Association	**Namespace**	localization	
	Classification Template	Vectorial	18
	Classification Template	Lineal	19
	Function Template	template <typename Indexical, typename Elemental> trajection::Referential< Elemental > To(Elemental *const &, const Indexical &)	19
	Function Template	template <typename Indexical, typename Elemental> trajection::Referential< const Elemental > From(Elemental *const &, const Indexical &)	19
	Function Template	template <typename Elemental> void Begin(Elemental *const &, Elemental* &)	19
	Function Template	template <typename Elemental> trajection::Referential< Elemental > To(Elemental *const &)	20
	Function Template	template <typename Elemental> trajection::Referential< const Elemental > From(Elemental *const &)	20
	Inner Association	Increment	20 to 21
	Inner Association	Decrement	21 to 22
	Objectification Template	Vector	22

Association which implements vectorial and lineal based trajection through a space relative to a pointer. Both the incremental and decremental lineal trajection begin at the pointer's origin.

Classification Template	localization::Vectorial	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Vectorial< const Elemental*, Indexical, Elemental >	

Classification template alias which models vectorial trajection through a space relative to a pointer.

Classification Template	localization::Lineal	
Template Parameters	Elemental	Type
Alias	trajection::Lineal< const Elemental*, Elemental*, Elemental >	

Classification template alias which models lineal trajection through a space relative to a pointer.

Function Template	localization::To	
Template Parameters	Indexical	Type
	Elemental	Type
Return	trajection::Referential< Elemental >	
Parameters	pointer	Elemental* const&
	index	const Indexical&

This function template returns an explicit reference to the element at subscript index in the space relative to pointer.

Function Template	localization::From	
Template Parameters	Indexical	Type
	Elemental	Type
Return	trajection::Referential< Elemental >	
Parameters	pointer	Elemental* const&
	index	const Indexical&

This function template returns an explicit reference to the element as a constant at subscript index in the space relative to pointer.

Function Template	localization::Begin	
Template Parameters	Indexical	Type
	Elemental	Type
Return	▨	
Parameters	pointer	Elemental* const&
	point	Elemental*&

This function template always sets point to pointer.

Function Template	localization::To	
Template Parameters	Elemental	Type
Return	trajection::Referential< Elemental >	
Parameters	point	Elemental* const&

This function template returns an explicit reference to the element at point.

Function Template	localization::From	
Template Parameters	Elemental	Type
Return	trajection::Referential< Elemental >	
Parameters	point	Elemental* const&

This function template returns an explicit reference to the element as a constant at point.

Inner Association	**Increment**		**Page**
	Namespace	localization::increment	
	Function Template	template <typename Elemental> void Traverse(Elemental* &)	20
	Objectification Template	Liner	21

Association containing incremental localization function and objectification.

Function Template	localization::increment::Traverse	
Template Parameters	Elemental	Type
Return		
Parameters	point	Elemental*&

This function template traverses the point position by increment.

Objectification Template	localization::increment::Liner	
Template Parameters	Elemental	Type
Classification	const Lineal< Elemental >	
Initialization	Begin< Elemental >	
	Traverse< Elemental >	
	To< Elemental >	
	From< Elemental >	

This objectification template is used to reference template specified instances of the incremental lineal over pointer relative space function template definitions.

Inner Association	**Decrement**		**Page**
	Namespace	localization::decrement	▨
	Function Template	template <typename Elemental> void Traverse(Elemental* &)	21
	Objectification Template	Liner	22

Association containing decremental localization function and objectification.

Function Template	localization::decrement::Traverse	
Template Parameters	Elemental	Type
Return	▨	▨
Parameters	point	Elemental*&

This function template traverses the point position by decrement.

Objectification Template	localization::decrement::Liner	
Template Parameters	Elemental	Type
Classification	const Lineal< Elemental >	
Initialization	Begin< Elemental >	
	Traverse< Elemental >	
	To< Elemental >	
	From< Elemental >	

This objectification template is used to reference template specified instances of the decremental lineal over pointer relative space function template definitions.

Objectification Template	localization::Vector	
Template Parameters	Indexical	Type
	Elemental	Type
Classification	const Vectorial< Elemental >	
Initialization	To< Indexical, Elemental >	
	From< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the vectorial over pointer relative space function template definitions.

Association	Segmentation		Page
	Namespace	segmentation	
	Conformation Template	Segmental	25
	Classification Template	Vectorial	25
	Classification Template	Locational	25
	Classification Template	Traversal	25
	Classification Template	Lineal	25
	Classification Template	Transpositional	26
	Classification Template	Directional	26
	Classification Template	Axial	26
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool Contains(Elemental *const &, const Indexical &)	26
	Function Template	template <typename Indexical, typename Elemental> trajection::Referential< Elemental > To(Elemental *const &, const Indexical &)	27
	Function Template	template <typename Indexical, typename Elemental, Indexical length> trajection::Referential< Elemental > SafeTo(Elemental *const &, const Indexical &)	27
	Function Template	template <typename Indexical, typename Elemental> trajection::Referential< const Elemental > From(Elemental *const &, const Indexical &)	27
	Function Template	template <typename Indexical, typename Elemental, Indexical length> trajection::Referential< const Elemental > SafeFrom(Elemental *const &, const Indexical &)	28

	Function Template	template <typename Indexical, typename Elemental> trajection::Referential< Elemental > To(const Segmental< Indexical, Elemental > &)	28
	Function Template	template <typename Indexical, typename Elemental, Indexical length> trajection::Referential< Elemental > SafeTo(const Segmental< Indexical, Elemental > &)	28
	Function Template	template <typename Indexical, typename Elemental> trajection::Referential< const Elemental > From(const Segmental< Indexical, Elemental > &)	
	Function Template	template <typename Indexical, typename Elemental, Indexical length> trajection::Referential< const Elemental > SafeFrom(const Segmental< Indexical, Elemental > &)	
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool Begins(Elemental *const &)	
	Inner Association	Increment	
	Inner Association	Decrement	
	Objectification Template	Vector	
	Objectification Template	SafeVector	
	Objectification Template	Locator	
	Objectification Template	Traverse	
	Objectification Template	Axis	
	Objectification Template	SafeAxis	

Association which implements traversal and axial based trajection through a space relative to a pointer where the range of contiguously repeating elements in the space is a compile time constant. The incremental lineal trajection begins at the pointer's origin and the decremental lineal trajection begins at the last element in the forward contiguous space relative to the pointer.

Conformation Template	segmentation::Segmental	
Template Parameters	Indexical	Type
	Elemental	Type
Members	pointer	Elemental*
	index	Indexical

Conformation template used during lineal trajection as the positional type template.

Classification Template	segmentation::Vectorial	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Vectorial< Elemental *const, Indexical, Elemental >	

Classification template alias which models vectorial trajection through a segment space.

Classification Template	segmentation::Locational	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Locational< Elemental *const, Indexical >	

Classification template alias which models querying if a position exists in a segment space.

Classification Template	segmentation::Traversal	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Traversal< Elemental *const, Indexical, Elemental >	

Classification template alias which models vectorial and locational objectification for combined use.

Classification Template	segmentation::Lineal	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Lineal< Elemental *const, Segmental< Indexical, Elemental >, Elemental >	

Classification template alias which models lineal trajection through a segment space.

Classification Template	segmentation::Transpositional	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Transpositional< Elemental *const, Segmental< Indexical, Elemental >, Elemental >	

Classification template alias which models querying if a lineal trajection begins for a segment space or is traversable from a position.

Classification Template	segmentation::Directional	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Directional< Elemental *const, Segmental< Indexical, Elemental >, Elemental >	

Classification template alias which models segment space lineal and transpositional objectification for combined use.

Classification Template	segmentation::Axial	
Template Parameters	Indexical	Type
	Elemental	Type
Alias	trajection::Axial< Elemental *const, Segmental< Indexical, Elemental >, Elemental >	

Classification template alias which models segment space incremental and decremental directional objectification for combined use.

Function Template	segmentation::Contains	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	pointer	Elemental *const &
	index	const Indexical &

This function template returns true if index is a valid subscript index within the pointer relative space.

Function Template	segmentation::To	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	pointer	Elemental *const &
	index	const Indexical &

This function template returns an explicit reference to the element at subscript index in the space relative to pointer.

Function Template	segmentation::SafeTo	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	pointer	Elemental *const &
	index	const Indexical &

- Throws index if it is not within range determined by length

This function template returns an explicit reference to the element at subscript index in the space relative to pointer.

Function Template	segmentation::From	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	pointer	Elemental *const &
	index	const Indexical &

This function template returns an explicit reference to the element as a constant at subscript index in the space relative to pointer.

Function Template	segmentation::SafeFrom	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	pointer	Elemental *const &
	index	const Indexical &

- Throws index if it is not within range determined by length

This function template returns an explicit reference to the element as a constant at subscript index in the space relative to pointer.

Function Template	segmentation::To	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	position	const Segmental< Indexical, Elemental > &

This function template returns an explicit reference to the element at position.

Function Template	segmentation::SafeTo	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	position	const Segmental< Indexical, Elemental > &

- Throws index if it is not within range determined by length

This function template returns an explicit reference to the element at position.

Function Template	segmentation::From	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	position	const Segmental< Indexical, Elemental > &

This function template returns an explicit reference to the element as a constant at position.

Function Template	segmentation::SafeFrom	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	position	const Segmental< Indexical, Elemental > &

- Throws the positional index if it is not within range determined by length

This function template returns an explicit reference to the element as a constant at position.

Function Template	segmentation::Begins	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	pointer	Elemental *const &

This function template returns true if pointer is not null and length is greater than 0.

	Increment		Page
Inner Association	**Namespace**	segmentation::increment	▨
	Function Template	template <typename Indexical, typename Elemental> void Begin(Elemental *const &, Segmental< Indexical, Elemental > &)	31
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void BeginSafely(Elemental *const &, Segmental< Indexical, Elemental > &)	31
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool Traversable(const Segmental< Indexical, Elemental > &)	31
	Function Template	template <typename Indexical, typename Elemental> void Traverse(Segmental< Indexical, Elemental > &)	32
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void TraverseSafely(Segmental< Indexical, Elemental > &)	32
	Objectification Template	Liner	32
	Objectification Template	SafeLiner	33
	Objectification Template	Transposer	33
	Objectification Template	Direction	34
	Objectification Template	SafeDirection	34

Association containing incremental segmentation function and objectification.

Function Template	segmentation::increment::Begin	
Template Parameters	Indexical	Type
	Elemental	Type
Return		
Parameters	pointer	Elemental *const &
	position	Segmental< Indexical, Elemental > &

This function template assigns pointer to positional pointer and 0 to the positional index.

Function Template	segmentation::increment::BeginSafely	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return		
Parameters	pointer	Elemental *const &
	position	Segmental< Indexical, Elemental > &

- Throws pointer if it is null
- Throws length if it is less than 1

This function template assigns pointer to positional pointer and 0 to the positional index.

Function Template	segmentation::increment::Traversable	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	position	const Segmental< Indexical, Elemental > &

This function returns true if the positional index is greater than or equal to 0 and less than length – 1.

Function Template	segmentation::increment::Traverse	
Template Parameters	Indexical	Type
	Elemental	Type
Return		
Parameters	position	Segmental< Indexical, Elemental > &

This function template traverses the positional index by increment.

Function Template	segmentation::increment::TraverseSafely	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return		
Parameters	position	Segmental< Indexical, Elemental > &

- Throws the positional index if it is not within range determined by length

This function template traverses the positional index by increment.

Objectification Template	segmentation::increment::Liner	
Template Parameters	Indexical	Type
	Elemental	Type
Classification	Lineal< Indexical, Elemental >	
Initialization	Begin< Indexical, Elemental >	
	Traverse< Indexical, Elemental >	
	To< Indexical, Elemental >	
	From< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the incremental lineal over segment space function template definitions.

Objectification Template	segmentation::increment::SafeLiner	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Lineal< Indexical, Elemental >	
Initialization	BeginSafely< Indexical, Elemental, length >	
	TraverseSafely< Indexical, Elemental, length >	
	SafeTo< Indexical, Elemental, length >	
	SafeFrom< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the incremental lineal over segment space safe function template definitions.

Objectification Template	segmentation::increment::Transposer	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Transpositional< Indexical, Elemental >	
Initialization	Begins< Indexical, Elemental >	
	Traversable< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the incremental transpositional over segment space function template definitions.

Objectification Template	segmentation::increment::Direction	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental >	
Initialization	Liner< Indexical, Elemental >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the incremental directional over segment space function template definitions.

Objectification Template	segmentation::increment::SafeDirection	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental >	
Initialization	SafeLiner< Indexical, Elemental, length >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the incremental directional over segment space safe function template definitions.

	Decrement		Page
Inner Association	**Namespace**	segmentation::decrement	
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void Begin(Elemental *const &, Segmental< Indexical, Elemental > &)	36
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void BeginSafely(Elemental *const &, Segmental< Indexical, Elemental > &)	36
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool Traversable(const Segmental< Indexical, Elemental > &)	36
	Function Template	template <typename Indexical, typename Elemental> void Traverse(Segmental< Indexical, Elemental > &)	37
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void TraverseSafely(Segmental< Indexical, Elemental > &)	37
	Objectification Template	Liner	37
	Objectification Template	SafeLiner	38
	Objectification Template	Transposer	38
	Objectification Template	Direction	39
	Objectification Template	SafeDirection	39

Association containing decremental segmentation function and objectification.

Function Template	segmentation::decrement::Begin	
Template Parameters	Indexical	Type
	Elemental	Type
Return	▨	
Parameters	pointer	Elemental *const &
	position	Segmental< Indexical, Elemental > &

This function template assigns pointer to the positional pointer and length - 1 to the positional index.

Function Template	segmentation::decrement::BeginSafely	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	▨	
Parameters	pointer	Elemental *const &
	position	Segmental< Indexical, Elemental > &

- Throws pointer if it is null
- Throws length if it is less than 1

This function template assigns pointer to the positional pointer and length - 1 to the positional index.

Function Template	segmentation::decrement::Traversable	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	position	const Segmental< Indexical, Elemental > &

This function returns true if the positional index is greater than 0 and less than length.

Function Template	segmentation::decrement::Traverse	
Template Parameters	Indexical	Type
	Elemental	Type
Return		
Parameters	position	Segmental< Indexical, Elemental > &

This function template traverses the positional index by decrement.

Function Template	segmentation::decrement::TraverseSafely	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return		
Parameters	position	Segmental< Indexical, Elemental > &

- Throws the positional index if it is not within range determined by length

This function template traverses the positional index by decrement.

Objectification Template	segmentation::decrement::Liner	
Template Parameters	Indexical	Type
	Elemental	Type
Classification	Lineal< Indexical, Elemental >	
Initialization	Begin< Indexical, Elemental >	
	Traverse< Indexical, Elemental >	
	To< Indexical, Elemental >	
	From< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the decremental lineal over segment space function template definitions.

Objectification Template	segmentation::decrement::SafeLiner	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Lineal< Indexical, Elemental >	
Initialization	BeginSafely< Indexical, Elemental, length >	
	TraverseSafely< Indexical, Elemental, length >	
	SafeTo< Indexical, Elemental, length >	
	SafeFrom< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the decremental lineal over segment space safe function template definitions.

Objectification Template	segmentation::decrement::Transposer	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Transpositional< Indexical, Elemental >	
Initialization	Begins< Indexical, Elemental >	
	Traversable< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the decremental transpositional over segment space function template definitions.

Objectification Template	segmentation::decrement::Direction	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental >	
Initialization	Liner< Indexical, Elemental >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the decremental directional over segment space function template definitions.

Objectification Template	segmentation::decrement::SafeDirection	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental >	
Initialization	SafeLiner< Indexical, Elemental, length >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the decremental directional over segment space safe function template definitions.

Objectification Template	segmentation::Vector	
Template Parameters	Indexical	Type
	Elemental	Type
Classification	Vectorial< Indexical, Elemental >	
Initialization	To< Indexical, Elemental >	
	From< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the vectorial over segment space function template definitions.

Objectification Template	segmentation::SafeVector	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Vectorial< Indexical, Elemental >	
Initialization	SafeTo< Indexical, Elemental, length >	
	SafeFrom< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the vectorial over segment space safe function template definitions.

Objectification Template	segmentation::Locator	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Locational< Indexical, Elemental >	
Initialization	Contains< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the locational over segment space function template definitions.

Objectification Template	segmentation::Traverse	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Traversal< Indexical, Elemental >	
Initialization	SafeVector< Indexical, Elemental, length >	
	Locator< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the traversal over segment space function template definitions.

Objectification Template	segmentation::Axis	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Axial< Indexical, Elemental >	
Initialization	increment::Direction< Indexical, Elemental, length >	
	decrement::Direction< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the axial over segment space function template definitions.

Objectification Template	segmentation::SafeAxis	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Traversal< Indexical, Elemental >	
Initialization	increment::SafeDirection< Indexical, Elemental, length >	
	decrement::SafeDirection< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the axial over segment space safe function template definitions.

Association	**Ordination**		**Page**
	Namespace	ordination	
	Conformation Template	Ordinal	44
	Classification Template	Vectorial	44
	Classification Template	Locational	44
	Classification Template	Traversal	44
	Classification Template	Lineal	45
	Classification Template	Transpositional	45
	Classification Template	Directional	45
	Classification Template	Axial	45
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool Contains(const Elemental(&)[length], const Indexical &)	46
	Function Template	template <typename Indexical, typename Elemental, Indexical length> trajection::Referential< Elemental > To(Elemental(&)[length], const Indexical &)	46
	Function Template	template <typename Indexical, typename Elemental, Indexical length> trajection::Referential< Elemental > SafeTo(Elemental(&)[length], const Indexical &)	46
	Function Template	template <typename Indexical, typename Elemental, Indexical length> trajection::Referential< const Elemental > From(const Elemental(&)[length], const Indexical &)	47

Association which implements traversal and axial based trajection through an array space. The incremental lineal trajection begins at the first element in the array and the decremental lineal trajection begins at the last element in the array.

Conformation Template	ordination::Ordinal	
Template Parameters	Indexical	Type
	Elemental	Type
Members	array	Elemental*
	index	Indexical

Conformation template used during lineal trajection as the positional type template.

Classification Template	ordination::Vectorial	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Alias	trajection::Vectorial< Elemental[length], Indexical, Elemental >	

Classification template alias which models vectorial trajection through an array space.

Classification Template	ordination::Locational	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Alias	trajection::Locational< Elemental[length], Indexical >	

Classification template alias which models querying if a position exists in an array space.

Classification Template	ordination::Traversal	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Alias	trajection::Traversal< Elemental[length], Indexical, Elemental >	

Classification template alias which models vectorial and locational objectification for combined use.

Classification Template	ordination::Lineal	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Alias	trajection::Lineal< Elemental[length], Ordinal< Indexical, Elemental >, Elemental >	

Classification template alias which models lineal trajection through an array space.

Classification Template	ordination::Transpositional	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Alias	trajection::Transpositional< Elemental[length], Ordinal< Indexical, Elemental > >	

Classification template alias which models querying if a lineal trajection begins for an array space or is traversable from a position.

Classification Template	ordination::Directional	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Alias	trajection::Directional< Elemental[length], Ordinal< Indexical, Elemental >, Elemental >	

Classification template alias which models array space lineal and transpositional objectification for combined use.

Classification Template	ordination::Axial	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Alias	trajection::Axial< Elemental[length], Ordinal< Indexical, Elemental >, Elemental >	

Classification template alias which models array space incremental and decremental directional objectification for combined use.

Function Template	ordination::Contains	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	array	const Elemental (&)[length]
	index	const Indexical &

This function template returns true if index is a valid subscript index within the array space.

Function Template	ordination::To	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	array	const Elemental (&)[length]
	index	const Indexical &

This function template returns an explicit reference to the element at subscript index in the array space.

Function Template	ordination::SafeTo	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	array	const Elemental (&)[length]
	index	const Indexical &

- Throws index if it is not within range determined by length

This function template returns an explicit reference to the element at subscript index in the array space.

Function Template	ordination::From	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	array	const Elemental (&)[length]
	index	const Indexical &

This function template returns an explicit reference to the element as a constant at index.

Function Template	ordination::SafeFrom	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	array	const Elemental (&)[length]
	index	const Indexical &

- Throws index if it is not within range determined by length

This function template returns an explicit reference to the element as a constant at index.

Function Template	ordination::To	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	position	const Ordinal< Indexical, Elemental > &

This function template returns an explicit reference to the element at position.

Function Template	ordination::SafeTo	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< Elemental >	
Parameters	position	const Ordinal< Indexical, Elemental > &

- Throws index if it is not within range determined by length

This function template returns an explicit reference to the element at position.

Function Template	ordination::From	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	position	const Ordinal< Indexical, Elemental > &

This function template returns an explicit reference to the element as a constant at position.

Function Template	ordination::SafeFrom	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	trajection::Referential< const Elemental >	
Parameters	position	const Ordinal< Indexical, Elemental > &

- Throws index if it is not within range determined by length

This function template returns an explicit reference to the element as a constant at position.

Function Template	ordination::Begins	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	array	const Elemental(&)[length]

This function template returns true only if length is greater than 0.

	Increment		**Page**
Inner Association	**Namespace**	ordination::increment	▨
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void Begin(Elemental(&)[length], Ordinal< Indexical, Elemental > &)	50
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool Traversable(const Ordinal< Indexical, Elemental > &)	50
	Function Template	template <typename Indexical, typename Elemental> void Traverse(Ordinal< Indexical, Elemental > &)	50
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void TraverseSafely(Ordinal< Indexical, Elemental > &)	51
	Objectification Template	Liner	51
	Objectification Template	SafeLiner	52
	Objectification Template	Transposer	52
	Objectification Template	Direction	53
	Objectification Template	SafeDirection	53

Association containing incremental ordination function and objectification.

Function Template	ordination::increment::Begin	
Template Parameters	Indexical	Type
	Elemental	Type
Return		
Parameters	array	const Elemental (&)[length]
	index	const Indexical &

This function template assigns pointer to the positional pointer and 0 to positional index.

Function Template	ordination::increment::Traversable	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	position	const Ordinal< Indexical, Elemental > &

This function returns true if position's index greater than or equal to 0 and less than length – 1.

Function Template	ordination::increment::Traverse	
Template Parameters	Indexical	Type
	Elemental	Type
Return		
Parameters	position	const Ordinal< Indexical, Elemental > &

This function template traverses positional index by increment.

Function Template	ordination::increment::TraverseSafely	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return		
Parameters	position	const Ordinal< Indexical, Elemental > &

- Throws the positional index if it is not within range determined by length

This function template traverses positional index by increment.

Objectification Template	ordination::increment::Liner	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Lineal< Indexical, Elemental, length >	
Initialization	Begin< Indexical, Elemental, length >	
	Traverse< Indexical, Elemental >	
	To< Indexical, Elemental >	
	From< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the incremental lineal over array space function template definitions.

Objectification Template	ordination::increment::SafeLiner	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Lineal< Indexical, Elemental, length >	
Initialization	Begin< Indexical, Elemental, length >	
	TraverseSafely< Indexical, Elemental, length >	
	SafeTo< Indexical, Elemental, length >	
	SafeFrom< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the incremental lineal over array space safe function template definitions.

Objectification Template	ordination::increment::Transposer	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Transpositional< Indexical, Elemental, length >	
Initialization	Begins< Indexical, Elemental, length >	
	Traversable< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the incremental transpositional over array space function template definitions.

Objectification Template	ordination::increment::Direction	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental, length >	
Initialization	Liner< Indexical, Elemental, length >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the incremental directional over array space function template definitions.

Objectification Template	ordination::increment::SafeDirection	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental, length >	
Initialization	SafeLiner< Indexical, Elemental, length >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the incremental directional over array space safe function template definitions.

	Decrement		Page
Inner Association	**Namespace**	ordination::decrement	
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void Begin(Elemental(&)[length], Ordinal< Indexical, Elemental > &)	55
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool Traversable(const Ordinal< Indexical, Elemental > &)	55
	Function Template	template <typename Indexical, typename Elemental> void Traverse(Ordinal< Indexical, Elemental > &)	55
	Function Template	template <typename Indexical, typename Elemental, Indexical length> void TraverseSafely(Ordinal< Indexical, Elemental > &)	56
	Objectification Template	Liner	56
	Objectification Template	SafeLiner	57
	Objectification Template	Transposer	57
	Objectification Template	Direction	57
	Objectification Template	SafeDirection	58

Association containing decremental ordination function and objectification.

Function Template	ordination::decrement::Begin	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Type
Return		
Parameters	array	Elemental (&)[length]
	position	Ordinal< Indexical, Elemental > &

This function template assigns pointer to the positional pointer and length - 1 to the positional index.

Function Template	ordination::decrement::Traversable	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	position	Ordinal< Indexical, Elemental > &

This function returns true if the position's index greater than 0 and less than length.

Function Template	ordination::decrement::Traverse	
Template Parameters	Indexical	Type
	Elemental	Type
Return		
Parameters	position	Ordinal< Indexical, Elemental > &

This function template traverses the positional index by decrement.

Function Template	ordination::decrement::TraverseSafely	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return		
Parameters	position	Ordinal< Indexical, Elemental > &

- Throws the positional index if it is not within range determined by length

This function template traverses the positional index by decrement.

Objectification Template	ordination::decrement::Liner	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Lineal< Indexical, Elemental, length >	
Initialization	Begin< Indexical, Elemental, length >	
	Traverse< Indexical, Elemental >	
	To< Indexical, Elemental >	
	From< Indexical, Elemental >	

This objectification template is used to reference template specified instances of the decremental lineal over array space function template definitions.

Objectification Template	ordination::decrement::SafeLiner	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Lineal< Indexical, Elemental, length >	
Initialization	Begin< Indexical, Elemental, length >	
	TraverseSafely< Indexical, Elemental, length >	
	SafeTo< Indexical, Elemental, length >	
	SafeFrom< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the decremental lineal over array space safe function template definitions.

Objectification Template	ordination::decrement::Transposer	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Transpositional< Indexical, Elemental, length >	
Initialization	Begins< Indexical, Elemental, length >	
	Traversable< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the decremental transpositional over array space function template definitions.

Objectification Template	ordination::decrement::Direction	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental, length >	
Initialization	Liner< Indexical, Elemental, length >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the decremental directional over array space function template definitions.

Objectification Template	ordination::decrement::SafeDirection	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Directional< Indexical, Elemental, length >	
Initialization	SafeLiner< Indexical, Elemental, length >	
	Transposer< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the decremental directional over array space safe function template definitions.

Objectification Template	ordination::Vector	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Vectorial< Indexical, Elemental, length >	
Initialization	To< Indexical, Elemental, length >	
	From< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the vectorial over array space function template definitions.

Objectification Template	ordination::SafeVector	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Vectorial< Indexical, Elemental, length >	
Initialization	SafeTo< Indexical, Elemental, length >	
	SafeFrom< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the vectorial over array space safe function template definitions.

Objectification Template	ordination::Locator	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Locational< Indexical, Elemental, length >	
Initialization	Contains< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the locational over array space function template definitions.

Objectification Template	ordination::Traverse	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Traversal< Indexical, Elemental, length >	
Initialization	SafeVector< Indexical, Elemental, length >	
	Locator< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the traversal over array space function template definitions.

Objectification Template	ordination::Axis	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Axial< Indexical, Elemental, length >	
Initialization	increment::Direction< Indexical, Elemental, length >	
	decrement::Direction< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the axial over array space function template definitions.

Objectification Template	ordination::SafeAxis	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Classification	Traversal< Indexical, Elemental, length >	
Initialization	increment::SafeDirection< Indexical, Elemental, length >	
	decrement::SafeDirection< Indexical, Elemental, length >	

This objectification template is used to reference template specified instances of the axial over array space safe function template definitions.

	Junction		Page
Association	**Namespace**	junction	
	Conformation Template	Junctional	62
	Classification Template	Lineal	62
	Classification Template	Transpositional	62
	Classification Template	Directional	62
	Classification Template	Axial	62
	Function Template	template <typename Elemental> trajection::Referential< Elemental > To(Junctional< Elemental > *const &)	63
	Function Template	template <typename Elemental> trajection::Referential< const Elemental > From(Junctional< Elemental > *const &)	63
	Function Template	template <typename Elemental> void Begin(Junctional< Elemental >* &, Junctional< Elemental >* &)	63
	Function Template	template <typename Elemental> bool Begins(Junctional< Elemental > *const &)	63
	Inner Association	Increment	64 to 65
	Inner Association	Decrement	66 to 67
	Objectification Template	Axis	67

Association which implements axial based trajection through a linked list space. The lineal trajection always begins at the node submitted to the Begin function, therefore the tail node should be passed when intending a decremental lineal trajection.

Conformation Template	junction::Junctional	
Template Parameters	Elemental	Type
Members	previous	Junctional< Elemental > *
	next	Junctional< Elemental > *
	element	Elemental

Conformation template which is a node template for a linked list.

Classification Template	junction::Lineal	
Template Parameters	Elemental	Type
Alias	trajection::Lineal< Junctional< Elemental >*, Junctional< Elemental >*, Elemental >	

Classification template alias which models lineal trajection through a linked list space.

Classification Template	junction::Transpositional	
Template Parameters	Elemental	Type
Alias	trajection::Transpositional< Junctional< Elemental >*, Junctional< Elemental >* >	

Classification template alias which models querying if a lineal trajection begins for a linked list space or is traversable from a node position.

Classification Template	junction::Directional	
Template Parameters	Elemental	Type
Alias	trajection::Directional< Junctional< Elemental >*, Junctional< Elemental >*, Elemental >	

Classification template alias which models linked list space lineal and transpositional objectification for combined use.

Classification Template	junction::Axial	
Template Parameters	Elemental	Type
Alias	trajection::Axial< Junctional< Elemental >*, Junctional< Elemental >*, Elemental >	

Classification template alias which models linked list space incremental and decremental directional objectification for combined use.

Function Template	junction::To	
Template Parameters	Elemental	Type
Return	trajection::Referential< Elemental >	
Parameters	position	Junctional< Elemental > *const &

This function template returns an explicit reference to the element at position.

Function Template	junction::From	
Template Parameters	Elemental	Type
Return	trajection::Referential< const Elemental >	
Parameters	position	Junctional< Elemental > *const &

This function template returns an explicit reference to the element as a constant at position.

Function Template	junction::Begin	
Template Parameters	Elemental	Type
Return		
Parameters	list	Junctional< Elemental >* &
	position	Junctional< Elemental >* &

This function template assigns list to position.

Function Template	junction::Begins	
Template Parameters	Elemental	Type
Return	bool	
Parameters	list	Junctional< Elemental > *const &

This function template returns true if list is not null.

	Increment		Page
Inner Association	**Namespace**	junction::increment	▨
	Function Template	template <typename Elemental> void Traverse(Junctional< Elemental >* &)	64
	Function Template	template <typename Elemental> bool Traversable(Junctional< Elemental > *const &)	64
	Objectification Template	Liner	65
	Objectification Template	Transposer	65
	Objectification Template	Direction	65

Association containing incremental junction function and objectification.

Function Template	junction::increment::Traverse	
Template Parameters	Elemental	Type
Return	▨	
Parameters	position	Junctional< Elemental >* &

This function template traverses position to the next node position.

Function Template	junction::increment::Traversable	
Template Parameters	Elemental	Type
Return	bool	
Parameters	position	Junctional< Elemental > *const &

This function returns true if the next position is not null.

Objectification Template	junction::increment::Liner	
Template Parameters	Elemental	Type
Classification	Lineal< Elemental >	
Initialization	Begin< Elemental >	
	Traverse< Elemental >	
	To< Elemental >	
	From< Elemental >	

This objectification template is used to reference template specified instances of the incremental lineal over linked list space function template definitions.

Objectification Template	junction::increment::Transposer	
Template Parameters	Elemental	Type
Classification	Transpositional< Elemental >	
Initialization	Begins< Elemental >	
	Traversable< Elemental >	

This objectification template is used to reference template specified instances of the incremental transpositional over linked list space function template definitions.

Objectification Template	junction::increment::Direction	
Template Parameters	Elemental	Type
Classification	Directional< Elemental >	
Initialization	Liner< Elemental >	
	Transposer< Elemental >	

This objectification template is used to reference template specified instances of the incremental directional over linked list space function template definitions.

		Decrement	Page
Inner Association	**Namespace**	junction::decrement	▨
	Function Template	template <typename Elemental> void Traverse(Junctional< Elemental >* &)	66
	Function Template	template <typename Elemental> bool Traversable(Junctional< Elemental > *const &)	66
	Objectification Template	Liner	67
	Objectification Template	Transposer	67
	Objectification Template	Direction	67

Association containing decremental junction function and objectification.

Function Template	junction::decrement::Traverse	
Template Parameters	Elemental	Type
Return	▨	
Parameters	point	Junctional< Elemental >* &

This function template traverses the position to the previous node position.

Function Template	junction::decrement::Traversable	
Template Parameters	Elemental	Type
Return	bool	
Parameters	point	Junctional< Elemental > *const &

This function returns true if the previous position is not null.

Objectification Template	junction::decrement::Liner	
Template Parameters	Elemental	Type
Classification	Lineal< Elemental >	
Initialization	Begin< Elemental >	
	Traverse< Elemental >	
	To< Elemental >	
	From< Elemental >	

This objectification template is used to reference template specified instances of the decremental lineal over linked list space function template definitions.

Objectification Template	junction::decrement::Transposer	
Template Parameters	Elemental	Type
Classification	Transpositional< Elemental >	
Initialization	Begins< Elemental >	
	Traversable< Elemental >	

This objectification template is used to reference template specified instances of the decremental transpositional over linked list space function template definitions.

Objectification Template	junction::decrement::Direction	
Template Parameters	Elemental	Type
Classification	Directional< Elemental >	
Initialization	Liner< Elemental >	
	Transposer< Elemental >	

This objectification template is used to reference template specified instances of the decremental directional over linked list space function template definitions.

Objectification Template	junction::Axis	
Template Parameters	Elemental	Type
Classification	Axial< Elemental >	
Initialization	increment::Direction< Elemental >	
	decrement::Direction< Elemental >	

This objectification template is used to reference template specified instances of the axial over linked list space function template definitions.

Association	Example		Page
	Using	trajection::Vectorial trajection::Traversal trajection::Lineal trajection::Directional	
	Classification Template	Numeral	69
	Information	TitleFormat	69
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool TestNumerator(const Numeral< Indexical > &, const Elemental(&)[length])	69
	Function Template	template <typename Spatial, typename Indexical, typename Endemical, Indexical length> bool TestVector(const Vectorial< Spatial, Indexical, Endemical > &, Spatial &, const Numeral< Indexical > &, const Endemical (&)[length])	70
	Function Template	template <typename Spatial, typename Indexical, typename Endemical, Indexical length> bool TestTraverse(const Traversal< Spatial, Indexical, Endemical > &, Spatial &, const Numeral< Indexical > &, const Endemical (&)[length])	70
	Function Template	template <typename Spatial, typename Positional, typename Endemical, typename Indexical, Indexical length> bool TestLiner(const Lineal< Spatial, Positional, Endemical > &, Spatial &, const Numeral< Indexical > &, const Endemical (&)[length])	71
	Function Template	template <typename Spatial, typename Positional, typename Endemical, typename Indexical, Indexical length> bool TestDirection(const Directional< Spatial, Positional, Endemical > &, Spatial &, const Numeral< Indexical > &, const Endemical (&)[length])	71
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool TestPointer(const Numeral< Indexical > &, const Elemental (&)[length])	72

	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool TestSegment(const Numeral< Indexical > &, const Elemental (&)[length])	72
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool TestArray(const Numeral< Indexical > &, const Elemental (&)[length])	72
	Function Template	template <typename Indexical, typename Elemental, Indexical length> bool TestLinkedList(const Numeral< Indexical > &, const Elemental (&)[length])	73
	Function	int main()	73

Association (file scope instead of namespace based) which runs simple tests to verify some of the functionality of the spatial trajection implementations.

Classification Template	Numeral	
Template Parameters	Integral	Type
Alias	numeration::Axial< Integral >	

Classification template alias which models axial trajection through an integral space.

Information	TitleFormat
Conformation	const char* const

Information used to display each test.

Function Template	TestNumerator	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	numerator	const Numeral< Indexical > &
	values	const Elemental(&)[length]

Function template which tests both of the incremental and decremental directions of the numerator verifying consistency between direct control of an integer and lineal traversal functions on another integer.

Function Template	TestVector	
Template Parameters	Spatial	Type
	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	vector	const Vectorial< Spatial, Indexical, Endemical > &
	space	Spatial &
	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which tests vectorial trajection to write then read back the sample values provided. It assumes that there are at least length elements in the space and uses the provided numerator to count.

Function Template	TestTraverse	
Template Parameters	Spatial	Type
	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	traverse	const Traversal< Spatial, Indexical, Endemical > &
	space	Spatial &
	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which tests traversal trajection to write then read back the sample values provided. It verifies that each element is a valid position in the space but requires at least length elements to conduct the test successfully and uses the provided numerator to count.

Function Template	TestLiner	
Template Parameters	Spatial	Type
	Positional	Type
	Elemental	Type
	Indexical	Type
	length	Indexical
Return	bool	
Parameters	liner	const Lineal< Spatial, Positional, Endemical > &
	space	Spatial &
	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which tests lineal trajection to write then read back the sample values provided. It assumes that there are at least length elements in the space and uses the provided numerator to count.

Function Template	TestDirection	
Template Parameters	Spatial	Type
	Positional	Type
	Elemental	Type
	Indexical	Type
	length	Indexical
Return	bool	
Parameters	direction	const Directional< Spatial, Positional, Endemical > &
	space	Spatial &
	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which tests directional trajection to write then read back the sample values provided. It verifies that the spatial trajectory begins and that the positions are each traversable until the last element and uses the provided numerator to count.

Function Template	TestPointer	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which allocates an array of elements on the heap to a first pointer and assigns the last element to a last pointer. It then conducts vectorial and increasing lineal tests on the first pointer and a decreasing lineal test on the last pointer. It deletes the array on the heap when done.

Function Template	TestSegment	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which allocates an array of elements on the heap to a pointer. It then conducts traversal, increasing and decreasing directional and finally increasing and decreasing safe directional tests on the pointer.

Function Template	TestArray	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which conducts traversal, increasing and decreasing directional and finally increasing and decreasing safe directional tests on the array.

Function Template	TestLinkedList	
Template Parameters	Indexical	Type
	Elemental	Type
	length	Indexical
Return	bool	
Parameters	numerator	const Numeral< Indexical > &
	values	const Endemical (&)[length]

Function template which creates a linked list one node at a time on the heap. It then conducts an increasing directional test on the first node in the list and a decreasing directional test on the last node in the list. It deletes each node in the linked list when done.

Function	main	
Return	int	
Parameters		

Function which first tests a numeration's axial trajection and if successful then tests pointer trajection, segment trajection, array trajection and linked list trajection assigning to their elements from a local sample values array. If the array trajection test was successful it then performs an axial trajection of the sample values array forward then backward, printing each text character element to the console.