

## Genetica Namespace

### Classes

Class	Description
 <a href="#">Population(TProgram, TOutput)</a>	<p>Represents a simple implementation of <a href="#">IPopulation(TProgram)</a> composed of <a href="#">ITreeProgram(TOutput)</a>. The algorithm follows a traditional evolutionary algorithm to step one generation:</p> <ol style="list-style-type: none"> <li>1. performs selection to get a pool of <math>n</math> parents for crossover, where <math>n</math> is the size of the population;</li> <li>2. performs crossover from parent pool to get some offspring (given percentage of the population);</li> <li>3. performs mutation from parent pool to get some mutated programs (given percentage of the population);</li> <li>4. performs elite selection (keeps a given percentage of the population corresponding to the best programs);</li> <li>5. creates some random programs (given percentage of the population).</li> </ol>

### Interfaces

Interface	Description
 <a href="#">IPopulation(TProgram)</a>	Represents a set of <a href="#">IProgram</a> to be evolved through GP according to the defined evolutionary operators.
 <a href="#">ITreeNode</a>	Represents an interface for three nodes, i.e., nodes that have some descendant nodes associated.

## IPopulation(TProgram) Interface

Represents a set of [IProgram](#) to be evolved through GP according to the defined evolutionary operators.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public interface IPopulation<TProgram> : ISet<TProgram>,
    ICollection<TProgram>, IEnumerable<TProgram>, IEnumerable, IDisposable
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

**TProgram**

The type of program.

The IPopulation(TProgram) type exposes the following members.

### Properties

	Name	Description
	<a href="#">BestProgram</a>	Gets the TProgram in the population attaining the maximal fitness across all programs.
	<a href="#">Count</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
	<a href="#">CrossoverPercent</a>	Gets or sets the percentage of a population used for the crossover operation during GP.
	<a href="#">ElitismPercent</a>	Gets or sets the percentage of a population selected to be copied to the next generation during GP.
	<a href="#">IsReadOnly</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
	<a href="#">MutationPercent</a>	Gets or sets the percentage of a population used for the mutation operation during GP.

### Methods

	Name	Description
	<a href="#">Add</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
	<a href="#">Clear</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
	<a href="#">Contains</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
	<a href="#">CopyTo</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)

=	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
=	<a href="#">ExceptWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">GetEnumerator</a>	(Inherited from <a href="#">IEnumerable(TProgram)</a> .)
=	<a href="#">Init</a>	Initializes this population with the given seed programs.
=	<a href="#">IntersectWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">IsProperSubsetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">IsProperSupersetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">IsSubsetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">IsSupersetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">Overlaps</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">Remove</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
=	<a href="#">SetEquals</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">Step</a>	Performs one step of GP across the whole population, i.e., it generates a new generation by applying the evolutionary operators.
=	<a href="#">SymmetricExceptWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
=	<a href="#">UnionWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)

## Extension Methods

	Name	Description
=	<a href="#">AddRange(TProgram)</a>	Adds the given values to the given set. (Defined by <a href="#">CollectionUtil</a> .)

## See Also

[Genetica Namespace](#)

## IPopulation(TProgram).IPopulation(TProgram) Properties

The [IPopulation\(TProgram\)](#) generic type exposes the following members.

### Properties

Name	Description
 <a href="#">BestProgram</a>	Gets the <i>TProgram</i> in the population attaining the maximal fitness across all programs.
 <a href="#">Count</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
 <a href="#">CrossoverPercent</a>	Gets or sets the percentage of a population used for the crossover operation during GP.
 <a href="#">ElitismPercent</a>	Gets or sets the percentage of a population selected to be copied to the next generation during GP.
 <a href="#">IsReadOnly</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
 <a href="#">MutationPercent</a>	Gets or sets the percentage of a population used for the mutation operation during GP.

### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## IPopulation(TProgram).BestProgram Property

Gets the *TProgram* in the population attaining the maximal fitness across all programs.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
TProgram BestProgram { get; }
```

[View Source](#)

### Property Value

Type: *TProgram*

### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## IPopulation(TProgram).CrossoverPercent Property

Gets or sets the percentage of a population used for the crossover operation during GP.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
double CrossoverPercent { get; set; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## IPopulation(TProgram).ElitismPercent Property

Gets or sets the percentage of a population selected to be copied to the next generation during GP.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
double ElitismPercent { get; set; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## IPopulation(TProgram).MutationPercent Property

Gets or sets the percentage of a population used for the mutation operation during GP.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
double MutationPercent { get; set; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## IPopulation(TProgram).IPopulation(TProgram) Methods

The [IPopulation\(TProgram\)](#) generic type exposes the following members.

### Methods

Name	Description
<a href="#">Add</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">Clear</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
<a href="#">Contains</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
<a href="#">CopyTo</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
<a href="#">ExceptWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">GetEnumerator</a>	(Inherited from <a href="#">IEnumerable(TProgram)</a> .)
<a href="#">Init</a>	Initializes this population with the given seed programs.
<a href="#">IntersectWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">IsProperSubsetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">IsProperSupersetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">IsSubsetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">IsSupersetOf</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">Overlaps</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">Remove</a>	(Inherited from <a href="#">ICollection(TProgram)</a> .)
<a href="#">SetEquals</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">Step</a>	Performs one step of GP across the whole population, i.e., it generates a new generation by applying the evolutionary operators.
<a href="#">SymmetricExceptWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)
<a href="#">UnionWith</a>	(Inherited from <a href="#">ISet(TProgram)</a> .)

### Extension Methods

Name	Description
<a href="#">AddRange(TProgram)</a>	Adds the given values to the given set. (Defined by <a href="#">CollectionUtil</a> .)

### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## IPopulation(TProgram).Init Method

Initializes this population with the given seed programs.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
void Init(  
    ISet<TProgram> seeds  
)
```

[View Source](#)

#### Parameters

*seeds*

Type: [System.Collections.Generic.ISet\(TProgram\)](#)

The seed programs used to initialize this population.

#### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## IPopulation(TProgram).Step Method

Performs one step of GP across the whole population, i.e., it generates a new generation by applying the evolutionary operators.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
void Step()
```

[View Source](#)

### See Also

[IPopulation\(TProgram\)Interface](#)

[Genetica Namespace](#)

## ITreeNode Interface

Represents an interface for three nodes, i.e., nodes that have some descendant nodes associated.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface ITreeNode
```

[View Source](#)

The **ITreeNode** type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node.

### Extension Methods

	Name	Description
	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <b>ITreeNode</b> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <b>ITreeNode</b> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <b>ITreeNode</b> to an image file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <b>ITreeNode</b> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[Genetica Namespace](#)

## ITreeNode.ITreeNode Properties

The [ITreeNode](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node.

### See Also

[ITreeNode Interface](#)

[Genetica Namespace](#)

## ITreeNode.Children Property

Gets the list of child nodes associated with this node.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
IReadOnlyList<ITreeNode> Children { get; }
```

[View Source](#)

*Property Value*

Type: [IReadOnlyList\(ITreeNode\)](#)

### See Also

[ITreeNode Interface](#)

[Genetica Namespace](#)

## ITreeNode.ITableView Methods

The [ITreeNode](#) type exposes the following members.

### Extension Methods

Name	Description
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[ITreeNode Interface](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput) Class

Represents a simple implementation of [IPopulation\(TProgram\)](#) composed of [ITreeProgram\(TOutput\)](#). The algorithm follows a traditional evolutionary algorithm to step one generation:

1. performs selection to get a pool of n parents for crossover, where n is the size of the population;
2. performs crossover from parent pool to get some offspring (given percentage of the population);
3. performs mutation from parent pool to get some mutated programs (given percentage of the population);
4. performs elite selection (keeps a given percentage of the population corresponding to the best programs);
5. creates some random programs (given percentage of the population).

### Inheritance Hierarchy

[System.Object](#)

[System.Collections.Generic.HashSet\(TProgram\)](#)

Genetica.Population(TProgram, TOutput)

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class Population<TProgram, TOutput> : HashSet<TProgram>,
    IPopulation<TProgram>, ISet<TProgram>, ICollection<TProgram>,
    IEnumerable<TProgram>, IEnumerable, IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

TOutput

The Population(TProgram, TOutput) type exposes the following members.

## Constructors

	Name	Description
	<a href="#">Population(TProgram, TOutput)</a>	Creates a new Population(TProgram, TOutput) with the given arguments.

## Properties

	Name	Description
	<a href="#">BestProgram</a>	Gets the TProgram in the population attaining the maximal fitness across all programs.
	<a href="#">Comparer</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">Count</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">CrossoverPercent</a>	Gets or sets the percentage of a population used for the crossover operation during GP.
	<a href="#">ElitismPercent</a>	Gets or sets the percentage of a population selected to be copied to the next generation during GP.
	<a href="#">MutationPercent</a>	Gets or sets the percentage of a population used for the mutation operation during GP.

## Methods

	Name	Description
	<a href="#">Add</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">Clear</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">Contains</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">CopyTo(T[])</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">CopyTo(T[], Int32)</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">CopyTo(T[], Int32, Int32)</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">Dispose</a>	Releases all resources used by the Population(TProgram, TOutput)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ExceptWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetEnumerator</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetObjectData</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Init</a>	Initializes this population with the given seed programs.
	<a href="#">IntersectWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)

 <a href="#">IsProperSubsetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">IsProperSupersetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">IsSubsetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">IsSupersetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">OnDeserialization</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">Overlaps</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">Remove</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">RemoveWhere</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">SetEquals</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">Step</a>	Performs one step of GP across the whole population, i.e., it generates a new generation by applying the evolutionary operators.
 <a href="#">SymmetricExceptWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">TrimExcess</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">UnionWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)

## Extension Methods

	Name	Description
 <a href="#">AddRange(TProgram)</a>		Adds the given values to the given set. (Defined by <a href="#">CollectionUtil</a> .)

## See Also

[Genetica Namespace](#)

## Population(TProgram, TOutput) Constructor

Creates a new [Population\(TProgram, TOutput\)](#) with the given arguments.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public Population(  
    uint maxSize,  
    PrimitiveSet<TProgram> primitives,  
    IProgramGenerator<TProgram, TOutput> programGenerator,  
    IComparer<TProgram> programComparer,  
    ISelectionOperator<TProgram> selectionOperator,  
    ICrossoverOperator<TProgram> crossoverOperator,  
    IMutationOperator<TProgram> mutationOperator,  
    uint maxGenerationDepth = 4,  
    uint maxElementLength = 20,  
    double crossoverPercent = 0.65,  
    double mutationPercent = 0.2,  
    double elitismPercent = 0.1  
)
```

[View Source](#)

### Parameters

*maxSize*

Type: [System.UInt32](#)

The maximum size of the population.

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The primitive set used to generate new programs.

*programGenerator*

Type: [Genetica.Operators.Generation.IProgramGenerator\(TProgram, TOutput\)](#)

The generator of new programs.

*programComparer*

Type: [System.Collections.Generic.IComparer\(TProgram\)](#)

The function used to compare programs and select the best program.

*selectionOperator*

Type: [Genetica.Operators.Selection.ISelectionOperator\(TProgram\)](#)

The operator to perform selection.

*crossoverOperator*

Type: [Genetica.Operators.Crossover.ICrossoverOperator\(TProgram\)](#)

The operator to crossover programs.

*mutationOperator*

Type: [Genetica.Operators.Mutation.IMutationOperator\(TProgram\)](#)

The operator to mutate programs.

*maxGenerationDepth* (Optional)

Type: [System.UInt32](#)

The maximum depth of elements generated during GP.

*maxLength* (Optional)

Type: [System.UInt32](#)

The maximum length of elements generated during GP.

*crossoverPercent* (Optional)

Type: [System.Double](#)

The percentage of a population used for the crossover operator during GP.

*mutationPercent* (Optional)

Type: [System.Double](#)

The percentage of a population used for the mutation operator during GP.

*elitismPercent* (Optional)

Type: [System.Double](#)

The percentage of a population used for elite selection during GP.

## See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).Population(TProgram, TOutput) Properties

The [Population\(TProgram, TOutput\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">BestProgram</a>	Gets the <i>TProgram</i> in the population attaining the maximal fitness across all programs.
	<a href="#">Comparer</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">Count</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
	<a href="#">CrossoverPercent</a>	Gets or sets the percentage of a population used for the crossover operation during GP.
	<a href="#">ElitismPercent</a>	Gets or sets the percentage of a population selected to be copied to the next generation during GP.
	<a href="#">MutationPercent</a>	Gets or sets the percentage of a population used for the mutation operation during GP.

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).BestProgram Property

Gets the *TProgram* in the population attaining the maximal fitness across all programs.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram BestProgram { get; }
```

[View Source](#)

### Property Value

Type: *TProgram*

Implements

[IPopulation\(TProgram\).BestProgram](#)

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).CrossoverPercent Property

Gets or sets the percentage of a population used for the crossover operation during GP.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double CrossoverPercent { get; set; }
```

[View Source](#)

**Property Value**

Type: [Double](#)

*Implements*

[IPopulation\(TProgram\).CrossoverPercent](#)

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).ElitismPercent Property

Gets or sets the percentage of a population selected to be copied to the next generation during GP.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double ElitismPercent { get; set; }
```

[View Source](#)

**Property Value**

Type: [Double](#)

*Implements*

[IPopulation\(TProgram\).ElitismPercent](#)

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).MutationPercent Property

Gets or sets the percentage of a population used for the mutation operation during GP.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double MutationPercent { get; set; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

*Implements*

[IPopulation\(TProgram\).MutationPercent](#)

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).Population(TProgram, TOutput) Methods

The [Population\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">Add</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">Clear</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">Contains</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">CopyTo(T[])</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">CopyTo(T[], Int32)</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">CopyTo(T[], Int32, Int32)</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">Dispose</a>	
≡	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">ExceptWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetEnumerator</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetObjectData</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Init</a>	Initializes this population with the given seed programs.
≡	<a href="#">IntersectWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">IsProperSubsetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">IsProperSupersetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">IsSubsetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">IsSupersetOf</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">OnDeserialization</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">Overlaps</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">Remove</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">RemoveWhere</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">SetEquals</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
≡	<a href="#">Step</a>	Performs one step of GP across the whole population, i.e., it generates a new generation by applying the evolutionary operators.
≡	<a href="#">SymmetricExceptWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)

 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">TrimExcess</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)
 <a href="#">UnionWith</a>	(Inherited from <a href="#">HashSet(TProgram)</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
 <a href="#">AddRange(TProgram)</a>		Adds the given values to the given set. (Defined by <a href="#">CollectionUtil</a> .)

## See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).Dispose Method

Releases all resources used by the [Population\(TProgram, TOutput\)](#)

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).Init Method

Initializes this population with the given seed programs.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Init(  
    ISet<TProgram> seeds  
)
```

[View Source](#)

#### Parameters

*seeds*

Type: [System.Collections.Generic.ISet\(TProgram\)](#)

The seed programs used to initialize this population.

#### Implements

[IPopulation\(TProgram\).Init\(ISet\(TProgram\)\)](#)

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Population(TProgram, TOutput).Step Method

Performs one step of GP across the whole population, i.e., it generates a new generation by applying the evolutionary operators.

**Namespace:** [Genetica](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public virtual void Step()
```

[View Source](#)

Implements

[IPopulation\(TProgram\).Step\(\)](#)

### See Also

[Population\(TProgram, TOutput\)Class](#)

[Genetica Namespace](#)

## Genetica.D3 Namespace

### Classes

	<b>Class</b>	<b>Description</b>
	<a href="#">Extensions</a>	Contains several extension methods to allow printing <a href="#">ITreeProgram(TOutput)</a> and <a href="#">IInformationTree(TProgram)</a> to d3.js files.
	<a href="#">UniqueItemsList(T)</a>	

## Extensions Class

Contains several extension methods to allow printing [ITreeProgram\(TOutput\)](#) and [IInformationTree\(TProgram\)](#) to d3.js files.

### Inheritance Hierarchy

[System.Object](#)

Genetica.D3.Extensions

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public static class Extensions
```

[View Source](#)

The **Extensions** type exposes the following members.

### Methods

	Name	Description
 	<a href="#">ToD3JsonFile(ITreeNode, String, Formatting)</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file.
 	<a href="#">ToD3JsonFile(TProgram)(IInformationTree(TProgram), String, Formatting)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file.
 	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout.

### See Also

[Genetica.D3 Namespace](#)

## Extensions.Extensions Methods

The [Extensions](#) type exposes the following members.

### Methods

	<b>Name</b>	<b>Description</b>
 	<a href="#">ToD3JsonFile(ITreeNode, String, Formatting)</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file.
 	<a href="#">ToD3JsonFile(TProgram)(IInformationTree(TProgram), String, Formatting)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file.
 	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout.

### See Also

[Extensions Class](#)

[Genetica.D3 Namespace](#)

## Extensions.ToD3JsonFile Method

### Overload List

Name	Description
  <a href="#">ToD3JsonFile(TProgram)(IInformationTree(TProgram), String, Formatting)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file.
  <a href="#">ToD3JsonFile(ITreeNode, String, Formatting)</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file.

### See Also

[Extensions Class](#)

[Genetica.D3 Namespace](#)

## Extensions.ToD3JsonFile(TProgram) Method (IInformationTree(TProgram), String, Formatting)

Saves the given [IInformationTree\(TProgram\)](#) to a d3.js tree file.

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public static void ToD3JsonFile<TProgram>(
    this IInformationTree<TProgram> tree,
    string filePath,
    Formatting formatting = Formatting.None
)
where TProgram : ITreeProgram
```

[View Source](#)

### Parameters

*tree*

Type: [Genetica.Trees.IInformationTree\(TProgram\)](#)

The information tree to be saved.

*filePath*

Type: [System.String](#)

The path of the file in which to save the given information tree.

*formatting* (Optional)

Type: [Formatting](#)

The formatting to be used to write to the json file.

### Type Parameters

*TProgram*

The type of program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IInformationTree\(TProgram\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[Extensions Class](#)

[ToD3JsonFile Overload](#)

[Genetica.D3 Namespace](#)



## Extensions.ToD3JsonFile Method (ITreeNode, String, Formatting)

Saves the tree of the given [ITreeNode](#) to a d3.js tree file.

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

**C#**

```
public static void ToD3JsonFile(  
    this ITreeNode rootNode,  
    string filePath,  
    Formatting formatting = Formatting.None  
)
```

[View Source](#)

### Parameters

*rootNode*

Type: [Genetica.ITreeNode](#)

The root node of the tree to be saved.

*filePath*

Type: [System.String](#)

The path of the file in which to save the given tree.

*formatting* (Optional)

Type: [Formatting](#)

The formatting to be used to write to the json file.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeNode](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[Extensions Class](#)

[ToD3JsonFile Overload](#)

[Genetica.D3 Namespace](#)

## Extensions.ToD3TreeJsonFile Method

Saves the tree of the given [ITreeNode](#) to a d3.js tree file using a small tree layout.

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

**C#**

```
public static void ToD3TreeJsonFile(  
    this ITreeNode rootNode,  
    string filePath,  
    Formatting formatting = Formatting.None  
)
```

[View Source](#)

### Parameters

*rootNode*

Type: [Genetica.ITreeNode](#)

The root node of the tree to be saved.

*filePath*

Type: [System.String](#)

The path of the file in which to save the given tree.

*formatting* (Optional)

Type: [Formatting](#)

The formatting to be used to write to the json file.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeNode](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[Extensions Class](#)

[Genetica.D3 Namespace](#)

## UniqueItemsList(T) Class

[Missing <summary> documentation for "T:Genetica.D3.UniqueItemsList`1"]

### Inheritance Hierarchy

[System.Object](#)

Genetica.D3.UniqueItemsList(T)

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public class UniqueItemsList<T> : IDisposable  
where T : class
```

[View Source](#)

### Type Parameters

T

[Missing <typeparam name="T"/> documentation for "T:Genetica.D3.UniqueItemsList`1"]

The UniqueItemsList(T) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">UniqueItemsList(T)</a>	Initializes a new instance of the UniqueItemsList(T) class

### Properties

	Name	Description
	<a href="#">Count</a>	

### Methods

	Name	Description
	<a href="#">Clear</a>	
	<a href="#">Dispose</a>	Releases all resources used by the UniqueItemsList(T)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetIndex</a>	

 <a href="#">GetItem</a>	
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.D3 Namespace](#)

## UniqueItemsList(T) Constructor

Initializes a new instance of the [UniqueItemsList\(T\)](#) class

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public UniqueItemsList(  
    int capacity = -1  
)
```

[View Source](#)

#### Parameters

capacity (Optional)

Type: [System.Int32](#)

[Missing <param name="capacity"/> documentation for  
"M:Genetica.D3.UniqueItemsList`1.#ctor(System.Int32)"]

### See Also

[UniqueItemsList\(T\)Class](#)

[Genetica.D3 Namespace](#)

## UniqueItemsList(*T*).UniqueItemsList(*T*) Properties

The [UniqueItemsList\(\*T\*\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">Count</a>	

### See Also

[UniqueItemsList\(\*T\*\)Class](#)

[Genetica.D3 Namespace](#)

## UniqueItemsList(T).Count Property

[Missing <summary> documentation for "P:Genetica.D3.UniqueItemsList`1.Count"]

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public int Count { get; }
```

[View Source](#)

**Property Value**

Type: [Int32](#)

### See Also

[UniqueItemsList\(T\)Class](#)

[Genetica.D3 Namespace](#)

## UniqueItemsList(T).UniqueItemsList(T) Methods

The [UniqueItemsList\(T\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Clear</a>	
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetIndex</a>	
	<a href="#">GetItem</a>	
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[UniqueItemsList\(T\)Class](#)

[Genetica.D3 Namespace](#)

## UniqueItemsList(T).Clear Method

[Missing <summary> documentation for "M:Genetica.D3.UniqueItemsList`1.Clear"]

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public void Clear()
```

[View Source](#)

### See Also

[UniqueItemsList\(T\)Class](#)

[Genetica.D3 Namespace](#)

## [UniqueItemsList\(T\).Dispose Method](#)

Releases all resources used by the [UniqueItemsList\(T\)](#)

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[UniqueItemsList\(T\)Class](#)

[Genetica.D3 Namespace](#)

## UniqueItemsList(T).GetIndex Method

[Missing <summary> documentation for "M:Genetica.D3.UniqueItemsList`1.GetIndex(`o)"]

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public uint GetIndex(  
    T item  
)
```

[View Source](#)

#### Parameters

item

Type: T

[Missing <param name="item"/> documentation for  
"M:Genetica.D3.UniqueItemsList`1.GetIndex(`o)"]

#### Return Value

Type: [UInt32](#)

[Missing <returns> documentation for "M:Genetica.D3.UniqueItemsList`1.GetIndex(`o)"]

#### See Also

[UniqueItemsList\(T\)Class](#)

[Genetica.D3 Namespace](#)

## UniqueItemsList(T).GetItem Method

[Missing <summary> documentation for  
"M:Genetica.D3.UniqueItemsList`1.GetItem(System.UInt32)"]

**Namespace:** [Genetica.D3](#)

**Assembly:** Genetica.D3 (in Genetica.D3.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public T GetItem(  
    uint idx  
)
```

[View Source](#)

### Parameters

idx

Type: [System.UInt32](#)

[Missing <param name="idx"/> documentation for  
"M:Genetica.D3.UniqueItemsList`1.GetItem(System.UInt32)"]

### Return Value

Type: T

[Missing <returns> documentation for "M:Genetica.D3.UniqueItemsList`1.GetItem(System.UInt32)"]

### See Also

[UniqueItemsList\(T\)Class](#)

[Genetica.D3 Namespace](#)

## Genetica.Elements Namespace

### Classes

Class	Description
 <a href="#">EvolutionaryDistanceCalculator(TProgram, TOutput)</a>	Represents a class for calculating the evolutionary distance between two given <a href="#">ITreeProgram(TOutput)</a> s, i.e., the number of evolutionary operations needed to transform one program into the other. This corresponds to a rather exhaustive search over some given <a href="#">IMutationOperator(TProgram)</a> and <a href="#">ICrossoverOperator(TProgram)</a> . The search is guided by some given <a href="#">ISimilarityMeasure(TProgram)</a> that serves as an heuristic for the transformation process.
 <a href="#">MathExpressionConverter</a>	Represents a class include a set of methods to convert <a href="#">MathProgram</a> objects to and from <a href="#">String</a> expressions in different notations.
 <a href="#">MathPrimitiveSets</a>	Contains different <a href="#">PrimitiveSet(TProgram)</a> for <a href="#">MathProgram</a> .
 <a href="#">MathProgram</a>	Represents a base class for mathematical program programs. A <a href="#">MathProgram</a> represents a mathematical expression whose output is a double-precision value. Math programs are useful to perform symbolic regression.
 <a href="#">MathProgramExtensions</a>	Declares a set of extension methods for <a href="#">MathProgram</a> .
 <a href="#">PrimitiveSet(TProgram)</a>	Represents a collection of <i>TProgram</i> primitives including a set of functions and terminals.
 <a href="#">TreeProgramExtensions</a>	Declares a set of extension methods for <a href="#">ITreeProgram(TOutput)</a> objects.

### Interfaces

Interface	Description
 <a href="#">ICommutativeTreeProgram(TOutput)</a>	Represents a specialization of <a href="#">ITreeProgram(TOutput)</a> where the order of the descendant sub-programs is not important, i.e., the program has the commutative property..
 <a href="#">IExpressionConverter(TProgram)</a>	Represents an interface for classes converting from <a href="#">IProgram</a> objects to <a href="#">String</a> and vice-versa.
 <a href="#">IProgram</a>	Represents an interface for a genetic program.

» <a href="#">IProgram(TInput, TOutput)</a>	Represents an interface for a genetic program that has some input of type <i>TInput</i> and output of type <i>TOutput</i> .
» <a href="#">ITreeExpressionConverter(TProgram)</a>	Represents an interface for classes converting from <a href="#">ITreeProgram</a> objects to <a href="#">String</a> and vice-versa both in normal and prefix notation. In normal-form notation, nodes are written in the form node-type(arg1 arg2 ...) or (arg1 node-type arg2). In prefix notation, nodes are written in the form (node-type arg1 arg2 ...).
» <a href="#">ITreeProgram</a>	Represents an interface for <a href="#">IProgram</a> objects that are represented by a hierarchical tree.
» <a href="#">ITreeProgram(TOutput)</a>	Represents a <a href="#">IProgram(TInput, TOutput)</a> that is represented as a hierarchical syntactic tree, in which the descendants (inputs) can be any kind of <a href="#">ITreeProgram(TOutput)</a> .

## EvolutionaryDistanceCalculator(TProgram, TOutput) Class

Represents a class for calculating the evolutionary distance between two given [ITreeProgram\(TOutput\)](#)s, i.e., the number of evolutionary operations needed to transform one program into the other. This corresponds to a rather exhaustive search over some given [IMutationOperator\(TProgram\)](#) and [ICrossoverOperator\(TProgram\)](#). The search is guided by some given [ISimilarityMeasure\(TProgram\)](#) that serves as an heuristic for the transformation process.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Elements.EvolutionaryDistanceCalculator(TProgram, TOutput)

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class EvolutionaryDistanceCalculator<TProgram, TOutput>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The EvolutionaryDistanceCalculator(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">EvolutionaryDistanceCalculator(TProgram, TOutput)</a>	Creates a new EvolutionaryDistanceCalculator(TProgram, TOutput) with the given arguments.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">GetDistance</a>	Calculates the evolutionary distance between the two given programs. The procedure stops whenever the similarity between candidate programs cannot be improved by using any of the crossover and mutation operators available.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Elements Namespace](#)

## EvolutionaryDistanceCalculator(TProgram, TOutput) Constructor

Creates a new [EvolutionaryDistanceCalculator\(TProgram, TOutput\)](#) with the given arguments.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public EvolutionaryDistanceCalculator(  
    ISimilarityMeasure<TProgram> similarityMeasure,  
    PrimitiveSet<TProgram> primitiveSet,  
    IEnumerable<ICrossoverOperator<TProgram>> crossovers,  
    IEnumerable<IMutationOperator<TProgram>> mutations  
)
```

[View Source](#)

#### Parameters

*similarityMeasure*

Type: [Genetica.Similarity.ISimilarityMeasure\(TProgram\)](#)

The similarity measure to guide the transformation search process.

*primitiveSet*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The primitives used for the crossover operator.

*crossovers*

Type: [System.Collections.Generic.IEnumerable\(ICrossoverOperator\(TProgram\)\)](#)

The crossover operators to generate programs during the search.

*mutations*

Type: [System.Collections.Generic.IEnumerable\(IMutationOperator\(TProgram\)\)](#)

The mutation operators to generate programs during the search.

### See Also

[EvolutionaryDistanceCalculator\(TProgram, TOutput\)Class](#)

[Genetica.Elements Namespace](#)

## EvolutionaryDistanceCalculator(TProgram, TOutput).EvolutionaryDistanceCalculator(TProgram, TOutput) Methods

The [EvolutionaryDistanceCalculator\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetDistance</a>	Calculates the evolutionary distance between the two given programs. The procedure stops whenever the similarity between candidate programs cannot be improved by using any of the crossover and mutation operators available.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[EvolutionaryDistanceCalculator\(TProgram, TOutput\)Class](#)

[Genetica.Elements Namespace](#)

## EvolutionaryDistanceCalculator(TProgram, TOutput).GetDistance Method

Calculates the evolutionary distance between the two given programs. The procedure stops whenever the similarity between candidate programs cannot be improved by using any of the crossover and mutation operators available.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IList<KeyValuePair<TProgram, Type>> GetDistance(
    TProgram prog1,
    TProgram prog2
)
```

[View Source](#)

### Parameters

*prog1*

Type: *TProgram*

The source program to be transformed into the second program.

*prog2*

Type: *TProgram*

The target program that we want to achieve.

### Return Value

Type: [IList\(KeyValuePair\(TProgram, Type\)\)](#)

A list containing pairs of candidate program - type of operator used.

### See Also

[EvolutionaryDistanceCalculator\(TProgram, TOutput\)Class](#)

[Genetica.Elements Namespace](#)

## ICommutativeTreeProgram(TOutput) Interface

Represents a specialization of [ITreeProgram\(TOutput\)](#) where the order of the descendant sub-programs is not important, i.e., the program has the commutative property..

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public interface ICommutativeTreeProgram<TOutput> : ITreeProgram<TOutput>,
    IProgram< IReadOnlyList<ITreeProgram<TOutput>>, TOutput>, IProgram,
    ITreeProgram, ITreeNode, IComparable<ITreeProgram<TOutput>>
```

[View Source](#)

#### Type Parameters

##### TOutput

The type of output.

The ICommutativeTreeProgram(TOutput) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Input</a>	Gets the input of this program. (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">ITreeProgram</a> .)
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### Methods

	Name	Description
	<a href="#">CompareTo</a>	(Inherited from <a href="#">IComparable(ITreeProgram(TOutput))</a> .)
	<a href="#">Compute</a>	Computes the output of the program based on it's <a href="#">Input</a> . (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
	<a href="#">CreateNew</a>	Creates a new <a href="#">ITreeProgram(TOutput)</a> of the same kind of this program with the given child programs. (Inherited from <a href="#">ITreeProgram(TOutput)</a> .)
	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">ITreeProgram(TOutput)</a> derived type. (Inherited from <a href="#">ITreeProgram(TOutput)</a> .)

 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">ITreeProgram(TOutput)</a> .)
--	--

## Extension Methods

Name	Description
 <a href="#">ContainsSubElement(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetCommonRegionIndexes(TOutput)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetLeaves(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxBreadth(TOutput)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxDepth(TOutput)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetPrimitives(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetSubCombinations(TOutput)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of

		the children and also the sub-combinations of each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetSubPrograms(TOutput)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">IsLeaf(TOutput)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">IsSubProgramOf(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">ProgramAt(TOutput)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">Replace(TOutput)(UInt32, ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">Replace(TOutput)(ITreeProgram(TOutput), ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i> . (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
💡	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
💡	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
💡	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

**See Also**

[Genetica.Elements Namespace](#)

## ICommutativeTreeProgram(TOutput).ICommutativeTreeProgram(TOutput) Properties

The [ICommutativeTreeProgram\(TOutput\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Input</a>	Gets the input of this program. (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">ITreeProgram</a> .)
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### See Also

[ICommutativeTreeProgram\(TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## ICommutativeTreeProgram(TOutput).ICommutativeTreeProgram(TOutput) Methods

The [ICommutativeTreeProgram\(TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">CompareTo</a>	(Inherited from <a href="#">IComparable(ITreeProgram(TOutput))</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on it's <a href="#">Input</a> . (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">ITreeProgram(TOutput)</a> of the same kind of this program with the given child programs. (Inherited from <a href="#">ITreeProgram(TOutput)</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">ITreeProgram(TOutput)</a> derived type. (Inherited from <a href="#">ITreeProgram(TOutput)</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">ITreeProgram(TOutput)</a> .)

### Extension Methods

	Name	Description
≡	<a href="#">ContainsSubElement(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
≡	<a href="#">GetCommonRegionIndexes(TOutput)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
≡	<a href="#">GetLeaves(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
≡	<a href="#">GetMaxBreadth(TOutput)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this

		program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">GetMaxDepth(TOutput)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">GetPrimitives(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">GetSubCombinations(TOutput)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">GetSubPrograms(TOutput)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">IsLeaf(TOutput)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">IsSubProgramOf(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">ProgramAt(TOutput)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">Replace(TOutput)(UInt32, ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">Replace(TOutput)(ITreeProgram(TOutput), ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i> . (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[ICommutativeTreeProgram\(TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## IExpressionConverter(TProgram) Interface

Represents an interface for classes converting from [IProgram](#) objects to [String](#) and vice-versa.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IExpressionConverter<TProgram>
where TProgram : IProgram
```

[View Source](#)

#### Type Parameters

*TProgram*

The type of program.

The IExpressionConverter(TProgram) type exposes the following members.

### Methods

	Name	Description
	<a href="#">FromString</a>	Converts the given <a href="#">String</a> to a program of type <i>TProgram</i> .
	<a href="#">ToString</a>	Converts the given <i>TProgram</i> to a string.

### See Also

[Genetica.Elements Namespace](#)

## IExpressionConverter(TProgram).IExpressionConverter(TProgram)

### Methods

The [IExpressionConverter\(TProgram\)](#) generic type exposes the following members.

#### Methods

	Name	Description
	<a href="#">FromString</a>	Converts the given <a href="#">String</a> to a program of type <a href="#">TProgram</a> .
	<a href="#">ToString</a>	Converts the given <a href="#">TProgram</a> to a string.

### See Also

[IExpressionConverter\(TProgram\)Interface](#)

[Genetica.Elements Namespace](#)

## IExpressionConverter(TProgram).FromString Method

Converts the given [String](#) to a program of type TProgram.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
TProgram FromString(  
    string programStr  
)
```

[View Source](#)

#### Parameters

*programStr*

Type: [System.String](#)

The string representing some program.

#### Return Value

Type: [TProgram](#)

The program corresponding to the given string representation.

### See Also

[IExpressionConverter\(TProgram\)Interface](#)

[Genetica.Elements Namespace](#)

## IExpressionConverter(TProgram).ToString Method

Converts the given *TProgram* to a string.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
string ToString(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to convert.

#### Return Value

Type: [String](#)

The string representing the given program.

### See Also

[IExpressionConverter\(TProgram\) Interface](#)

[Genetica.Elements Namespace](#)

## IProgram Interface

Represents an interface for a genetic program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IProgram
```

[View Source](#)

The **IProgram** type exposes the following members.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets a string representing this program's expression.
	<a href="#">Length</a>	Gets the program's length.

### See Also

[Genetica.Elements Namespace](#)

## IProgram.IProgram Properties

The [IProgram](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets a string representing this program's expression.
	<a href="#">Length</a>	Gets the program's length.

### See Also

[IProgram Interface](#)

[Genetica.Elements Namespace](#)

## IProgram.Expression Property

Gets a string representing this program's expression.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
string Expression { get; }
```

[View Source](#)

*Property Value*

Type: [String](#)

### See Also

[IProgram Interface](#)

[Genetica.Elements Namespace](#)

## IProgram.Length Property

Gets the program's length.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
ushort Length { get; }
```

[View Source](#)

*Property Value*

Type: [UInt16](#)

### See Also

[IProgram Interface](#)

[Genetica.Elements Namespace](#)

## IProgram(TInput, TOutput) Interface

Represents an interface for a genetic program that has some input of type *TInput* and output of type *TOutput*.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IProgram<out TInput, out TOutput> : IProgram
```

[View Source](#)

#### Type Parameters

*TInput*

The type of program input.

*TOutput*

The type of program output.

The IProgram(*TInput*, *TOutput*) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Input</a>	Gets the input of this program.
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### Methods

	Name	Description
	<a href="#">Compute</a>	Computes the output of the program based on it's <a href="#">Input</a> .

### See Also

[Genetica.Elements Namespace](#)

## IProgram(TInput, TOutput).IProgram(TInput, TOutput) Properties

The [IProgram\(TInput, TOutput\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Input</a>	Gets the input of this program.
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### See Also

[IProgram\(TInput, TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## IProgram<TInput, TOutput>.Input Property

Gets the input of this program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
TInput Input { get; }
```

[View Source](#)

### Property Value

Type: *TInput*

### See Also

[IProgram<TInput, TOutput> Interface](#)

[Genetica.Elements Namespace](#)

## IProgram(TInput, TOutput).IProgram(TInput, TOutput) Methods

The [IProgram\(TInput, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Compute</a>	Computes the output of the program based on it's <a href="#">Input</a> .

### See Also

[IProgram\(TInput, TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## IProgram<TInput, TOutput>.Compute Method

Computes the output of the program based on it's [Input](#).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
TOutput Compute()
```

[View Source](#)

### Return Value

Type: *TOutput*

A *TOutput* value corresponding to this program's computed output.

### See Also

[IProgram<TInput, TOutput>Interface](#)

[Genetica.Elements Namespace](#)

## ITreeExpressionConverter(TProgram) Interface

Represents an interface for classes converting from [ITreeProgram](#) objects to [String](#) and vice-versa both in normal and prefix notation. In normal-form notation, nodes are written in the form node-type(arg1 arg2 ...) or (arg1 node-type arg2). In prefix notation, nodes are written in the form (node-type arg1 arg2 ...).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public interface ITreeExpressionConverter<TProgram> : IExpressionConverter<TProgram>
where TProgram : ITreeProgram
```

[View Source](#)

#### Type Parameters

*TProgram*

The type of program.

The ITreeExpressionConverter(*TProgram*) type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">FromNormalNotation</a>	Converts the given <a href="#">String</a> expression written in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2) to a <i>TProgram</i> .
≡	<a href="#">FromPrefixNotation</a>	Converts the given <a href="#">String</a> expression written in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...) to a <i>TProgram</i> .
≡	<a href="#">FromString</a>	Converts the given <a href="#">String</a> to a program of type <i>TProgram</i> . (Inherited from <a href="#">IExpressionConverter(TProgram)</a> .)
≡	<a href="#">ToNormalNotation</a>	Converts the given <i>TProgram</i> to a <a href="#">String</a> expression in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2).
≡	<a href="#">ToPrefixNotation</a>	Converts the given <i>TProgram</i> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
≡	<a href="#">ToString</a>	Converts the given <i>TProgram</i> to a string. (Inherited from <a href="#">IExpressionConverter(TProgram)</a> .)

### See Also

[Genetica.Elements Namespace](#)

## [ITreeExpressionConverter\(TProgram\).ITreeExpressionConverter\(TProgram\) Methods](#)

The [ITreeExpressionConverter\(TProgram\)](#) generic type exposes the following members.

### Methods

Name	Description
<a href="#">FromNormalNotation</a>	Converts the given <a href="#">String</a> expression written in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2) to a <a href="#">TProgram</a> .
<a href="#">FromPrefixNotation</a>	Converts the given <a href="#">String</a> expression written in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...) to a <a href="#">TProgram</a> .
<a href="#">FromString</a>	Converts the given <a href="#">String</a> to a program of type <a href="#">TProgram</a> . (Inherited from <a href="#">IExpressionConverter(TProgram)</a> .)
<a href="#">ToNormalNotation</a>	Converts the given <a href="#">TProgram</a> to a <a href="#">String</a> expression in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2).
<a href="#">ToPrefixNotation</a>	Converts the given <a href="#">TProgram</a> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
<a href="#">ToString</a>	Converts the given <a href="#">TProgram</a> to a string. (Inherited from <a href="#">IExpressionConverter(TProgram)</a> .)

### See Also

[ITreeExpressionConverter\(TProgram\)Interface](#)

[Genetica.Elements Namespace](#)

## [ITreeExpressionConverter\(TProgram\).FromNormalNotation](#) Method

Converts the given [String](#) expression written in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2) to a *TProgram*.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
TProgram FromNormalNotation(  
    string expression  
)
```

[View Source](#)

#### Parameters

*expression*

Type: [System.String](#)

The expression we want to convert to an program.

#### Return Value

Type: [TProgram](#)

A *TProgram* converted from the given expression.

#### See Also

[ITreeExpressionConverter\(TProgram\)Interface](#)

[Genetica.Elements Namespace](#)

## [ITreeExpressionConverter\(TProgram\).FromPrefixNotation](#) Method

Converts the given [String](#) expression written in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...) to a *TProgram*.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
TProgram FromPrefixNotation(  
    string expression  
)
```

[View Source](#)

#### Parameters

*expression*

Type: [System.String](#)

The expression we want to convert to an program.

#### Return Value

Type: *TProgram*

A *TProgram* converted from the given expression.

### See Also

[ITreeExpressionConverter\(TProgram\)Interface](#)

[Genetica.Elements Namespace](#)

## [ITreeExpressionConverter\(TProgram\).ToNormalNotation](#) Method

Converts the given *TProgram* to a [String](#) expression in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
string ToNormalNotation(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to convert to an expression.

#### Return Value

Type: [String](#)

A [String](#) representing the given program in prefix notation.

#### See Also

[ITreeExpressionConverter\(TProgram\)Interface](#)

[Genetica.Elements Namespace](#)

## [ITreeExpressionConverter\(TProgram\).ToPrefixNotation](#) Method

Converts the given *TProgram* to a [String](#) expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
string ToPrefixNotation(  
    TProgram program,  
    bool includeParentheses = true  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to convert to an expression.

*includeParentheses* (Optional)

Type: [System.Boolean](#)

Whether to write opening '(' and closing ')' parentheses when writing the expression of functions.

#### Return Value

Type: [String](#)

A [String](#) representing the given program in prefix notation.

#### See Also

[ITreeExpressionConverter\(TProgram\)Interface](#)

[Genetica.Elements Namespace](#)

## ITreeProgram Interface

Represents an interface for [IProgram](#) objects that are represented by a hierarchical tree.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface ITreeProgram : IProgram,  
    ITreeNode
```

[View Source](#)

The **ITreeProgram** type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes.
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### Extension Methods

	Name	Description
	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

**See Also**

[Genetica.Elements Namespace](#)

## ITreeProgram.ITreeProgram Properties

The [ITreeProgram](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes.
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### See Also

[ITreeProgram Interface](#)  
[Genetica.Elements Namespace](#)

## ITreeProgram.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
string Label { get; }
```

[View Source](#)

*Property Value*

Type: [String](#)

### See Also

[ITreeProgram Interface](#)

[Genetica.Elements Namespace](#)

## ITreeProgram. ITreeProgram Methods

The [ITreeProgram](#) type exposes the following members.

### Extension Methods

Name	Description
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[ITreeProgram Interface](#)  
[Genetica.Elements Namespace](#)

## ITreeProgram(TOutput) Interface

Represents a [IProgram\(TInput, TOutput\)](#) that is represented as a hierarchical syntactic tree, in which the descendants (inputs) can be any kind of ITreeProgram(TOutput).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface ITreeProgram<TOutput> : IProgram< IReadOnlyList<ITreeProgram<TOutput>>, TOutput>,  
    IProgram, ITreeProgram, ITreeNode, IComparable<ITreeProgram<TOutput>>
```

[View Source](#)

#### Type Parameters

TOutput

The type of output.

The ITreeProgram(TOutput) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Input</a>	Gets the input of this program. (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">ITreeProgram</a> .)
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### Methods

	Name	Description
	<a href="#">CompareTo</a>	(Inherited from <a href="#">IComparable(ITreeProgram(TOutput))</a> .)
	<a href="#">Compute</a>	Computes the output of the program based on it's <a href="#">Input</a> . (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
	<a href="#">CreateNew</a>	Creates a new ITreeProgram(TOutput) of the same kind of this program with the given child programs.
	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding ITreeProgram(TOutput) derived type.
	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant

		operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.
--	--	---

## Extension Methods

Name	Description
 <a href="#">ContainsSubElement(TOutput)</a>	Checks whether a given ITreeProgram(TOutput) contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetCommonRegionIndexes(TOutput)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetLeaves(TOutput)</a>	Gets a dictionary containing all the ITreeProgram(TOutput) leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxBreadth(TOutput)</a>	Gets the maximum breadth of the program, i.e., the number of ITreeProgram(TOutput) leaves encountered starting from this program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxDepth(TOutput)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetPrimitives(TOutput)</a>	Gets a dictionary containing all the ITreeProgram(TOutput) primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetSubCombinations(TOutput)</a>	Gets all the ITreeProgram(TOutput) sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of

		each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetSubPrograms(TOutput)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">IsLeaf(TOutput)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">IsSubProgramOf(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is a program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">ProgramAt(TOutput)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">Replace(TOutput)(UInt32, ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">Replace(TOutput)(ITreeProgram(TOutput), ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i> . (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
⬇️	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
⬇️	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
⬇️	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

**See Also**

[Genetica.Elements Namespace](#)

## **ITreeProgram(TOutput).ITreeProgram(TOutput) Properties**

The [ITreeProgram\(TOutput\)](#) generic type exposes the following members.

### Properties

	<b>Name</b>	<b>Description</b>
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">Expression</a>	Gets a string representing this program's expression. (Inherited from <a href="#">IProgram</a> .)
	<a href="#">Input</a>	Gets the input of this program. (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">ITreeProgram</a> .)
	<a href="#">Length</a>	Gets the program's length. (Inherited from <a href="#">IProgram</a> .)

### See Also

[ITreeProgram\(TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## ITreeProgram(TOutput).ITreeProgram(TOutput) Methods

The [ITreeProgram\(TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">CompareTo</a>	(Inherited from <a href="#">IComparable(ITreeProgram(TOutput))</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on it's <a href="#">Input</a> . (Inherited from <a href="#">IProgram(TInput, TOutput)</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">ITreeProgram(TOutput)</a> of the same kind of this program with the given child programs.
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">ITreeProgram(TOutput)</a> derived type.
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

### Extension Methods

	Name	Description
≡	<a href="#">ContainsSubElement(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
≡	<a href="#">GetCommonRegionIndexes(TOutput)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
≡	<a href="#">GetLeaves(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
≡	<a href="#">GetMaxBreadth(TOutput)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program. (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">GetMaxDepth(TOutput)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetPrimitives(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetSubCombinations(TOutput)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetSubPrograms(TOutput)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">IsLeaf(TOutput)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">IsSubProgramOf(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is a program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">ProgramAt(TOutput)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Replace(TOutput)(UInt32, ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Replace(TOutput)(ITreeProgram(TOutput), ITreeProgram(TOutput))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by

		newSubProgram. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
⬇	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
⬇	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
⬇	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[ITreeProgram\(TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## **ITreeProgram(TOutput).CreateNew Method**

Creates a new [ITreeProgram\(TOutput\)](#) of the same kind of this program with the given child programs.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
ITreeProgram<TOutput> CreateNew(  
    IList<ITreeProgram<TOutput>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(TOutput\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(TOutput\)](#)

A new [ITreeProgram\(TOutput\)](#) of the same kind of this program with the given child programs.

#### See Also

[ITreeProgram\(TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## [ITreeProgram\(TOutput\).GetPrimitive](#) Method

Gets a program node representing the primitive of the corresponding [ITreeProgram\(TOutput\)](#) derived type.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
ITreeProgram<TOutput> GetPrimitive()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(TOutput\)](#)

The program node representing the primitive of the corresponding [ITreeProgram\(TOutput\)](#) derived type.

### See Also

[ITreeProgram\(TOutput\)Interface](#)

[Genetica.Elements Namespace](#)

## [ITreeProgram<TOutput>.Simplify Method](#)

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
ITreeProgram<TOutput> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram<TOutput>](#)

A program corresponding to a simplification of the given program.

### See Also

[ITreeProgram<TOutput>Interface](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter Class

Represents a class include a set of methods to convert [MathProgram](#) objects to and from [String](#) expressions in different notations.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Elements.MathExpressionConverter

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class MathExpressionConverter : ITreeExpressionConverter<MathProgram>,
IExpressionConverter<MathProgram>, IDisposable
```

[View Source](#)

The **MathExpressionConverter** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">MathExpressionConverter</a>	Creates a new <b>MathExpressionConverter</b> according to the given primitive set.

### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the <b>MathExpressionConverter</b>
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">FromNormalNotation</a>	Converts the given <a href="#">String</a> expression written in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2) to a <a href="#">MathProgram</a> .
	<a href="#">FromPrefixNotation</a>	Converts the given <a href="#">String</a> expression written in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...) to a <a href="#">MathProgram</a> .
	<a href="#">FromString</a>	Converts the given <a href="#">String</a> to a program of type <a href="#">TProgram</a> .
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)

 	<a href="#">ToNormalNotation</a>	Converts the given <i>TProgram</i> to a <a href="#">String</a> expression in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2).
 	<a href="#">ToNormalNotationExpression</a>	Converts the given <a href="#">MathProgram</a> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
 	<a href="#">ToPrefixNotation</a>	Converts the given <i>TProgram</i> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
 	<a href="#">ToPrefixNotationExpression</a>	Converts the given <a href="#">MathProgram</a> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
 	<a href="#">ToString()</a>	(Inherited from <a href="#">Object</a> .)
 	<a href="#">ToString(MathProgram)</a>	Converts the given <i>TProgram</i> to a string.

## See Also

[Genetica.Elements Namespace](#)

## MathExpressionConverter Constructor

Creates a new [MathExpressionConverter](#) according to the given primitive set.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public MathExpressionConverter(  
    PrimitiveSet<MathProgram> primitives  
)
```

[View Source](#)

#### Parameters

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(MathProgram\)](#)

The set containing the primitives that the converter can read from the expressions.

#### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.MathExpressionConverter Methods

The [MathExpressionConverter](#) type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">Dispose</a>	
≡	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">FromNormalNotation</a>	Converts the given <a href="#">String</a> expression written in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2) to a <a href="#">MathProgram</a> .
≡	<a href="#">FromPrefixNotation</a>	Converts the given <a href="#">String</a> expression written in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...) to a <a href="#">MathProgram</a> .
≡	<a href="#">FromString</a>	Converts the given <a href="#">String</a> to a program of type <a href="#">TProgram</a> .
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">ToNormalNotation</a>	Converts the given <a href="#">TProgram</a> to a <a href="#">String</a> expression in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2).
≡	<a href="#">ToNormalNotationExpression</a>	Converts the given <a href="#">MathProgram</a> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
≡	<a href="#">ToPrefixNotation</a>	Converts the given <a href="#">TProgram</a> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
≡	<a href="#">ToPrefixNotationExpression</a>	Converts the given <a href="#">MathProgram</a> to a <a href="#">String</a> expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).
≡	<a href="#">ToString()</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">ToString(MathProgram)</a>	Converts the given <a href="#">TProgram</a> to a string.

### See Also

[MathExpressionConverter Class](#)  
[Genetica.Elements Namespace](#)

## MathExpressionConverter.Dispose Method

Releases all resources used by the [MathExpressionConverter](#)

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.FromNormalNotation Method

Converts the given [String](#) expression written in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2) to a [MathProgram](#).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public MathProgram FromNormalNotation(  
    string expression  
)
```

[View Source](#)

#### Parameters

expression

Type: [System.String](#)

The expression we want to convert to an program.

#### Return Value

Type: [MathProgram](#)

A [MathProgram](#) converted from the given expression.

#### Implements

[ITreeExpressionConverter\(TProgram\).FromNormalNotation\(String\)](#)

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.FromPrefixNotation Method

Converts the given [String](#) expression written in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...) to a [MathProgram](#).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public MathProgram FromPrefixNotation(  
    string expression  
)
```

[View Source](#)

#### Parameters

expression

Type: [System.String](#)

The expression we want to convert to an program.

#### Return Value

Type: [MathProgram](#)

A [MathProgram](#) converted from the given expression.

#### Implements

[ITreeExpressionConverter\(TProgram\).FromPrefixNotation\(String\)](#)

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.FromString Method

Converts the given [String](#) to a program of type [TProgram](#).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public MathProgram FromString(  
    string programStr  
)
```

[View Source](#)

#### Parameters

*programStr*

Type: [System.String](#)

The string representing some program.

#### Return Value

Type: [MathProgram](#)

The program corresponding to the given string representation.

#### Implements

[IExpressionConverter\(TProgram\).FromString\(String\)](#)

#### Remarks

Equivalent to [FromPrefixNotation\(String\)](#).

#### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.ToNormalNotation Method

Converts the given *TProgram* to a [String](#) expression in normal notation, i.e., where functions are written in the form func(arg1 arg2 ...) or (arg1 func arg2).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public string ToNormalNotation(  
    MathProgram program  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to convert to an expression.

### Return Value

Type: [String](#)

A [String](#) representing the given program in prefix notation.

### Implements

[ITreeExpressionConverter\(TProgram\).ToNormalNotation\(TProgram\)](#)

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## [MathExpressionConverter](#).[ToNormalNotationExpression](#) Method

Converts the given [MathProgram](#) to a [String](#) expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static string ToNormalNotationExpression(  
    MathProgram program  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to convert to an expression.

### Return Value

Type: [String](#)

A [String](#) representing the given program in prefix notation.

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.ToPrefixNotation Method

Converts the given *TProgram* to a [String](#) expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public string ToPrefixNotation(  
    MathProgram program,  
    bool includeParentheses = true  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to convert to an expression.

*includeParentheses* (Optional)

Type: [System.Boolean](#)

Whether to write opening '(' and closing ')' parentheses when writing the expression of functions.

#### Return Value

Type: [String](#)

A [String](#) representing the given program in prefix notation.

#### Implements

[ITreeExpressionConverter\(TProgram\).ToPrefixNotation\(TProgram, Boolean\)](#)

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.ToPrefixNotationExpression Method

Converts the given [MathProgram](#) to a [String](#) expression in prefix notation, i.e., where functions are written in the form (func arg1 arg2 ...).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static string ToPrefixNotationExpression(  
    MathProgram program,  
    bool includeParentheses = true  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to convert to an expression.

*includeParentheses* (Optional)

Type: [System.Boolean](#)

Whether to write opening '(' and closing ')' parentheses when writing the expression of functions.

### Return Value

Type: [String](#)

A [String](#) representing the given program in prefix notation.

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## MathExpressionConverter.ToString Method

### Overload List

	Name	Description
	<a href="#">ToString()</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString(MathProgram)</a>	Converts the given <i>TProgram</i> to a string.

### See Also

[MathExpressionConverter Class](#)

[Genetica.Elements Namespace](#)

## [MathExpressionConverter.ToString Method \(MathProgram\)](#)

Converts the given *TProgram* to a string.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public string ToString(  
    MathProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to convert.

#### Return Value

Type: [String](#)

The string representing the given program.

#### Implements

[IExpressionConverter\(TProgram\).ToString\(TProgram\)](#)

#### Remarks

Equivalent to [ToPrefixNotationExpression\(MathProgram, Boolean\)](#).

#### See Also

[MathExpressionConverter Class](#)

[ToString Overload](#)

[Genetica.Elements Namespace](#)

## MathPrimitiveSets Class

Contains different [PrimitiveSet\(TProgram\)](#) for [MathProgram](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Elements.MathPrimitiveSets

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static class MathPrimitiveSets
```

[View Source](#)

The **MathPrimitiveSets** type exposes the following members.

### Fields

	Name	Description
◆	<a href="#">Default</a>	Gets the default <a href="#">PrimitiveSet(TProgram)</a> for <a href="#">MathProgram</a> containing <a href="#">Zero</a> and <a href="#">One</a> as the terminals and all available mathematical functions.
◆		

### See Also

[Genetica.Elements Namespace](#)

## MathPrimitiveSets.MathPrimitiveSets Fields

The [MathPrimitiveSets](#) type exposes the following members.

### Fields

	Name	Description
◆	<a href="#">Default</a>	Gets the default <a href="#">PrimitiveSet(TProgram)</a> for <a href="#">MathProgram</a> containing <a href="#">Zero</a> and <a href="#">One</a> as the terminals and all available mathematical functions.
◆	<a href="#">S</a>	

### See Also

[MathPrimitiveSets Class](#)

[Genetica.Elements Namespace](#)

## MathPrimitiveSets.Default Field

Gets the default [PrimitiveSet\(TProgram\)](#) for [MathProgram](#) containing [Zero](#) and [One](#) as the terminals and all available mathematical functions.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static readonly PrimitiveSet<MathProgram> Default
```

[View Source](#)

### Field Value

Type: [PrimitiveSet\(MathProgram\)](#)

### See Also

[MathPrimitiveSets Class](#)

[Genetica.Elements Namespace](#)

## MathProgram Class

Represents a base class for mathematical program programs. A **MathProgram** represents a mathematical expression whose output is a double-precision value. Math programs are useful to perform symbolic regression.

### Inheritance Hierarchy

#### [System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

[Genetica.Elements.Functions.IfFunction](#)

[Genetica.Elements.Functions.UnaryFunction](#)

[Genetica.Elements.Terminals.Terminal](#)

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public abstract class MathProgram : ITreeProgram<double>,
    IProgram<IReadOnlyList<ITreeProgram<double>>, double>,
    IProgram, ITreeProgram, ITreeNode, IComparable<ITreeProgram<double>>
```

[View Source](#)

The **MathProgram** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">MathProgram</a>	Creates a new <b>MathProgram</b> with the given children / input nodes.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root).
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <b>MathProgram</b> descendants of this program.
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes.
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <b>MathProgram</b> descendant programs encountered starting from this program.

## Methods

	Name	Description
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareOrdinal</code> comparison over their <a href="#">Expression</a> .
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> .
≡	<a href="#">CreateNew</a>	Creates a new <b>MathProgram</b> which is a copy of this program.
≡	<a href="#">Equals(Object)</a>	(Overrides <code>Object.Equals(Object)</code> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another.
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Overrides <code>Object.GetHashCode()</code> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <b>MathProgram</b> derived type. For functions this corresponds to a <b>MathProgram</b> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself.
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.
≡	<a href="#">ToString</a>	(Overrides <code>Object.ToString()</code> .)

## Operators

	Name	Description
≡≡S	<a href="#">Equality</a>	Checks if two <b>MathProgram</b> are equal.
≡≡S	<a href="#">Inequality</a>	Checks if two <b>MathProgram</b> are not equal.

## Extension Methods

	Name	Description
≡	<a href="#">ContainsConstant</a>	Verifies whether the <b>MathProgram</b> contains a constant value, i.e., whether any one of its descendant programs are instances have a constant value equal to <code>val</code> . (Defined by <a href="#">MathProgramExtensions</a> .)
≡	<a href="#">ContainsSubElement(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">EqualsConstant</a>	Verifies whether the <b>MathProgram</b> is a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> , and whether the associated value equals to <i>val</i> . (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">GetCommonRegionIndexes(Double)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetLeaves(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxBreadth(Double)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxDepth(Double)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetPrimitives(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetRange</a>	Computes a <a href="#">Range</a> representing the minimum and maximum values that a given <b>MathProgram</b> can compute, as dictated by its sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">GetSubCombinations(Double)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetSubPrograms(Double)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">GetValueRmsd</a>	Gets the root-mean-square deviation (RMSD) between the values computed for the given programs. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the square of the difference between the values computed by both programs is calculated. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">IsConstant</a>	Verifies whether the <b>MathProgram</b> has a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> . (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">IsLeaf(Double)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">IsSubProgramOf(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">IsValueEquivalent</a>	Checks whether the given <b>MathProgram</b> computes a value that is consistently equivalent to that computed by another program according to some margin. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the difference between the values computed by both programs is calculated. If the difference is less than a given margin for all the trials, then the programs are considered to be value-equivalent. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">ProgramAt(Double)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Replace(Double)(UInt32, ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Replace(Double)(ITreeProgram(Double), ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to

		oldSubProgram are replaced by newSubProgram. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Simplify</a>		Simplifies the expression of the given <b>MathProgram</b> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <i>margin</i> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">ToD3JsonFile</a>		Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>		Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>		Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>		Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[Genetica.Elements Namespace](#)

## MathProgram Constructor

Creates a new [MathProgram](#) with the given children / input nodes.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected MathProgram(  
    ITreeProgram<double>[] children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [Genetica.Elements.ITreeProgram\(Double\)\[\]](#)

The input nodes to this math program.

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.MathProgram Properties

The [MathProgram](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root).
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program.
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes.
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program.

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Input Property

Gets the children of this program, i.e., the direct [MathProgram](#) descendants of this program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IReadOnlyList<ITreeProgram<double>> Input { get; }
```

[View Source](#)

### Property Value

Type: [IReadOnlyList\(ITreeProgram\(Double\)\)](#)

### Implements

[IProgram\(TInput, TOutput\).Input](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract string Label { get; }
```

[View Source](#)

*Property Value*

Type: [String](#)

*Implements*

[ITreeProgram.Label](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Length Property

Gets the program's length, i.e., the total number of [MathProgram](#) descendant programs encountered starting from this program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ushort Length { get; }
```

[View Source](#)

### Property Value

Type: [UInt16](#)

Implements

[IProgram.Length](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.MathProgram Methods

The [MathProgram](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareOrdinal</code> comparison over their <a href="#">Expression</a> .
	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> .
	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program.
	<a href="#">Equals(Object)</a>	(Overrides <a href="#">Object.Equals(Object)</a> .)
	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another.
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Overrides <a href="#">Object.GetHashCode()</a> .)
	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself.
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.
	<a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

### Extension Methods

	Name	Description
	<a href="#">ContainsConstant</a>	Verifies whether the <a href="#">MathProgram</a> contains a constant value, i.e., whether any one of its descendant programs are instances have a constant value equal to <code>val</code> . (Defined by <a href="#">MathProgramExtensions</a> .)
	<a href="#">ContainsSubElement(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
	<a href="#">EqualsConstant</a>	Verifies whether the <a href="#">MathProgram</a> is a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> , and

		whether the associated value equals to <i>val</i> . (Defined by <a href="#">MathProgramExtensions</a> .)
💡	<a href="#">GetCommonRegionIndexes(Double)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetLeaves(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetMaxBreadth(Double)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetMaxDepth(Double)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetPrimitives(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetRange</a>	Computes a <a href="#">Range</a> representing the minimum and maximum values that a given <a href="#">MathProgram</a> can compute, as dictated by its sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
💡	<a href="#">GetSubCombinations(Double)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetSubPrograms(Double)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">GetValueRmsd</a>	Gets the root-mean-square deviation (RMSD) between the values computed for the given programs. The method works by substituting

		the variables in both programs' expressions by random values for a certain number of trials. In each trial, the square of the difference between the values computed by both programs is calculated. (Defined by <a href="#">MathProgramExtensions</a> .)
💡	<a href="#">IsConstant</a>	Verifies whether the <a href="#">MathProgram</a> has a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> . (Defined by <a href="#">MathProgramExtensions</a> .)
💡	<a href="#">IsLeaf(Double)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">IsSubProgramOf(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is a program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">IsValueEquivalent</a>	Checks whether the given <a href="#">MathProgram</a> computes a value that is consistently equivalent to that computed by another program according to some margin. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the difference between the values computed by both programs is calculated. If the difference is less than a given margin for all the trials, then the programs are considered to be value-equivalent. (Defined by <a href="#">MathProgramExtensions</a> .)
💡	<a href="#">ProgramAt(Double)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">Replace(Double)(UInt32, ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
💡	<a href="#">Replace(Double)(ITreeProgram(Double), ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i> . (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <i>margin</i> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.CompareTo Method

Compares this program with another program using a `string.CompareTo` comparison over their [Expression](#).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public int CompareTo(  
    ITreeProgram<double> other  
)
```

[View Source](#)

### Parameters

`other`

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The other program to compare to.

### Return Value

Type: [Int32](#)

A value indicating if this program is less, equal or more than the other program, according to their [Expression](#) comparison.

### Implements

[IComparable\(T\).CompareTo\(T\)](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## [MathProgram.CreateNew Method](#)

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Equals Method

### Overload List

	Name	Description
	<a href="#">Equals(Object)</a>	(Overrides <a href="#">Object.Equals(Object)</a> .)
	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another.

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Equals Method (Object)

[Missing <summary> documentation for  
"M:Genetica.Elements.MathProgram.Equals(System.Object)"]

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override bool Equals(  
    Object obj  
)
```

[View Source](#)

### Parameters

*obj*

Type: [System.Object](#)

[Missing <param name="obj"/> documentation for  
"M:Genetica.Elements.MathProgram.Equals(System.Object)"]

### Return Value

Type: [Boolean](#)

[Missing <returns> documentation for  
"M:Genetica.Elements.MathProgram.Equals(System.Object)"]

### See Also

[MathProgram Class](#)

[Equals Overload](#)

[Genetica.Elements Namespace](#)

## MathProgram.Equals Method (MathProgram)

Checks whether this program is equal to another.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public bool Equals(  
    MathProgram other  
)
```

[View Source](#)

#### Parameters

other

Type: [Genetica.Elements.MathProgram](#)

The other program to check for equality.

#### Return Value

Type: [Boolean](#)

true if the objects are the same or have the same [Label](#) and their [Input](#) sequence is the same.

#### See Also

[MathProgram Class](#)

[Equals Overload](#)

[Genetica.Elements Namespace](#)

## MathProgram.GetHashCode Method

[Missing <summary> documentation for "M:Genetica.Elements.MathProgram.GetHashCode"]

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override int GetHashCode()
```

[View Source](#)

### Return Value

Type: [Int32](#)

[Missing <returns> documentation for "M:Genetica.Elements.MathProgram.GetHashCode"]

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.GetPrimitive Method

Gets a program node representing the primitive of the corresponding [MathProgram](#) derived type. For functions this corresponds to a [MathProgram](#) whose inputs are [Zero](#), for terminals it returns the terminal itself.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> GetPrimitive()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

The default program node of the corresponding [MathProgram](#) derived type.

### Implements

[ITreeProgram\(TOutput\).GetPrimitive\(\)](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public virtual ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

*Implements*

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.ToString Method

[Missing <summary> documentation for "M:Genetica.Elements.MathProgram.ToString"]

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string ToString()
```

[View Source](#)

**Return Value**

Type: [String](#)

[Missing <returns> documentation for "M:Genetica.Elements.MathProgram.ToString"]

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgram.MathProgram Operators

The [MathProgram](#) type exposes the following members.

### Operators

	Name	Description
	<a href="#">Equality</a>	Checks if two <a href="#">MathProgram</a> are equal.
	<a href="#">Inequality</a>	Checks if two <a href="#">MathProgram</a> are not equal.

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## [MathProgram.Equality Operator](#)

Checks if two [MathProgram](#) are equal.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public static bool operator ==(  
    MathProgram left,  
    MathProgram right  
)
```

[View Source](#)

### Parameters

*left*

Type: [Genetica.Elements.MathProgram](#)

The first program.

*right*

Type: [Genetica.Elements.MathProgram](#)

The second program.

### Return Value

Type: [Boolean](#)

*true* if the programs are equal *false* otherwise.

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## [MathProgram.Inequality Operator](#)

Checks if two [MathProgram](#) are not equal.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public static bool operator !=(
    MathProgram left,
    MathProgram right
)
```

[View Source](#)

### Parameters

*left*

Type: [Genetica.Elements.MathProgram](#)

The first program.

*right*

Type: [Genetica.Elements.MathProgram](#)

The second program.

### Return Value

Type: [Boolean](#)

*true* if the programs are not equal *false* otherwise.

### See Also

[MathProgram Class](#)

[Genetica.Elements Namespace](#)

## MathProgramExtensions Class

Declares a set of extension methods for [MathProgram](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Elements.MathProgramExtensions

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static class MathProgramExtensions
```

[View Source](#)

The **MathProgramExtensions** type exposes the following members.

### Methods

	Name	Description
 	<a href="#">ContainsConstant</a>	Verifies whether the <a href="#">MathProgram</a> contains a constant value, i.e., whether any one of its descendant programs are instances have a constant value equal to <i>val</i> .
 	<a href="#">EqualsConstant</a>	Verifies whether the <a href="#">MathProgram</a> is a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> , and whether the associated value equals to <i>val</i> .
 	<a href="#">GetRange</a>	Computes a <a href="#">Range</a> representing the minimum and maximum values that a given <a href="#">MathProgram</a> can compute, as dictated by its sub-programs.
 	<a href="#">GetValueRmsd</a>	Gets the root-mean-square deviation (RMSD) between the values computed for the given programs. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the square of the difference between the values computed by both programs is calculated.
 	<a href="#">IsConstant</a>	Verifies whether the <a href="#">MathProgram</a> has a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> .
 	<a href="#">IsValueEquivalent</a>	Checks whether the given <a href="#">MathProgram</a> computes a value that is consistently equivalent to that computed by another program according to some margin. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the difference between the values computed by both programs is calculated. If the difference is less than a given margin for all the trials, then the programs are considered to be value-equivalent.

 <b>Simplify</b>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <i>margin</i> , and whose expression is naturally simpler, i.e., has fewer sub-programs.
---	--

## See Also

[Genetica.Elements Namespace](#)

## MathProgramExtensions.MathProgramExtensions Methods

The [MathProgramExtensions](#) type exposes the following members.

### Methods

	Name	Description
 	<a href="#">ContainsConstant</a>	Verifies whether the <a href="#">MathProgram</a> contains a constant value, i.e., whether any one of its descendant programs are instances have a constant value equal to <i>val</i> .
 	<a href="#">EqualsConstant</a>	Verifies whether the <a href="#">MathProgram</a> is a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> , and whether the associated value equals to <i>val</i> .
 	<a href="#">GetRange</a>	Computes a <a href="#">Range</a> representing the minimum and maximum values that a given <a href="#">MathProgram</a> can compute, as dictated by its sub-programs.
 	<a href="#">GetValueRmsd</a>	Gets the root-mean-square deviation (RMSD) between the values computed for the given programs. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the square of the difference between the values computed by both programs is calculated.
 	<a href="#">IsConstant</a>	Verifies whether the <a href="#">MathProgram</a> has a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> .
 	<a href="#">IsValueEquivalent</a>	Checks whether the given <a href="#">MathProgram</a> computes a value that is consistently equivalent to that computed by another program according to some margin. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the difference between the values computed by both programs is calculated. If the difference is less than a given margin for all the trials, then the programs are considered to be value-equivalent.
 	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <i>margin</i> , and whose expression is naturally simpler, i.e., has fewer sub-programs.

### See Also

[MathProgramExtensions Class](#)  
[Genetica.Elements Namespace](#)

## MathProgramExtensions.ContainsConstant Method

Verifies whether the [MathProgram](#) contains a constant value, i.e., whether any one of its descendant programs are instances have a constant value equal to *val*.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static bool ContainsConstant(  
    this ITreeProgram<double> program,  
    double val  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The program to verify whether it contains a constant.

*val*

Type: [System.Double](#)

The value to test for the program.

#### Return Value

Type: [Boolean](#)

true, if program contains a constant value equal to *val*, false otherwise.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(Double\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[MathProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## MathProgramExtensions.EqualsConstant Method

Verifies whether the [MathProgram](#) is a constant value, i.e., whether all its descendant leaf programs are instances of [Constant](#), and whether the associated value equals to *val*.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static bool EqualsConstant(  
    this ITreeProgram<double> program,  
    double val  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The program to verify whether it is a constant.

*val*

Type: [System.Double](#)

The value to test for the program.

### Return Value

Type: [Boolean](#)

true, if program is a constant and its value equals *val*, false otherwise.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(Double\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[MathProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## MathProgramExtensions.GetRange Method

Computes a [Range](#) representing the minimum and maximum values that a given [MathProgram](#) can compute, as dictated by its sub-programs.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static Range GetRange(  
    this ITreeProgram<double> program  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The program whose range we want to compute.

### Return Value

Type: [Range](#)

The range of the given program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(Double\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[MathProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## MathProgramExtensions.GetValueRmsd Method

Gets the root-mean-square deviation (RMSD) between the values computed for the given programs. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the square of the difference between the values computed by both programs is calculated.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static double GetValueRmsd(  
    this ITreeProgram<double> program,  
    ITreeProgram<double> other,  
    uint numTrials = 1000  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first program that we want to test.

*other*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second program that we want to test.

*numTrials* (Optional)

Type: [System.UInt32](#)

The number of trials used to compute the squared difference.

### Return Value

Type: [Double](#)

The RMSD between the several values computed for the given programs.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(Double\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[MathProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## MathProgramExtensions.IsConstant Method

Verifies whether the [MathProgram](#) has a constant value, i.e., whether all its descendant leaf programs are instances of [Constant](#).

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static bool IsConstant(  
    this ITreeProgram<double> program  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The program to verify whether it is a constant.

### Return Value

Type: [Boolean](#)

true, if all leaf programs are constant, false otherwise.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(Double\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[MathProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## MathProgramExtensions.IsValueEquivalent Method

Checks whether the given [MathProgram](#) computes a value that is consistently equivalent to that computed by another program according to some margin. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the difference between the values computed by both programs is calculated. If the difference is less than a given margin for all the trials, then the programs are considered to be value-equivalent.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static bool IsValueEquivalent(  
    this ITreeProgram<double> program,  
    ITreeProgram<double> other,  
    double margin = 1E-06,  
    uint numTrials = 1000  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first program that we want to test.

*other*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second program that we want to test.

*margin* (Optional)

Type: [System.Double](#)

The margin used to compare against the difference of values computed by both expressions in each trial.

*numTrials* (Optional)

Type: [System.UInt32](#)

The number of trials used to discern whether the given programs are equivalent.

### Return Value

Type: [Boolean](#)

True if the given programs are considered to be value-equivalent, False otherwise.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(Double\)](#). When you use instance method syntax to call this method, omit the first

parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

**See Also**

[MathProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## MathProgramExtensions.Simplify Method

Simplifies the expression of the given [MathProgram](#) by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of *margin*, and whose expression is naturally simpler, i.e., has fewer sub-programs.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static ITreeProgram<double> Simplify(  
    this MathProgram program,  
    IFitnessFunction<MathProgram> fitnessFunction,  
    double margin = 1E-06  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to simplify.

*fitnessFunction*

Type: [Genetica.Evaluation.IFitnessFunction\(MathProgram\)](#)

The fitness function used to determine equivalence.

*margin* (Optional)

Type: [System.Double](#)

The acceptable difference between the fitness of the given program and that of a simplified program for them to be considered equivalent.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

An program corresponding to a simplification of the given program.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [MathProgram](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[MathProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram) Class

Represents a collection of TProgram primitives including a set of functions and terminals.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Elements.PrimitiveSet(TProgram)

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class PrimitiveSet<TProgram> : IDisposable  
where TProgram : ITreeProgram
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

The PrimitiveSet(TProgram) type exposes the following members.

#### Constructors

	Name	Description
	<a href="#">PrimitiveSet(TProgram)</a>	Creates a new PrimitiveSet(TProgram) with the give terminals and functions.

#### Properties

	Name	Description
	<a href="#">Functions</a>	Gets the programs representing the function primitives.
	<a href="#">Terminals</a>	Gets the programs representing the terminal primitives.

#### Methods

	Name	Description
	<a href="#">Add</a>	Adds the functions and terminals in the given PrimitiveSet(TProgram) to this set.
	<a href="#">Dispose</a>	Releases all resources used by the PrimitiveSet(TProgram)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

## See Also

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram) Constructor

Creates a new [PrimitiveSet\(TProgram\)](#) with the give terminals and functions.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public PrimitiveSet(  
    IEnumerable<TProgram> terminals,  
    IEnumerable<TProgram> functions  
)
```

[View Source](#)

#### Parameters

*terminals*

Type: [System.Collections.Generic.IEnumerable\(TProgram\)](#)

The programs representing the terminal primitives.

*functions*

Type: [System.Collections.Generic.IEnumerable\(TProgram\)](#)

The programs representing the function primitives.

### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram).PrimitiveSet(TProgram) Properties

The [PrimitiveSet\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">Functions</a>	Gets the programs representing the function primitives.
	<a href="#">Terminals</a>	Gets the programs representing the terminal primitives.

### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram).Functions Property

Gets the programs representing the function primitives.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IReadOnlyCollection<TProgram> Functions { get; }
```

[View Source](#)

### Property Value

Type: [IReadOnlyCollection\(TProgram\)](#)

### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram).Terminals Property

Gets the programs representing the terminal primitives.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IReadOnlyCollection<TProgram> Terminals { get; }
```

[View Source](#)

### Property Value

Type: [IReadOnlyCollection\(TProgram\)](#)

### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram).PrimitiveSet(TProgram) Methods

The [PrimitiveSet\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Add</a>	Adds the functions and terminals in the given <a href="#">PrimitiveSet(TProgram)</a> to this set.
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram).Add Method

Adds the functions and terminals in the given [PrimitiveSet\(TProgram\)](#) to this set.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Add(  
    PrimitiveSet<TProgram> primitiveSet  
)
```

[View Source](#)

#### Parameters

primitiveSet

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The set containing the functions and terminals to be added to this set.

#### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram).Dispose Method

Releases all resources used by the [PrimitiveSet\(TProgram\)](#)

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## PrimitiveSet(TProgram).ToString Method

[Missing <summary> documentation for "M:Genetica.Elements.PrimitiveSet`1.ToString"]

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string ToString()
```

[View Source](#)

**Return Value**

Type: [String](#)

[Missing <returns> documentation for "M:Genetica.Elements.PrimitiveSet`1.ToString"]

### See Also

[PrimitiveSet\(TProgram\)Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions Class

Declares a set of extension methods for [ITreeProgram\(TOutput\)](#) objects.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Elements.TreeProgramExtensions

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static class TreeProgramExtensions
```

[View Source](#)

The **TreeProgramExtensions** type exposes the following members.

### Methods

	Name	Description
 	<a href="#">ContainsSubElement(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program.
 	<a href="#">GetCommonRegionIndexes(TOutput)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs.
 	<a href="#">GetLeaves(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program.
 	<a href="#">GetMaxBreadth(TOutput)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program.
 	<a href="#">GetMaxDepth(TOutput)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program.

 <a href="#">GetPrimitives(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count.
 <a href="#">GetSubCombinations(TOutput)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child.
 <a href="#">GetSubPrograms(TOutput)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program.
 <a href="#">IsLeaf(TOutput)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes.
 <a href="#">IsSubProgramOf(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is a program that is a descendant of a given program.
 <a href="#">ProgramAt(TOutput)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner.
 <a href="#">Replace(TOutput)(ITreeProgram(TOutput), ITreeProgram(TOutput), ITreeProgram(TOutput))</a>	Gets a new copy of program where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i> .
 <a href="#">Replace(TOutput)(ITreeProgram(TOutput), UInt32, ITreeProgram(TOutput))</a>	Gets a new copy of program where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner.

## See Also

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.TreeProgramExtensions Methods

The [TreeProgramExtensions](#) type exposes the following members.

### Methods

Name	Description
  <a href="#">ContainsSubElement(TOutput)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program.
  <a href="#">GetCommonRegionIndexes(TOutput)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs.
  <a href="#">GetLeaves(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program.
  <a href="#">GetMaxBreadth(TOutput)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program.
  <a href="#">GetMaxDepth(TOutput)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program.
  <a href="#">GetPrimitives(TOutput)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count.
  <a href="#">GetSubCombinations(TOutput)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child.
  <a href="#">GetSubPrograms(TOutput)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program.
  <a href="#">IsLeaf(TOutput)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes.

 <a href="#">IsSubProgramOf(TOutput)</a> 	<p>Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a>. A sub-program is a program that is a descendant of a given program.</p>
 <a href="#">ProgramAt(TOutput)</a> 	<p>Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner.</p>
 <a href="#">Replace(TOutput)(ITreeProgram(TOutput), ITreeProgram(TOutput), ITreeProgram(TOutput))</a> 	<p>Gets a new copy of program where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i>.</p>
 <a href="#">Replace(TOutput)(ITreeProgram(TOutput), UInt32, ITreeProgram(TOutput))</a> 	<p>Gets a new copy of program where the descendant program at the given index is replaced by <i>newSubProgram</i>. Elements are indexed in a zero-based, depth first search manner.</p>

## See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.ContainsSubElement(TOutput) Method

Checks whether a given [ITreeProgram\(TOutput\)](#) contains the given sub-program. A sub-program is an program that is a descendant of a given program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static bool ContainsSubElement<TOutput>(  
    this ITreeProgram<TOutput> program,  
    ITreeProgram<TOutput> other  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program for which to look for a descendant equal to the given program.

*other*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program we want to know if it is a sub-program.

#### Type Parameters

*TOutput*

The type of program output.

#### Return Value

Type: [Boolean](#)

true, if the given program is a descendant of the given program, false otherwise.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.GetCommonRegionIndexes(TOutput)

### Method

Gets a [IDictionary\(TKey, TValue\)](#) representing the program indexes in the common region between *program* and *otherProgram*. The dictionary represents the index correspondence between the sub-programs of the given programs.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static IDictionary<uint, uint> GetCommonRegionIndexes<TOutput>(
    this ITreeProgram<TOutput> program,
    ITreeProgram<TOutput> otherProgram
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The first program

*otherProgram*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The other program to get the common region.

### Type Parameters

*TOutput*

The type of program output.

### Return Value

Type: [IDictionary\(UInt32, UInt32\)](#)

The index correspondence between the sub-programs of the given programs.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.GetLeaves(TOutput) Method

Gets a dictionary containing all the [ITreeProgram\(TOutput\)](#) leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static IDictionary<ITreeProgram<TOutput>, uint> GetLeaves<TOutput>(
    this ITreeProgram<TOutput> program
)
```

[View Source](#)

### Parameters

program

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program whose leaf sub-programs we want to retrieve.

### Type Parameters

TOutput

The type of program output.

### Return Value

Type: [IDictionary\(ITreeProgram\(TOutput\), UInt32\)](#)

A set containing all the [ITreeProgram\(TOutput\)](#) sub-programs of the given program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.GetMaxBreadth(TOutput) Method

Gets the maximum breadth of the program, i.e., the number of [ITreeProgram\(TOutput\)](#) leaves encountered starting from this program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static uint GetMaxBreadth<TOutput>(  
    this ITreeProgram<TOutput> program  
)
```

[View Source](#)

### Parameters

program

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The root program to calculate the breadth.

### Type Parameters

TOutput

The type of program output.

### Return Value

Type: [UInt32](#)

The maximum breadth of the program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.GetMaxDepth(TOutput) Method

Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a **Terminal** program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static uint GetMaxDepth<TOutput>(  
    this ITreeProgram<TOutput> program  
)
```

[View Source](#)

### Parameters

program

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The root program to calculate the depth.

### Type Parameters

TOutput

The type of program output.

### Return Value

Type: [UInt32](#)

The maximum depth of the program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.GetPrimitives(TOutput) Method

Gets a dictionary containing all the [ITreeProgram\(TOutput\)](#) primitives in the given program and their count.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static IDictionary<ITreeProgram<TOutput>, uint> GetPrimitives<TOutput>(
    this ITreeProgram<TOutput> program
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program whose terminal sub-programs we want to retrieve.

### Type Parameters

*TOutput*

The type of program output.

### Return Value

Type: [IDictionary\(ITreeProgram\(TOutput\), UInt32\)](#)

A set containing all the [ITreeProgram\(TOutput\)](#) primitives of the given program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.GetSubCombinations(TOutput) Method

Gets all the [ITreeProgram\(TOutput\)](#) sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static ISet<ITreeProgram<TOutput>> GetSubCombinations<TOutput>(
    this ITreeProgram<TOutput> program
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program we want to get the sub-combinations.

#### Type Parameters

*TOutput*

The type of program output.

#### Return Value

Type: [ISet\(ITreeProgram\(TOutput\)\)](#)

All the sub-combinations of the given program.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.GetSubPrograms(TOutput) Method

Gets a [ISet\(T\)](#) containing all the descendant [ITreeProgram\(TOutput\)](#) of the given program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static ITreeProgram<TOutput>[] GetSubPrograms<TOutput>(
    this ITreeProgram<TOutput> program
)
```

[View Source](#)

### Parameters

program

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program we want to get the sub-programs.

### Type Parameters

TOutput

The type of program output.

### Return Value

Type: [ITreeProgram\(TOutput\)\[\]](#)

A set containing all the sub programs of the given program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.IsLeaf(*TOutput*) Method

Checks whether the given [ITreeProgram\(\*TOutput\*\)](#) is a leaf node, i.e., it has no child nodes.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static bool IsLeaf<TOutput>(
    this ITreeProgram<TOutput> program
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(\*TOutput\*\)](#)

The program we want to check if it is a leaf.

### Type Parameters

*TOutput*

The type of program output.

### Return Value

Type: [Boolean](#)

*true* if the given program is a leaf, *false* otherwise.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(\*TOutput\*\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.IsSubProgramOf(TOutput) Method

Checks whether a given [ITreeProgram\(TOutput\)](#) is a sub-program of another [ITreeProgram\(TOutput\)](#). A sub-program is an program that is a descendant of a given program.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static bool IsSubProgramOf<TOutput>(
    this ITreeProgram<TOutput> program,
    ITreeProgram<TOutput> other
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program we want to know if it is a sub-program.

*other*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The program for which to look for a descendant equal to the given program.

#### Type Parameters

*TOutput*

The type of program output.

#### Return Value

Type: [Boolean](#)

true, if the program is a descendant of the other program, false otherwise.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.ProgramAt(TOutput) Method

Get the [ITreeProgram\(TOutput\)](#) at the given index, where programs are indexed in a zero-based, depth first search manner.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static ITreeProgram<TOutput> ProgramAt<TOutput>(
    this ITreeProgram<TOutput> program,
    uint index
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The root program to search for the child program at the given index.

*index*

Type: [System.UInt32](#)

The index of the program we want to search for.

#### Type Parameters

*TOutput*

The type of program output.

#### Return Value

Type: [ITreeProgram\(TOutput\)](#)

The [ITreeProgram\(TOutput\)](#) at the given index, or a null reference (Nothing in Visual Basic) if the given index is greater than or equal to the program's length.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.Replace Method

### Overload List

Name	Description
  <a href="#">Replace(TOutput)(ITreeProgram(TOutput), ITreeProgram(TOutput), ITreeProgram(TOutput))</a>	Gets a new copy of program where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i> .
  <a href="#">Replace(TOutput)(ITreeProgram(TOutput), UInt32, ITreeProgram(TOutput))</a>	Gets a new copy of program where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner.

### See Also

[TreeProgramExtensions Class](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.Replace(*TOutput*) Method (*ITreeProgram*(*TOutput*), *ITreeProgram*(*TOutput*), *ITreeProgram*(*TOutput*))

Gets a new copy of program where all sub-programs that are equal to *oldSubProgram* are replaced by *newSubProgram*.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static ITreeProgram<TOutput> Replace<TOutput>(
    this ITreeProgram<TOutput> program,
    ITreeProgram<TOutput> oldSubProgram,
    ITreeProgram<TOutput> newSubProgram
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram](#)(*TOutput*)

The root program to copy and search for the given sub-program .

*oldSubProgram*

Type: [Genetica.Elements.ITreeProgram](#)(*TOutput*)

The sub-program we want to replace.

*newSubProgram*

Type: [Genetica.Elements.ITreeProgram](#)(*TOutput*)

The new sub-program to replace the given one.

### Type Parameters

*TOutput*

The type of program output.

### Return Value

Type: [ITreeProgram](#)(*TOutput*)

A copy of the program with the given descendant replaced by the new program. If the given program is equal to the sub-program we want to replace, then the replacement is returned. If the given sub-program is not found, a copy of the original program is returned, or *null* if the program is *null*.

#### **Usage Note**

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### **See Also**

[TreeProgramExtensions Class](#)

[Replace Overload](#)

[Genetica.Elements Namespace](#)

## TreeProgramExtensions.Replace(*TOutput*) Method (*ITreeProgram(TOutput)*, *UInt32*, *ITreeProgram(TOutput)*)

Gets a new copy of *program* where the descendant program at the given index is replaced by *newSubProgram*. Elements are indexed in a zero-based, depth first search manner.

**Namespace:** [Genetica.Elements](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static ITreeProgram<TOutput> Replace<TOutput>(
    this ITreeProgram<TOutput> program,
    uint index,
    ITreeProgram<TOutput> newSubProgram
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The root program to copy and search for the sub-program at the given index.

*index*

Type: [System.UInt32](#)

The index of the sub-program we want to replace.

*newSubProgram*

Type: [Genetica.Elements.ITreeProgram\(TOutput\)](#)

The new program to replace the one at the given index.

### Type Parameters

*TOutput*

The type of program output.

### Return Value

Type: [ITreeProgram\(TOutput\)](#)

A copy of the program with the descendant at the given index replaced by the new program.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeProgram\(TOutput\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

## See Also

[TreeProgramExtensions Class](#)

[Replace Overload](#)

[Genetica.Elements Namespace](#)

## Genetica.Elements.Functions Namespace

### Classes

Class	Description
 <a href="#">AdditionFunction</a>	Represents a <a href="#">CommutativeBinaryFunction</a> performing the sum of two sub-programs.
 <a href="#">BinaryFunction</a>	Represents a <a href="#">MathProgram</a> whose result is a mathematical operation involving two input sub-programs.
 <a href="#">CommutativeBinaryFunction</a>	Represents a <a href="#">BinaryFunction</a> in which the order of the operands is not important. Therefore they are sorted to avoid creating repeated functions.
 <a href="#">CosineFunction</a>	Represents a <a href="#">UnaryFunction</a> performing the cosine operation over some sub-program.
 <a href="#">DivisionFunction</a>	Represents a <a href="#">BinaryFunction</a> performing the division between two sub-programs.
 <a href="#">IfFunction</a>	Represents a <a href="#">MathProgram</a> performing a conditional operation according to the value of the first sub-program.
 <a href="#">LogarithmFunction</a>	Represents a <a href="#">BinaryFunction</a> performing the logarithm operation.
 <a href="#">MaxFunction</a>	Represents a <a href="#">CommutativeBinaryFunction</a> selecting the maximum between two sub-programs.
 <a href="#">MinFunction</a>	Represents a <a href="#">CommutativeBinaryFunction</a> selecting the minimum between two sub-programs.
 <a href="#">MultiplicationFunction</a>	Represents a <a href="#">CommutativeBinaryFunction</a> performing the multiplication of two sub-programs.
 <a href="#">PowerFunction</a>	Represents a <a href="#">BinaryFunction</a> performing the exponentiation operation.
 <a href="#">SineFunction</a>	Represents a <a href="#">UnaryFunction</a> performing the sine operation over some sub-program.
 <a href="#">SubtractionFunction</a>	Represents a <a href="#">BinaryFunction</a> performing the subtraction operation between two sub-programs.
 <a href="#">UnaryFunction</a>	Represents a <a href="#">MathProgram</a> whose result is a mathematical operation involving a single input sub-program.

## AdditionFunction Class

Represents a [CommutativeBinaryFunction](#) performing the sum of two sub-programs.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

[Genetica.Elements.Functions.CommutativeBinaryFunction](#)

Genetica.Elements.Functions.AdditionFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class AdditionFunction : CommutativeBinaryFunction
```

[View Source](#)

The **AdditionFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">AdditionFunction</a>	Creates a new <b>AdditionFunction</b> .

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction Constructor

Creates a new [AdditionFunction](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public AdditionFunction(  
    ITreeProgram<double> firstProgram,  
    ITreeProgram<double> secondProgram  
)
```

[View Source](#)

#### Parameters

*firstProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first sub-program.

*secondProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second sub-program.

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction.AdditionFunction Properties

The [AdditionFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction.AdditionFunction Methods

The [AdditionFunction](#) type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

	Name	Description
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## AdditionFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[AdditionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction Class

Represents a [MathProgram](#) whose result is a mathematical operation involving two input sub-programs.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

[Genetica.Elements.Functions.CommutativeBinaryFunction](#)

[Genetica.Elements.Functions.DivisionFunction](#)

[Genetica.Elements.Functions.LogarithmFunction](#)

[Genetica.Elements.Functions.PowerFunction](#)

[Genetica.Elements.Functions.SubtractionFunction](#)

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract class BinaryFunction : MathProgram
```

[View Source](#)

The **BinaryFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">BinaryFunction</a>	Creates a new <b>BinaryFunction</b> with the given programs as parameters.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program.
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)

	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program.

## Methods

	Name	Description
	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	Name	Description
	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

**See Also**

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction Constructor

Creates a new [BinaryFunction](#) with the given programs as parameters.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected BinaryFunction(  
    ITreeProgram<double> firstProgram,  
    ITreeProgram<double> secondProgram  
)
```

[View Source](#)

#### Parameters

*firstProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first parameter of the program.

*secondProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second parameter of the program.

### See Also

[BinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction.BinaryFunction Properties

The [BinaryFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program.
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program.

### See Also

[BinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[BinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction.FirstParameter Property

Gets the first parameter of the program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> FirstParameter { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeProgram\(Double\)](#)

### See Also

[BinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[BinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction.SecondParameter Property

Gets the second parameter of the program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> SecondParameter { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeProgram\(Double\)](#)

### See Also

[BinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## BinaryFunction.BinaryFunction Methods

The [BinaryFunction](#) type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

	Name	Description
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[BinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)



## CommutativeBinaryFunction Class

Represents a [BinaryFunction](#) in which the order of the operands is not important. Therefore they are sorted to avoid creating repeated functions.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

Genetica.Elements.Functions.CommutativeBinaryFunction

[Genetica.Elements.Functions.AdditionFunction](#)

[Genetica.Elements.Functions.MaxFunction](#)

[Genetica.Elements.Functions.MinFunction](#)

[Genetica.Elements.Functions.MultiplicationFunction](#)

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public abstract class CommutativeBinaryFunction : BinaryFunction,  
    ICommutativeTreeProgram<double>, ITreeProgram<double>,  
    IProgram< IReadOnlyList<ITreeProgram<double>>, double>,  
    IProgram, ITreeProgram, ITreeNode, IComparable<ITreeProgram<double>>
```

[View Source](#)

The **CommutativeBinaryFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">CommutativeBinaryFunction</a>	Creates a new <b>CommutativeBinaryFunction</b> with the given programs as parameters.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)

 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
 <a href="#">CompareTo</a>		Compares this program with another program using a <code>string.CompareOrdinal</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>		Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">CreateNew</a>		Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(Object)</a>		(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>		Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>		(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>		Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>		Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">ToString</a>		(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
 <a href="#">ContainsConstant</a>		Verifies whether the <a href="#">MathProgram</a> contains a constant value, i.e., whether any one of its descendant programs are instances have a

		constant value equal to <i>val</i> . (Defined by <a href="#">MathProgramExtensions</a> .)
⬇️	<a href="#">ContainsSubElement(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">EqualsConstant</a>	Verifies whether the <a href="#">MathProgram</a> is a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> , and whether the associated value equals to <i>val</i> . (Defined by <a href="#">MathProgramExtensions</a> .)
⬇️	<a href="#">GetCommonRegionIndexes(Double)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetLeaves(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetMaxBreadth(Double)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetMaxDepth(Double)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <b>Terminal</b> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetPrimitives(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetRange</a>	Computes a <a href="#">Range</a> representing the minimum and maximum values that a given <a href="#">MathProgram</a> can compute, as dictated by its sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
⬇️	<a href="#">GetSubCombinations(Double)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations

		between the sub-programs of the children and also the sub-combinations of each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetSubPrograms(Double)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">GetValueRmsd</a>	Gets the root-mean-square deviation (RMSD) between the values computed for the given programs. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the square of the difference between the values computed by both programs is calculated. (Defined by <a href="#">MathProgramExtensions</a> .)
⬇️	<a href="#">IsConstant</a>	Verifies whether the <a href="#">MathProgram</a> has a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> . (Defined by <a href="#">MathProgramExtensions</a> .)
⬇️	<a href="#">IsLeaf(Double)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">IsSubProgramOf(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is a program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
⬇️	<a href="#">IsValueEquivalent</a>	Checks whether the given <a href="#">MathProgram</a> computes a value that is consistently equivalent to that computed by another program according to some margin. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the difference between the values computed by both programs is calculated. If the difference is less than a given margin for all the trials, then the programs are considered to be value-equivalent. (Defined by <a href="#">MathProgramExtensions</a> .)
⬇️	<a href="#">ProgramAt(Double)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">Replace(Double)(UInt32, ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Replace(Double)(ITreeProgram(Double), ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to <i>oldSubProgram</i> are replaced by <i>newSubProgram</i> . (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <i>margin</i> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## CommutativeBinaryFunction Constructor

Creates a new [CommutativeBinaryFunction](#) with the given programs as parameters.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected CommutativeBinaryFunction(  
    ITreeProgram<double> firstProgram,  
    ITreeProgram<double> secondProgram  
)
```

[View Source](#)

### Parameters

*firstProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first parameter of the program.

*secondProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second parameter of the program.

### See Also

[CommutativeBinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CommutativeBinaryFunction.CommutativeBinaryFunction Properties

The [CommutativeBinaryFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[CommutativeBinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CommutativeBinaryFunction.CommutativeBinaryFunction Methods

The [CommutativeBinaryFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareOrdinal</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">ContainsConstant</a>	Verifies whether the <a href="#">MathProgram</a> contains a constant value, i.e., whether any one of its descendant programs are instances have a constant value equal to <code>val</code> . (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">ContainsSubElement(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> contains the given sub-program. A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">EqualsConstant</a>	Verifies whether the <a href="#">MathProgram</a> is a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> , and whether the associated value equals to <i>val</i> . (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">GetCommonRegionIndexes(Double)</a>	Gets a <a href="#">IDictionary(TKey, TValue)</a> representing the program indexes in the common region between <i>program</i> and <i>otherProgram</i> . The dictionary represents the index correspondence between the sub-programs of the given programs. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetLeaves(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> leaves of the given program and their count. This corresponds to the leaf nodes of the expression tree of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxBreadth(Double)</a>	Gets the maximum breadth of the program, i.e., the number of <a href="#">ITreeProgram(TOutput)</a> leaves encountered starting from this program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetMaxDepth(Double)</a>	Gets the maximum depth of the program, i.e., the maximum number of child programs encountered starting from this program until reaching a <a href="#">Terminal</a> program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetPrimitives(Double)</a>	Gets a dictionary containing all the <a href="#">ITreeProgram(TOutput)</a> primitives in the given program and their count. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetRange</a>	Computes a <a href="#">Range</a> representing the minimum and maximum values that a given <a href="#">MathProgram</a> can compute, as dictated by its sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">GetSubCombinations(Double)</a>	Gets all the <a href="#">ITreeProgram(TOutput)</a> sub-combinations of the given program. If the program is a leaf, there are no combinations, otherwise returns all the possible combinations between the sub-programs of the children and also the sub-combinations of each child. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">GetSubPrograms(Double)</a>	Gets a <a href="#">ISet(T)</a> containing all the descendant <a href="#">ITreeProgram(TOutput)</a> of the given program. (Defined by <a href="#">TreeProgramExtensions</a> .)

 <a href="#">GetValueRmsd</a>	Gets the root-mean-square deviation (RMSD) between the values computed for the given programs. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the square of the difference between the values computed by both programs is calculated. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">IsConstant</a>	Verifies whether the <a href="#">MathProgram</a> has a constant value, i.e., whether all its descendant leaf programs are instances of <a href="#">Constant</a> . (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">IsLeaf(Double)</a>	Checks whether the given <a href="#">ITreeProgram(TOutput)</a> is a leaf node, i.e., it has no child nodes. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">IsSubProgramOf(Double)</a>	Checks whether a given <a href="#">ITreeProgram(TOutput)</a> is a sub-program of another <a href="#">ITreeProgram(TOutput)</a> . A sub-program is an program that is a descendant of a given program. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">IsValueEquivalent</a>	Checks whether the given <a href="#">MathProgram</a> computes a value that is consistently equivalent to that computed by another program according to some margin. The method works by substituting the variables in both programs' expressions by random values for a certain number of trials. In each trial, the difference between the values computed by both programs is calculated. If the difference is less than a given margin for all the trials, then the programs are considered to be value-equivalent. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">ProgramAt(Double)</a>	Get the <a href="#">ITreeProgram(TOutput)</a> at the given index, where programs are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Replace(Double)(UInt32, ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where the descendant program at the given index is replaced by <i>newSubProgram</i> . Elements are indexed in a zero-based, depth first search manner. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Replace(Double)(ITreeProgram(Double), ITreeProgram(Double))</a>	Overloaded. Gets a new copy of <i>program</i> where all sub-programs that are equal to

		oldSubProgram are replaced by newSubProgram. (Defined by <a href="#">TreeProgramExtensions</a> .)
 <a href="#">Simplify</a>		Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <i>margin</i> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)
 <a href="#">ToD3JsonFile</a>		Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>		Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>		Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>		Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[CommutativeBinaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CosineFunction Class

Represents a [UnaryFunction](#) performing the cosine operation over some sub-program.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.UnaryFunction](#)

Genetica.Elements.Functions.CosineFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class CosineFunction : UnaryFunction
```

[View Source](#)

The **CosineFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">CosineFunction</a>	Creates a new <b>CosineFunction</b> with the given parameter.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">UnaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">UnaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Operand</a>	Gets the parameter associated with this function. (Inherited from <a href="#">UnaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## CosineFunction Constructor

Creates a new [CosineFunction](#) with the given parameter.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public CosineFunction(  
    ITreeProgram<double> parameter  
)
```

[View Source](#)

#### Parameters

*parameter*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The parameter of the program.

### See Also

[CosineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CosineFunction.CosineFunction Properties

The [CosineFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">UnaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">UnaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Operand</a>	Gets the parameter associated with this function. (Inherited from <a href="#">UnaryFunction</a> .)

### See Also

[CosineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CosineFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[CosineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CosineFunction.CosineFunction Methods

The [CosineFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[CosineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CosineFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public override double Compute()
```

[View Source](#)

### Return Value

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

### Implements

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[CosineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## CosineFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[CosineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction Class

Represents a [BinaryFunction](#) performing the division between two sub-programs.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

Genetica.Elements.Functions.DivisionFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class DivisionFunction : BinaryFunction
```

[View Source](#)

The **DivisionFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">DivisionFunction</a>	Creates a new <b>DivisionFunction</b> .

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction Constructor

Creates a new [DivisionFunction](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public DivisionFunction(  
    ITreeProgram<double> numeratorProgram,  
    ITreeProgram<double> denominatorProgram  
)
```

[View Source](#)

#### Parameters

*numeratorProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the numerator value.

*denominatorProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the denominator value.

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction.DivisionFunction Properties

The [DivisionFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction.DivisionFunction Methods

The [DivisionFunction](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareOrdinal</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

	Name	Description
	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## DivisionFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[DivisionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction Class

Represents a [MathProgram](#) performing a conditional operation according to the value of the first sub-program.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

Genetica.Elements.Functions.IfFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class IfFunction : MathProgram
```

[View Source](#)

The **IfFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">IfFunction</a>	Creates a new <b>IfFunction</b> with the given sub-programs.

### Properties

	Name	Description
	<a href="#">ConditionProgram</a>	Gets the sub-program computing the conditional value.
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">NegativeProgram</a>	Gets the sub-program computing the value if the conditional is less than o.
	<a href="#">PositiveProgram</a>	Gets the sub-program computing the value if the conditional is greater than o.
	<a href="#">ZeroProgram</a>	Gets the sub-program computing the value if the conditional is equal to o.

## Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareOrdinal</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## IfFunction Constructor

Creates a new [IfFunction](#) with the given sub-programs.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IfFunction(  
    ITreeProgram<double> conditionProgram,  
    ITreeProgram<double> zeroProgram,  
    ITreeProgram<double> positiveProgram,  
    ITreeProgram<double> negativeProgram  
)
```

[View Source](#)

#### Parameters

*conditionProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the conditional value.

*zeroProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the value if the conditional is 0.

*positiveProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the value if the conditional is >0

*negativeProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the value if the conditional is <0

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.IfFunction Properties

The [IfFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">ConditionProgram</a>	Gets the sub-program computing the conditional value.
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">NegativeProgram</a>	Gets the sub-program computing the value if the conditional is less than 0.
 <a href="#">PositiveProgram</a>	Gets the sub-program computing the value if the conditional is greater than 0
 <a href="#">ZeroProgram</a>	Gets the sub-program computing the value if the conditional is equal to 0.

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.ConditionProgram Property

Gets the sub-program computing the conditional value.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> ConditionProgram { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeProgram\(Double\)](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.NegativeProgram Property

Gets the sub-program computing the value if the conditional is less than 0.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> NegativeProgram { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeProgram\(Double\)](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.PositiveProgram Property

Gets the sub-program computing the value if the conditional is greater than 0

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> PositiveProgram { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeProgram\(Double\)](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.ZeroProgram Property

Gets the sub-program computing the value if the conditional is equal to 0.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> ZeroProgram { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeProgram\(Double\)](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.IfFunction Methods

The [IfFunction](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

	Name	Description
	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## IfFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[IfFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## LogarithmFunction Class

Represents a [BinaryFunction](#) performing the logarithm operation.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

Genetica.Elements.Functions.LogarithmFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class LogarithmFunction : BinaryFunction
```

[View Source](#)

The **LogarithmFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">LogarithmFunction</a>	Creates a new <b>LogarithmFunction</b> with the given parameters.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## LogarithmFunction Constructor

Creates a new [LogarithmFunction](#) with the given parameters.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public LogarithmFunction(  
    ITreeProgram<double> valueProgram,  
    ITreeProgram<double> baseProgram  
)
```

[View Source](#)

#### Parameters

*valueProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the value of the logarithm.

*baseProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the base of the logarithm.

### See Also

[LogarithmFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## LogarithmFunction.LogarithmFunction Properties

The [LogarithmFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[LogarithmFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## LogarithmFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[LogarithmFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## LogarithmFunction.LogarithmFunction Methods

The [LogarithmFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[LogarithmFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## LogarithmFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[LogarithmFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## LogarithmFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[LogarithmFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MaxFunction Class

Represents a [CommutativeBinaryFunction](#) selecting the maximum between two sub-programs.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

[Genetica.Elements.Functions.CommutativeBinaryFunction](#)

Genetica.Elements.Functions.MaxFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class MaxFunction : CommutativeBinaryFunction
```

[View Source](#)

The **MaxFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">MaxFunction</a>	Creates a new <b>MaxFunction</b> .

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## MaxFunction Constructor

Creates a new [MaxFunction](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public MaxFunction(  
    ITreeProgram<double> firstProgram,  
    ITreeProgram<double> secondProgram  
)
```

[View Source](#)

#### Parameters

*firstProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first sub-program.

*secondProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second sub-program.

### See Also

[MaxFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MaxFunction.MaxFunction Properties

The [MaxFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[MaxFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MaxFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[MaxFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MaxFunction.MaxFunction Methods

The [MaxFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[MaxFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MaxFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[MaxFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MaxFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[MaxFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MaxFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[MaxFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MinFunction Class

Represents a [CommutativeBinaryFunction](#) selecting the minimum between two sub-programs.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

[Genetica.Elements.Functions.CommutativeBinaryFunction](#)

Genetica.Elements.Functions.MinFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class MinFunction : CommutativeBinaryFunction
```

[View Source](#)

The **MinFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">MinFunction</a>	Creates a new <b>MinFunction</b> .

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## MinFunction Constructor

Creates a new [MinFunction](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public MinFunction(  
    ITreeProgram<double> firstProgram,  
    ITreeProgram<double> secondProgram  
)
```

[View Source](#)

#### Parameters

*firstProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first sub-program.

*secondProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second sub-program.

### See Also

[MinFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MinFunction.MinFunction Properties

The [MinFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[MinFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MinFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

#### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[MinFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MinFunction.MinFunction Methods

The [MinFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[MinFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MinFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[MinFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MinFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[MinFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MinFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[MinFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction Class

Represents a [CommutativeBinaryFunction](#) performing the multiplication of two sub-programs.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

[Genetica.Elements.Functions.CommutativeBinaryFunction](#)

Genetica.Elements.Functions.MultiplicationFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class MultiplicationFunction : CommutativeBinaryFunction
```

[View Source](#)

The **MultiplicationFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">MultiplicationFunction</a>	Creates a new <b>MultiplicationFunction</b> .

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction Constructor

Creates a new [MultiplicationFunction](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public MultiplicationFunction(  
    ITreeProgram<double> firstProgram,  
    ITreeProgram<double> secondProgram  
)
```

[View Source](#)

#### Parameters

*firstProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first sub-program.

*secondProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second sub-program.

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction.MultiplicationFunction Properties

The [MultiplicationFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction.MultiplicationFunction Methods

The [MultiplicationFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## MultiplicationFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[MultiplicationFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction Class

Represents a [BinaryFunction](#) performing the exponentiation operation.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

Genetica.Elements.Functions.PowerFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class PowerFunction : BinaryFunction
```

[View Source](#)

The **PowerFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">PowerFunction</a>	Creates a new <b>PowerFunction</b> with the given parameters.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## PowerFunction Constructor

Creates a new [PowerFunction](#) with the given parameters.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public PowerFunction(  
    ITreeProgram<double> baseProgram,  
    ITreeProgram<double> exponentProgram  
)
```

[View Source](#)

#### Parameters

*baseProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the number to be raised to the power.

*exponentProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The sub-program computing the power.

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction.PowerFunction Properties

The [PowerFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction.PowerFunction Methods

The [PowerFunction](#) type exposes the following members.

### Methods

	Name	Description
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

	Name	Description
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## PowerFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[PowerFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SineFunction Class

Represents a [UnaryFunction](#) performing the sine operation over some sub-program.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.UnaryFunction](#)

Genetica.Elements.Functions.SineFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SineFunction : UnaryFunction
```

[View Source](#)

The **SineFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SineFunction</a>	Creates a new <b>SineFunction</b> with the given parameter.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">UnaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">UnaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Operand</a>	Gets the parameter associated with this function. (Inherited from <a href="#">UnaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## SineFunction Constructor

Creates a new [SineFunction](#) with the given parameter.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SineFunction(  
    ITreeProgram<double> parameter  
)
```

[View Source](#)

#### Parameters

parameter

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The parameter of the program.

#### See Also

[SineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SineFunction.SineFunction Properties

The [SineFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">UnaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">UnaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Operand</a>	Gets the parameter associated with this function. (Inherited from <a href="#">UnaryFunction</a> .)

### See Also

[SineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SineFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[SineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SineFunction.SineFunction Methods

The [SineFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[SineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SineFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[SineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SineFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[SineFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction Class

Represents a [BinaryFunction](#) performing the subtraction operation between two sub-programs.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.BinaryFunction](#)

Genetica.Elements.Functions.SubtractionFunction

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SubtractionFunction : BinaryFunction
```

[View Source](#)

The **SubtractionFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SubtractionFunction</a>	Creates a new <b>SubtractionFunction</b> .

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
	<a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
≡	<a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction Constructor

Creates a new [SubtractionFunction](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SubtractionFunction(  
    ITreeProgram<double> firstProgram,  
    ITreeProgram<double> secondProgram  
)
```

[View Source](#)

#### Parameters

*firstProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The first sub-program.

*secondProgram*

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The second sub-program.

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction.SubtractionFunction Properties

The [SubtractionFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">BinaryFunction.Expression</a> .)
 <a href="#">FirstParameter</a>	Gets the first parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">BinaryFunction.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">SecondParameter</a>	Gets the second parameter of the program. (Inherited from <a href="#">BinaryFunction</a> .)

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction.SubtractionFunction Methods

The [SubtractionFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Overrides <a href="#">MathProgram.CreateNew(ITreeProgram(Double))</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Overrides <a href="#">MathProgram.Simplify()</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction.CreateNew Method

Creates a new [MathProgram](#) which is a copy of this program.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

The children of the new program.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

A new [MathProgram](#) of the same kind of this program with the given child programs.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## SubtractionFunction.Simplify Method

Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

A program corresponding to a simplification of the given program.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[SubtractionFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## UnaryFunction Class

Represents a [MathProgram](#) whose result is a mathematical operation involving a single input sub-program.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Functions.UnaryFunction](#)

[Genetica.Elements.Functions.CosineFunction](#)

[Genetica.Elements.Functions.SineFunction](#)

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract class UnaryFunction : MathProgram
```

[View Source](#)

The **UnaryFunction** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">UnaryFunction</a>	Creates a new <b>UnaryFunction</b> with the given parameter.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Operand</a>	Gets the parameter associated with this function.

## Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Functions Namespace](#)

## UnaryFunction Constructor

Creates a new [UnaryFunction](#) with the given parameter.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected UnaryFunction(  
    ITreeProgram<double> parameter  
)
```

[View Source](#)

#### Parameters

parameter

Type: [Genetica.Elements.ITreeProgram\(Double\)](#)

The parameter of the program.

### See Also

[UnaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## UnaryFunction.UnaryFunction Properties

The [UnaryFunction](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Operand</a>	Gets the parameter associated with this function.

### See Also

[UnaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## UnaryFunction.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[UnaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## UnaryFunction.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[UnaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## UnaryFunction.Operand Property

Gets the parameter associated with this function.

**Namespace:** [Genetica.Elements.Functions](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ITreeProgram<double> Operand { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeProgram\(Double\)](#)

### See Also

[UnaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)

## UnaryFunction.UnaryFunction Methods

The [UnaryFunction](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">CreateNew</a>	Creates a new <a href="#">MathProgram</a> which is a copy of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Simplifies the expression of the given program by returning a new program which value will always be equal to the original program, i.e. it removes redundant operations. Implementations can e.g. remove subtrees that will always result in the value of one of its operands. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[UnaryFunction Class](#)

[Genetica.Elements.Functions Namespace](#)



## Genetica.Elements.Terminals Namespace

### Classes

	Class	Description
	<a href="#">Constant</a>	Represents a <a href="#">Terminal</a> program whose output value does not change.
	<a href="#">Terminal</a>	A terminal represents a <a href="#">MathProgram</a> whose output value depends only on itself. It can be used as a leaf node in a program's tree, i.e., it has no children/input.
	<a href="#">Variable</a>	Represents a mathematical variable represented by some label and whose value is in some given <a href="#">Range</a> .

### Structures

	Structure	Description
	<a href="#">Range</a>	Allows the definition of a range of <a href="#">Double</a> values by setting the minimum and maximum values allowed.

## Constant Class

Represents a [Terminal](#) program whose output value does not change.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Terminals.Terminal](#)

Genetica.Elements.Terminals.Constant

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class Constant : Terminal
```

[View Source](#)

The **Constant** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">Constant</a>	Creates a new <b>Constant</b> with the given value.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">Terminal</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)

## Methods

	Name	Description
💡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
💡	<a href="#">CreateNew</a>	Because terminals have no children, this returns the object itself. (Inherited from <a href="#">Terminal</a> .)
💡	<a href="#">Equals(Object)</a>	(Overrides <a href="#">Terminal.Equals(Object)</a> .)
💡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">GetHashCode</a>	(Overrides <a href="#">Terminal.GetHashCode()</a> .)
💡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
💡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">Simplify</a>	Because terminals have no children, the program cannot be simplified so this method returns the object itself. (Inherited from <a href="#">Terminal</a> .)
💡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Fields

	Name	Description
💡	<a href="#">One</a>	Returns a <a href="#">Constant</a> with a value of 1.
💡	<a href="#">Zero</a>	Returns a <a href="#">Constant</a> with a value of 0.

## Extension Methods

	Name	Description
💡	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Terminals Namespace](#)



## Constant Constructor

Creates a new [Constant](#) with the given value.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public Constant(  
    double val  
)
```

[View Source](#)

#### Parameters

*val*

Type: [System.Double](#)

The value of this constant.

#### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Constant Properties

The [Constant](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">Terminal</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[ITreeProgram.Label](#)

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Constant Methods

The [Constant](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Because terminals have no children, this returns the object itself. (Inherited from <a href="#">Terminal</a> .)
 <a href="#">Equals(Object)</a>	(Overrides <a href="#">Terminal.Equals(Object)</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Overrides <a href="#">Terminal.GetHashCode()</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Because terminals have no children, the program cannot be simplified so this method returns the object itself. (Inherited from <a href="#">Terminal</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Equals Method

### Overload List

	Name	Description
	<a href="#">Equals(Object)</a>	(Overrides <a href="#">Terminal.Equals(Object)</a> .)
	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Equals Method (Object)

[Missing <summary> documentation for  
"M:Genetica.Elements.Terminals.Constant.Equals(System.Object)"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override bool Equals(  
    Object obj  
)
```

[View Source](#)

### Parameters

obj

Type: [System.Object](#)

[Missing <param name="obj"/> documentation for  
"M:Genetica.Elements.Terminals.Constant.Equals(System.Object)"]

### Return Value

Type: [Boolean](#)

[Missing <returns> documentation for  
"M:Genetica.Elements.Terminals.Constant.Equals(System.Object)"]

### See Also

[Constant Class](#)

[Equals Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.GetHashCode Method

[Missing <summary> documentation for "M:Genetica.Elements.Terminals.Constant.GetHashCode"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override int GetHashCode()
```

[View Source](#)

### Return Value

Type: [Int32](#)

[Missing <returns> documentation for "M:Genetica.Elements.Terminals.Constant.GetHashCode"]

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Constant Fields

The [Constant](#) type exposes the following members.

### Fields

	Name	Description
 	<a href="#">One</a>	Returns a <a href="#">Constant</a> with a value of 1.
 	<a href="#">Zero</a>	Returns a <a href="#">Constant</a> with a value of 0.

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.One Field

Returns a [Constant](#) with a value of 1.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static readonly Constant One
```

[View Source](#)

#### *Field Value*

Type: [Constant](#)

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Constant.Zero Field

Returns a [Constant](#) with a value of 0.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static readonly Constant Zero
```

[View Source](#)

#### *Field Value*

Type: [Constant](#)

### See Also

[Constant Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Range Structure

Allows the definition of a range of [Double](#) values by setting the minimum and maximum values allowed.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public struct Range : IEquatable<Range>
```

[View Source](#)

The **Range** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">Range(Double)</a>	Creates a new <b>Range</b> corresponding to some constant value.
	<a href="#">Range(Double, Double)</a>	Creates a new <b>Range</b> with the given minimum and maximum values.

### Properties

	Name	Description
	<a href="#">Interval</a>	Gets the difference between the minimum and maximum values.
	<a href="#">Max</a>	Gets or sets the maximum value allowed in this range.
	<a href="#">Min</a>	Gets or sets the minimum value allowed in this range.

### Methods

	Name	Description
	<a href="#">Equals(Object)</a>	(Overrides <a href="#">ValueType.Equals(Object)</a> .)
	<a href="#">Equals(Range)</a>	
	<a href="#">GetHashCode</a>	(Overrides <a href="#">ValueType.GetHashCode()</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Overrides <a href="#">ValueType.ToString()</a> .)

### Operators

	Name	Description
	<a href="#">Equality</a>	Checks whether two <b>Range</b> objects are equal, i.e., whether their <a href="#">Max</a> and <a href="#">Min</a> properties are equal.

	<a href="#">Inequality</a>	Checks whether two <b>Range</b> objects are not equal, i.e., whether their <a href="#">Max</a> or <a href="#">Min</a> properties are not equal.
---	----------------------------	---

## Fields

	Name	Description
	<a href="#">Default</a>	Returns the default value for the full <a href="#">Double</a> range.

## See Also

[Genetica.Elements.Terminals Namespace](#)

## Range Constructor

### Overload List

	Name	Description
	<a href="#">Range(Double)</a>	Creates a new <a href="#">Range</a> corresponding to some constant value.
	<a href="#">Range(Double, Double)</a>	Creates a new <a href="#">Range</a> with the given minimum and maximum values.

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range Constructor (Double)

Creates a new [Range](#) corresponding to some constant value.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public Range(  
    double constValue  
)
```

[View Source](#)

#### Parameters

*constValue*

Type: [System.Double](#)

The minimum and maximum of this range.

### See Also

[Range Structure](#)

[Range Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Range Constructor (Double, Double)

Creates a new [Range](#) with the given minimum and maximum values.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public Range(  
    double min,  
    double max  
)
```

[View Source](#)

### Parameters

*min*

Type: [System.Double](#)

The minimum value allowed in this range.

*max*

Type: [System.Double](#)

The maximum value allowed in this range.

### See Also

[Range Structure](#)

[Range Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Range Properties

The [Range](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Interval</a>	Gets the difference between the minimum and maximum values.
	<a href="#">Max</a>	Gets or sets the maximum value allowed in this range.
	<a href="#">Min</a>	Gets or sets the minimum value allowed in this range.

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Interval Property

Gets the difference between the minimum and maximum values.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Interval { get; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Max Property

Gets or sets the maximum value allowed in this range.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Max { get; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Min Property

Gets or sets the minimum value allowed in this range.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Min { get; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Range Methods

The [Range](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Equals(Object)</a>	(Overrides <a href="#">ValueType.Equals(Object)</a> .)
	<a href="#">Equals(Range)</a>	
	<a href="#">GetHashCode</a>	(Overrides <a href="#">ValueType.GetHashCode()</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Overrides <a href="#">ValueType.ToString()</a> .)

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Equals Method

### Overload List

	Name	Description
	<a href="#">Equals(Object)</a>	(Overrides <a href="#">ValueType.Equals(Object)</a> .)
	<a href="#">Equals(Range)</a>	

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Equals Method (Object)

[Missing <summary> documentation for  
"M:Genetica.Elements.Terminals.Range.Equals(System.Object)"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override bool Equals(  
    Object obj  
)
```

[View Source](#)

### Parameters

obj

Type: [System.Object](#)

[Missing <param name="obj"/> documentation for  
"M:Genetica.Elements.Terminals.Range.Equals(System.Object)"]

### Return Value

Type: [Boolean](#)

[Missing <returns> documentation for  
"M:Genetica.Elements.Terminals.Range.Equals(System.Object)"]

### See Also

[Range Structure](#)

[Equals Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Equals Method (Range)

[Missing <summary> documentation for  
"M:Genetica.Elements.Terminals.Range.Equals(Genetica.Elements.Terminals.Range)"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public bool Equals(  
    Range other  
)
```

[View Source](#)

### Parameters

other

Type: [Genetica.Elements.Terminals.Range](#)

[Missing <param name="other"/> documentation for  
"M:Genetica.Elements.Terminals.Range.Equals(Genetica.Elements.Terminals.Range)"]

### Return Value

Type: [Boolean](#)

[Missing <returns> documentation for  
"M:Genetica.Elements.Terminals.Range.Equals(Genetica.Elements.Terminals.Range)"]

### Implements

[IEquatable\(T\).Equals\(T\)](#)

### See Also

[Range Structure](#)

[Equals Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.GetHashCode Method

[Missing <summary> documentation for "M:Genetica.Elements.Terminals.Range.GetHashCode"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override int GetHashCode()
```

[View Source](#)

### Return Value

Type: [Int32](#)

[Missing <returns> documentation for "M:Genetica.Elements.Terminals.Range.GetHashCode"]

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.ToString Method

[Missing <summary> documentation for "M:Genetica.Elements.Terminals.Range.ToString"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string ToString()
```

[View Source](#)

**Return Value**

Type: [String](#)

[Missing <returns> documentation for "M:Genetica.Elements.Terminals.Range.ToString"]

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Range Operators

The [Range](#) type exposes the following members.

### Operators

	Name	Description
 	<a href="#">Equality</a>	Checks whether two <a href="#">Range</a> objects are equal, i.e., whether their <a href="#">Max</a> and <a href="#">Min</a> properties are equal.
 	<a href="#">Inequality</a>	Checks whether two <a href="#">Range</a> objects are not equal, i.e., whether their <a href="#">Max</a> or <a href="#">Min</a> properties are not equal.

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Equality Operator

Checks whether two [Range](#) objects are equal, i.e., whether their [Max](#) and [Min](#) properties are equal.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public static bool operator ==(  
    Range left,  
    Range right  
)
```

[View Source](#)

### Parameters

*left*

Type: [Genetica.Elements.Terminals.Range](#)

The first range.

*right*

Type: [Genetica.Elements.Terminals.Range](#)

The second range.

### Return Value

Type: [Boolean](#)

*true* if the ranges are equal, *false* otherwise.

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Inequality Operator

Checks whether two [Range](#) objects are not equal, i.e., whether their [Max](#) or [Min](#) properties are not equal.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static bool operator !=(
    Range left,
    Range right
)
```

[View Source](#)

### Parameters

*left*

Type: [Genetica.Elements.Terminals.Range](#)

The first range.

*right*

Type: [Genetica.Elements.Terminals.Range](#)

The second range.

### Return Value

Type: [Boolean](#)

true if the ranges are not equal, false otherwise.

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Range Fields

The [Range](#) type exposes the following members.

### Fields

	Name	Description
 <b>s</b>	<a href="#">Default</a>	Returns the default value for the full <a href="#">Double</a> range.

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Range.Default Field

Returns the default value for the full [Double](#) range.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static Range Default
```

[View Source](#)

#### *Field Value*

Type: [Range](#)

### See Also

[Range Structure](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal Class

A terminal represents a [MathProgram](#) whose output value depends only on itself. It can be used as a leaf node in a program's tree, i.e., it has no children/input.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

Genetica.Elements.Terminals.Terminal

[Genetica.Elements.Terminals.Constant](#)

[Genetica.Elements.Terminals.Variable](#)

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract class Terminal : MathProgram
```

[View Source](#)

The **Terminal** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">Terminal</a>	Creates a new <b>Terminal</b> .

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)

## Methods

	<b>Name</b>	<b>Description</b>
≡	<a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">CreateNew</a>	Because terminals have no children, this returns the object itself. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
≡	<a href="#">Equals(Object)</a>	(Overrides <a href="#">MathProgram.Equals(Object)</a> .)
≡	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Overrides <a href="#">MathProgram.GetHashCode()</a> .)
≡	<a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Simplify</a>	Because terminals have no children, the program cannot be simplified so this method returns the object itself. (Overrides <a href="#">MathProgram.Simplify()</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

	<b>Name</b>	<b>Description</b>
⬇	<a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Terminals Namespace](#)

## Terminal Constructor

Creates a new [Terminal](#).

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected Terminal()
```

[View Source](#)

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.Terminal Properties

The [Terminal](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Overrides <a href="#">MathProgram.Expression</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.Expression Property

Gets this program's expression in normal form by combining the [Labels](#) of all descendant programs encountered starting from this program (the root).

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Expression { get; }
```

[View Source](#)

### Property Value

Type: [String](#)

Implements

[IProgram.Expression](#)

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.Terminal Methods

The [Terminal](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">CreateNew</a>	Because terminals have no children, this returns the object itself. (Overrides <a href="#">MathProgram.CreateNew(IList(ITreeProgram(Double)))</a> .)
 <a href="#">Equals(Object)</a>	(Overrides <a href="#">MathProgram.Equals(Object)</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Overrides <a href="#">MathProgram.GetHashCode()</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Because terminals have no children, the program cannot be simplified so this method returns the object itself. (Overrides <a href="#">MathProgram.Simplify()</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.CreateNew Method

Because terminals have no children, this returns the object itself.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> CreateNew(  
    IList<ITreeProgram<double>> children  
)
```

[View Source](#)

#### Parameters

*children*

Type: [System.Collections.Generic.IList\(ITreeProgram\(Double\)\)](#)

This parameter will not be used, so it may be null.

#### Return Value

Type: [ITreeProgram\(Double\)](#)

The same [Terminal](#) object.

#### Implements

[ITreeProgram\(TOutput\).CreateNew\(IList\(ITreeProgram\(TOutput\)\)\)](#)

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.Equals Method

### Overload List

	Name	Description
	<a href="#">Equals(Object)</a>	(Overrides <a href="#">MathProgram.Equals(Object)</a> .)
	<a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.Equals Method (Object)

[Missing <summary> documentation for  
"M:Genetica.Elements.Terminals.Terminal.Equals(System.Object)"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override bool Equals(  
    Object obj  
)
```

[View Source](#)

### Parameters

obj

Type: [System.Object](#)

[Missing <param name="obj"/> documentation for  
"M:Genetica.Elements.Terminals.Terminal.Equals(System.Object)"]

### Return Value

Type: [Boolean](#)

[Missing <returns> documentation for  
"M:Genetica.Elements.Terminals.Terminal.Equals(System.Object)"]

### See Also

[Terminal Class](#)

[Equals Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.GetHashCode Method

[Missing <summary> documentation for "M:Genetica.Elements.Terminals.Terminal.GetHashCode"]

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override int GetHashCode()
```

[View Source](#)

### Return Value

Type: [Int32](#)

[Missing <returns> documentation for "M:Genetica.Elements.Terminals.Terminal.GetHashCode"]

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Terminal.Simplify Method

Because terminals have no children, the program cannot be simplified so this method returns the object itself.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override ITreeProgram<double> Simplify()
```

[View Source](#)

### Return Value

Type: [ITreeProgram\(Double\)](#)

The same [Terminal](#) object.

### Implements

[ITreeProgram\(TOutput\).Simplify\(\)](#)

### See Also

[Terminal Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable Class

Represents a mathematical variable represented by some label and whose value is in some given [Range](#).

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Elements.MathProgram](#)

[Genetica.Elements.Terminals.Terminal](#)

Genetica.Elements.Terminals.Variable

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class Variable : Terminal
```

[View Source](#)

The **Variable** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">Variable(String)</a>	Creates a new <b>Variable</b> with the given label and default range.
	<a href="#">Variable(String, Range)</a>	Creates a new <b>Variable</b> with the given label and range.
	<a href="#">Variable(String, Double, Range)</a>	Creates a new <b>Variable</b> with the given label, range and initial value.

### Properties

	Name	Description
	<a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">Terminal</a> .)
	<a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
	<a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
	<a href="#">Range</a>	Gets the minimum and maximum value expected for this variable.

	<a href="#">Value</a>	Gets or sets the value associated with this variable.
---	-----------------------	---

## Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareOrdinal</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Because terminals have no children, this returns the object itself. (Inherited from <a href="#">Terminal</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">Terminal</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Terminal</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Because terminals have no children, the program cannot be simplified so this method returns the object itself. (Inherited from <a href="#">Terminal</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

## Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

## See Also

[Genetica.Elements.Terminals Namespace](#)

## Variable Constructor

### Overload List

Name	Description
<a href="#">Variable(String)</a>	Creates a new <a href="#">Variable</a> with the given label and default range.
<a href="#">Variable(String, Range)</a>	Creates a new <a href="#">Variable</a> with the given label and range.
<a href="#">Variable(String, Double, Range)</a>	Creates a new <a href="#">Variable</a> with the given label, range and initial value.

### See Also

[Variable Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable Constructor (String)

Creates a new [Variable](#) with the given label and default range.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public Variable(  
    string label  
)
```

[View Source](#)

#### Parameters

*label*

Type: [System.String](#)

The label associated with the variable.

#### See Also

[Variable Class](#)

[Variable Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable Constructor (String, Range)

Creates a new [Variable](#)with the given label and range.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public Variable(  
    string label,  
    Range range  
)
```

[View Source](#)

### Parameters

*label*

Type: [System.String](#)

The label associated with the variable.

*range*

Type: [Genetica.Elements.Terminals.Range](#)

The range of possible values for the variable.

### See Also

[Variable Class](#)

[Variable Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable Constructor (String, Double, Range)

Creates a new [Variable](#)with the given label, range and initial value.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public Variable(  
    string label,  
    double value,  
    Range range  
)
```

[View Source](#)

### Parameters

*label*

Type: [System.String](#)

The label associated with the variable.

*value*

Type: [System.Double](#)

The initial value of the variable.

*range*

Type: [Genetica.Elements.Terminals.Range](#)

The range of possible values for the variable.

### See Also

[Variable Class](#)

[Variable Overload](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable.Variable Properties

The [Variable](#) type exposes the following members.

### Properties

Name	Description
 <a href="#">Expression</a>	Gets this program's expression in normal form by combining the <a href="#">Labels</a> of all descendant programs encountered starting from this program (the root). (Inherited from <a href="#">Terminal</a> .)
 <a href="#">Input</a>	Gets the children of this program, i.e., the direct <a href="#">MathProgram</a> descendants of this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Label</a>	Gets the program's label, i.e., its representation independently of its descendant nodes. (Overrides <a href="#">MathProgram.Label</a> .)
 <a href="#">Length</a>	Gets the program's length, i.e., the total number of <a href="#">MathProgram</a> descendant programs encountered starting from this program. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Range</a>	Gets the minimum and maximum value expected for this variable.
 <a href="#">Value</a>	Gets or sets the value associated with this variable.

### See Also

[Variable Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable.Label Property

Gets the program's label, i.e., its representation independently of its descendant nodes.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string Label { get; }
```

[View Source](#)

*Property Value*

Type: [String](#)

*Implements*

[ITreeProgram.Label](#)

### See Also

[Variable Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable.Range Property

Gets the minimum and maximum value expected for this variable.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public Range Range { get; }
```

[View Source](#)

**Property Value**

Type: [Range](#)

### See Also

[Variable Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable.Value Property

Gets or sets the value associated with this variable.

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Value { get; set; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[Variable Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable.Variable Methods

The [Variable](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">CompareTo</a>	Compares this program with another program using a <code>string.CompareTo</code> comparison over their <a href="#">Expression</a> . (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Compute</a>	Computes the output of the program based on this program's mathematical expression and its <a href="#">Input</a> . (Overrides <a href="#">MathProgram.Compute()</a> .)
 <a href="#">CreateNew</a>	Because terminals have no children, this returns the object itself. (Inherited from <a href="#">Terminal</a> .)
 <a href="#">Equals(Object)</a>	(Inherited from <a href="#">Terminal</a> .)
 <a href="#">Equals(MathProgram)</a>	Checks whether this program is equal to another. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Terminal</a> .)
 <a href="#">GetPrimitive</a>	Gets a program node representing the primitive of the corresponding <a href="#">MathProgram</a> derived type. For functions this corresponds to a <a href="#">MathProgram</a> whose inputs are <a href="#">Zero</a> , for terminals it returns the terminal itself. (Inherited from <a href="#">MathProgram</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Simplify</a>	Because terminals have no children, the program cannot be simplified so this method returns the object itself. (Inherited from <a href="#">Terminal</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">MathProgram</a> .)

### Extension Methods

Name	Description
 <a href="#">Simplify</a>	Simplifies the expression of the given <a href="#">MathProgram</a> by returning a sub-combination whose fitness is approximately equal to that of the original program by a margin of <code>margin</code> , and whose expression is naturally simpler, i.e., has fewer sub-programs. (Defined by <a href="#">MathProgramExtensions</a> .)

### See Also

[Variable Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Variable.Compute Method

Computes the output of the program based on this program's mathematical expression and its [Input](#).

**Namespace:** [Genetica.Elements.Terminals](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Compute()
```

[View Source](#)

**Return Value**

Type: [Double](#)

A value corresponding to the execution of this program's mathematical expression and its [Input](#).

[Implements](#)

[IProgram\(TInput, TOutput\).Compute\(\)](#)

### See Also

[Variable Class](#)

[Genetica.Elements.Terminals Namespace](#)

## Genetica.Evaluation Namespace

### Classes

Class	Description
 <a href="#">LinearRankFitnessFunction(TProgram)</a>	Represents a <a href="#">RankFitnessFunction(TProgram)</a> that computes the fitness of a program directly according to its rank.
 <a href="#">NonLinearRankFitnessFunction(TProgram)</a>	Calculates a transformed fitness based on a polynomial ranking. It implements the function in [1].
 <a href="#">RankFitnessFunction(TProgram)</a>	Represents an <a href="#">IFitnessFunction(TProgram)</a> that computes a fitness based on the relative ranking of elements as given by some <a href="#">IComparer(T)</a> .

### Interfaces

Interface	Description
 <a href="#">IFitnessFunction(TProgram)</a>	Represents an interface for methods of evaluating the fitness of <a href="#">IProgram(TInput, TOutput)</a> .

## IFitnessFunction(TProgram) Interface

Represents an interface for methods of evaluating the fitness of [IProgram\(TInput, TOutput\)](#).

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IFitnessFunction<in TProgram> : IComparer<TProgram>
where TProgram : IProgram
```

[View Source](#)

#### Type Parameters

**TProgram**

The type of program.

The IFitnessFunction(TProgram) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Compare</a>	(Inherited from <a href="#">IComparer(TProgram)</a> .)
	<a href="#">Evaluate</a>	Evaluates the fitness of the given program.

### See Also

[Genetica.Evaluation Namespace](#)

## [IFitnessFunction\(TProgram\).IFitnessFunction\(TProgram\) Methods](#)

The [IFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Compare</a>	(Inherited from <a href="#">IComparer(TProgram)</a> .)
	<a href="#">Evaluate</a>	Evaluates the fitness of the given program.

### See Also

[IFitnessFunction\(TProgram\)Interface](#)

[Genetica.Evaluation Namespace](#)

## [IFitnessFunction\(TProgram\).Evaluate Method](#)

Evaluates the fitness of the given program.

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
double Evaluate(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be evaluated.

#### Return Value

Type: [Double](#)

A score value denoting the fitness of the given program.

### See Also

[IFitnessFunction\(TProgram\)Interface](#)

[Genetica.Evaluation Namespace](#)

## LinearRankFitnessFunction(TProgram) Class

Represents a [RankFitnessFunction\(TProgram\)](#) that computes the fitness of a program directly according to its rank.

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Evaluation.RankFitnessFunction\(TProgram\)](#)

Genetica.Evaluation.LinearRankFitnessFunction(TProgram)

[Genetica.Evaluation.NonLinearRankFitnessFunction\(TProgram\)](#)

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class LinearRankFitnessFunction<TProgram> : RankFitnessFunction<TProgram>
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

The LinearRankFitnessFunction(TProgram) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">LinearRankFitnessFunction(TProgram)</a>	Creates a new LinearRankFitnessFunction(TProgram) with the given parameters.

### Properties

	Name	Description
	<a href="#">SelectivePressure</a>	Gets or sets the "probability of the best individual being selected compared to the average probability of selection of all individuals". The value should be in [1.0, 2.0]. (Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

### Methods

	Name	Description
	<a href="#">Compare</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Evaluate</a>	Evaluates the fitness of the given program. (Overrides <a href="#">RankFitnessFunction(TProgram).Evaluate(TProgram)</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Fields

	Name	Description
 <a href="#">individualRankings</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)	

## Remarks

[http://www.pohlheim.com/Papers/mpga\\_gal95/gal2\\_3.html](http://www.pohlheim.com/Papers/mpga_gal95/gal2_3.html)

## See Also

[Genetica.Evaluation Namespace](#)

## LinearRankFitnessFunction(TProgram) Constructor

Creates a new [LinearRankFitnessFunction\(TProgram\)](#) with the given parameters.

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public LinearRankFitnessFunction(  
    IComparer<TProgram> programComparer,  
    IPopulation<TProgram> population  
)
```

[View Source](#)

### Parameters

*programComparer*

Type: [System.Collections.Generic.IComparer\(TProgram\)](#)

The object used to compare programs and determine their ranking.

*population*

Type: [Genetica.IPopulation\(TProgram\)](#)

The list of programs whose fitness can be computed by this function.

### See Also

[LinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## LinearRankFitnessFunction(TProgram).LinearRankFitnessFunction(TProgram) Properties

The [LinearRankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">SelectivePressure</a>	Gets or sets the "probability of the best individual being selected compared to the average probability of selection of all individuals". The value should be in [1.0, 2.0]. (Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

### See Also

[LinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## LinearRankFitnessFunction(TProgram).LinearRankFitnessFunction(TProgram) Methods

The [LinearRankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Compare</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Evaluate</a>	Evaluates the fitness of the given program. (Overrides <a href="#">RankFitnessFunction(TProgram).Evaluate(TProgram)</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[LinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## LinearRankFitnessFunction(TProgram).Evaluate Method

Evaluates the fitness of the given program.

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Evaluate(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be evaluated.

#### Return Value

Type: [Double](#)

A score value denoting the fitness of the given program.

#### Implements

[IFitnessFunction\(TProgram\).Evaluate\(TProgram\)](#)

### See Also

[LinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## LinearRankFitnessFunction(TProgram).LinearRankFitnessFunction(TProgram) Fields

The [LinearRankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Fields

	Name	Description
	<a href="#">individualRankings</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

### See Also

[LinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## NonLinearRankFitnessFunction(TProgram) Class

Calculates a transformed fitness based on a polynomial ranking. It implements the function in [1].

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Evaluation.RankFitnessFunction\(TProgram\)](#)

[Genetica.Evaluation.LinearRankFitnessFunction\(TProgram\)](#)

Genetica.Evaluation.NonLinearRankFitnessFunction(TProgram)

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class NonLinearRankFitnessFunction<TProgram> : LinearRankFitnessFunction<TProgram>
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

TProgram

[Missing <typeparam name="TProgram"/> documentation for  
"T:Genetica.Evaluation.NonLinearRankFitnessFunction`1"]

The NonLinearRankFitnessFunction(TProgram) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">NonLinearRankFitnessFunction(TProgram)</a>	Creates a new NonLinearRankFitnessFunction(TProgram) with the given parameters.

### Properties

	Name	Description
	<a href="#">SelectivePressure</a>	Gets or sets the "probability of the best individual being selected compared to the average probability of selection of all individuals". The value should be in [1.0, 2.0]. (Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

### Methods

	Name	Description
	<a href="#">Compare</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Evaluate</a>	Evaluates the fitness of the given program. (Overrides <a href="#">LinearRankFitnessFunction(TProgram).Evaluate(TProgram)</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Fields

	Name	Description
 <a href="#">individualRankings</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)	

## Remarks

[1] - Pohlheim, H. (1995). The multipopulation genetic algorithm: Local selection and migration. Technical report, Technical University Ilmenau.  
[http://www.pohlheim.com/Papers/mpga\\_gal95/gal2\\_3.html](http://www.pohlheim.com/Papers/mpga_gal95/gal2_3.html)[http://www.geatbx.com/docu/algindex-02.html#P249\\_16387](http://www.geatbx.com/docu/algindex-02.html#P249_16387)

## See Also

[Genetica.Evaluation Namespace](#)

## NonLinearRankFitnessFunction(TProgram) Constructor

Creates a new [NonLinearRankFitnessFunction\(TProgram\)](#) with the given parameters.

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public NonLinearRankFitnessFunction(  
    IComparer<TProgram> programComparer,  
    IPopulation<TProgram> population,  
    double selectivePressure  
)
```

[View Source](#)

### Parameters

*programComparer*

Type: [System.Collections.Generic.IComparer\(TProgram\)](#)

The object used to compare programs and determine their ranking.

*population*

Type: [Genetica.IPopulation\(TProgram\)](#)

The list of programs whose fitness can be computed by this function.

*selectivePressure*

Type: [System.Double](#)

The probability of the best individual being selected compared to the average probability of selection of all individuals.

### See Also

[NonLinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## NonLinearRankFitnessFunction(TProgram).NonLinearRankFitnessFunction(TProgram) Properties

The [NonLinearRankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">SelectivePressure</a>	Gets or sets the "probability of the best individual being selected compared to the average probability of selection of all individuals". The value should be in [1.0, 2.0]. (Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

### See Also

[NonLinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## NonLinearRankFitnessFunction(TProgram).NonLinearRankFitnessFunction(TProgram) Methods

The [NonLinearRankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Compare</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Evaluate</a>	Evaluates the fitness of the given program. (Overrides <a href="#">LinearRankFitnessFunction(TProgram).Evaluate(TProgram)</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[NonLinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## NonLinearRankFitnessFunction(TProgram).Evaluate Method

Evaluates the fitness of the given program.

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override double Evaluate(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be evaluated.

#### Return Value

Type: [Double](#)

A score value denoting the fitness of the given program.

#### Implements

[IFitnessFunction\(TProgram\).Evaluate\(TProgram\)](#)

### See Also

[NonLinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## NonLinearRankFitnessFunction(TProgram).NonLinearRankFitnessFunction(TProgram) Fields

The [NonLinearRankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Fields

	Name	Description
	<a href="#">individualRankings</a>	(Inherited from <a href="#">RankFitnessFunction(TProgram)</a> .)

### See Also

[NonLinearRankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram) Class

Represents an [IFitnessFunction\(TProgram\)](#) that computes a fitness based on the relative ranking of elements as given by some [IComparer\(T\)](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Evaluation.RankFitnessFunction(TProgram)  
[Genetica.Evaluation.LinearRankFitnessFunction\(TProgram\)](#)

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract class RankFitnessFunction<TProgram> : IFitnessFunction<TProgram>,
    IComparer<TProgram>
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

*TProgram*

The type of program.

The RankFitnessFunction(TProgram) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">RankFitnessFunction(TProgram)</a>	Creates a new RankFitnessFunction(TProgram) with the given parameters.

### Properties

	Name	Description
	<a href="#">SelectivePressure</a>	Gets or sets the "probability of the best individual being selected compared to the average probability of selection of all individuals". The value should be in [1.0, 2.0].

### Methods

	Name	Description
	<a href="#">Compare</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">Evaluate</a>	Evaluates the fitness of the given program.
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Fields

	<b>Name</b>	<b>Description</b>
 <a href="#">individualRankings</a>		

## Remarks

[http://www.pohlheim.com/Papers/mpga\\_gal95/gal2\\_3.html](http://www.pohlheim.com/Papers/mpga_gal95/gal2_3.html)

## See Also

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram) Constructor

Creates a new [RankFitnessFunction\(TProgram\)](#) with the given parameters.

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected RankFitnessFunction(  
    IComparer<TProgram> programComparer,  
    IPopulation<TProgram> population  
)
```

[View Source](#)

### Parameters

*programComparer*

Type: [System.Collections.Generic.IComparer\(TProgram\)](#)

The object used to compare programs and determine their ranking.

*population*

Type: [Genetica.IPopulation\(TProgram\)](#)

The list of programs whose fitness can be computed by this function.

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram).RankFitnessFunction(TProgram) Properties

The [RankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">SelectivePressure</a>	Gets or sets the "probability of the best individual being selected compared to the average probability of selection of all individuals". The value should be in [1.0, 2.0].

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram).SelectivePressure Property

Gets or sets the "probability of the best individual being selected compared to the average probability of selection of all individuals". The value should be in [1.0, 2.0].

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double SelectivePressure { get; set; }
```

[View Source](#)

### Property Value

Type: [Double](#)

### Remarks

[http://www.pohlheim.com/Papers/mpga\\_gal95/gal2\\_3.html](http://www.pohlheim.com/Papers/mpga_gal95/gal2_3.html)

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram).RankFitnessFunction(TProgram) Methods

The [RankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Compare</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Evaluate</a>	Evaluates the fitness of the given program.
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram).Compare Method

[Missing <summary> documentation for  
"M:Genetica.Evaluation.RankFitnessFunction`1.Compare(`o,`o)"]

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public int Compare(  
    TProgram x,  
    TProgram y  
)
```

[View Source](#)

### Parameters

x

Type: *TProgram*

[Missing <param name="x"/> documentation for  
"M:Genetica.Evaluation.RankFitnessFunction`1.Compare(`o,`o)"]

y

Type: *TProgram*

[Missing <param name="y"/> documentation for  
"M:Genetica.Evaluation.RankFitnessFunction`1.Compare(`o,`o)"]

### Return Value

Type: [Int32](#)

[Missing <returns> documentation for  
"M:Genetica.Evaluation.RankFitnessFunction`1.Compare(`o,`o)"]

### Implements

[IComparer\(T\).Compare\(T, T\)](#)

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram).Evaluate Method

Evaluates the fitness of the given program.

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public abstract double Evaluate(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be evaluated.

#### Return Value

Type: [Double](#)

A score value denoting the fitness of the given program.

#### Implements

[IFitnessFunction\(TProgram\).Evaluate\(TProgram\)](#)

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram).RankFitnessFunction(TProgram) Fields

The [RankFitnessFunction\(TProgram\)](#) generic type exposes the following members.

### Fields

	Name	Description
	<a href="#">individualRankings</a>	

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## RankFitnessFunction(TProgram).individualRankings Field

[Missing <summary> documentation for  
"F:Genetica.Evaluation.RankFitnessFunction`1.individualRankings"]

**Namespace:** [Genetica.Evaluation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected readonly Dictionary<TProgram, int> individualRankings
```

[View Source](#)

### Field Value

Type: [Dictionary\(TProgram, Int32\)](#)

### See Also

[RankFitnessFunction\(TProgram\)Class](#)

[Genetica.Evaluation Namespace](#)

## Genetica.Graphviz Namespace

### Classes

Class	Description
 <a href="#">Edge</a>	Represents a vertex between two <a href="#">Vertex</a> to be used in GraphViz.
 <a href="#">Extensions</a>	Provides extension methods to print <a href="#">ITreeNode</a> elements such as <a href="#">ITreeProgram</a> to image files using Graphviz.
 <a href="#">Vertex</a>	Represents a vertex for <a href="#">ITreeNode</a> types to be used in GraphViz.

## Edge Class

Represents a vertex between two [Vertex](#) to be used in GraphViz.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Graphviz.Edge

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public class Edge : IEdge<Vertex>
```

[View Source](#)

The **Edge** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">Edge</a>	Creates a new <a href="#">Vertex</a> between the two given vertexes.

### Properties

	Name	Description
	<a href="#">Source</a>	Gets the source vertex
	<a href="#">Target</a>	Gets the target vertex

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[Genetica.Graphviz Namespace](#)

## Edge Constructor

Creates a new [Vertex](#) between the two given vertexes.

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public Edge(  
    Vertex source,  
    Vertex target  
)
```

[View Source](#)

### Parameters

*source*

Type: [Genetica.Graphviz.Vertex](#)

The source vertex.

*target*

Type: [Genetica.Graphviz.Vertex](#)

The destination vertex.

### See Also

[Edge Class](#)

[Genetica.Graphviz Namespace](#)

## Edge.Edge Properties

The [Edge](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Source</a>	Gets the source vertex
	<a href="#">Target</a>	Gets the target vertex

### See Also

[Edge Class](#)

[Genetica.Graphviz Namespace](#)

## Edge.Source Property

Gets the source vertex

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public Vertex Source { get; }
```

[View Source](#)

*Property Value*

Type: [Vertex](#)

*Implements*

**IEdge.Source**

### See Also

[Edge Class](#)

[Genetica.Graphviz Namespace](#)

## Edge.Target Property

Gets the target vertex

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public Vertex Target { get; }
```

[View Source](#)

*Property Value*

Type: [Vertex](#)

*Implements*

[IEdge.Target](#)

### See Also

[Edge Class](#)

[Genetica.Graphviz Namespace](#)

## Edge.Edge Methods

The [Edge](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[Edge Class](#)

[Genetica.Graphviz Namespace](#)

## Extensions Class

Provides extension methods to print [ITreeNode](#) elements such as [ITreeProgram](#) to image files using Graphviz.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Graphviz.Extensions

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public static class Extensions
```

[View Source](#)

### Methods

	Name	Description
 	<a href="#">ToGraphvizFile(ITreeNode, String, String, GraphvizImageType, Int32)</a>	Saves the given <a href="#">ITreeNode</a> to an image file.
 	<a href="#">ToGraphvizFile(ITreeNode, String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Saves the given <a href="#">ITreeNode</a> to an image file.
 	<a href="#">ToGraphvizFile(TProgram)(IInformationTree(TProgram), String, String, GraphvizImageType, Int32)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file.

### See Also

[Genetica.Graphviz Namespace](#)

## Extensions.Extensions Methods

### Methods

Name	Description
  <a href="#">ToGraphvizFile(ITreeNode, String, String, GraphvizImageType, Int32)</a>	Saves the given <a href="#">ITreeNode</a> to an image file.
  <a href="#">ToGraphvizFile(ITreeNode, String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Saves the given <a href="#">ITreeNode</a> to an image file.
  <a href="#">ToGraphvizFile(TProgram)(IInformationTree(TProgram), String, String, GraphvizImageType, Int32)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file.

### See Also

[Extensions Class](#)

[Genetica.Graphviz Namespace](#)

## Extensions.ToGraphvizFile Method

### Overload List

Name	Description
  <a href="#">ToGraphvizFile(TProgram)(IInformationTree(TProgram), String, String, GraphvizImageType, Int32)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file.
  <a href="#">ToGraphvizFile(ITreeNode, String, String, GraphvizImageType, Int32)</a>	Saves the given <a href="#">ITreeNode</a> to an image file.
  <a href="#">ToGraphvizFile(ITreeNode, String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Saves the given <a href="#">ITreeNode</a> to an image file.

### See Also

[Extensions Class](#)

[Genetica.Graphviz Namespace](#)

## Extensions.ToGraphvizFile(TProgram) Method (IInformationTree(TProgram), String, String, GraphvizImageType, Int32)

Saves the given [IInformationTree\(TProgram\)](#) to an image file.

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

#### C#

```
public static string ToGraphvizFile<TProgram>(  
    this IInformationTree<TProgram> tree,  
    string basePath,  
    string fileName,  
    GraphvizImageType imageType = GraphvizImageType.Pdf,  
    int timeout = 2000  
)  
where TProgram : ITreeProgram
```

[View Source](#)

#### Parameters

*tree*

Type: [Genetica.Trees.IInformationTree\(TProgram\)](#)

The information tree to be saved.

*basePath*

Type: [System.String](#)

The path in which to save the given information tree

*fileName*

Type: [System.String](#)

The name of the image file to be saved (without extension)

*imageType* (Optional)

Type: [GraphvizImageType](#)

The type of image file in which to save the tree.

*timeout* (Optional)

Type: [System.Int32](#)

The maximum time to wait for Graphviz to create the image file.

#### Type Parameters

*TProgram*

The type of program.

[Return Value](#)

Type: [String](#)

The path to file where the tree image file was saved.

[Usage Note](#)

In Visual Basic and C#, you can call this method as an instance method on any object of type [IInformationTree\(TProgram\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

[See Also](#)

[Extensions Class](#)

[ToGraphvizFile Overload](#)

[Genetica.Graphviz Namespace](#)

## Extensions.ToGraphvizFile Method (ITreeNode, String, String, GraphvizImageType, Int32)

Saves the given [ITreeNode](#) to an image file.

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public static string ToGraphvizFile(  
    this ITreeNode rootNode,  
    string basePath,  
    string fileName,  
    GraphvizImageType imageType = GraphvizImageType.Pdf,  
    int timeout = 2000  
)
```

[View Source](#)

### Parameters

*rootNode*

Type: [Genetica.ITreeNode](#)

The root node of the tree to be saved.

*basePath*

Type: [System.String](#)

The path in which to save the given tree

*fileName*

Type: [System.String](#)

The name of the image file to be saved (without extension)

*imageType* (Optional)

Type: [GraphvizImageType](#)

The type of image file in which to save the tree.

*timeout* (Optional)

Type: [System.Int32](#)

The maximum time to wait for Graphviz to create the image file.

### Return Value

Type: [String](#)

The path to file where the tree image file was saved.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeNode](#). When you use instance method syntax to call this method, omit the first parameter. For

more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

## See Also

[Extensions Class](#)

[ToGraphvizFile Overload](#)

[Genetica.Graphviz Namespace](#)

## Extensions.ToGraphvizFile Method (ITreeNode, String, String, FormatVertexEventHandler`1(Vertex), FormatEdgeAction`2(Vertex, Edge), GraphvizImageType, Int32)

Saves the given [ITreeNode](#) to an image file.

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

#### C#

```
public static string ToGraphvizFile(  
    this ITreeNode rootNode,  
    string basePath,  
    string fileName,  
    FormatVertexEventHandler<Vertex> formatVertex,  
    FormatEdgeAction<Vertex, Edge> formatEdge,  
    GraphvizImageType imageType = GraphvizImageType.Pdf,  
    int timeout = 2000  
)
```

[View Source](#)

#### Parameters

*rootNode*

Type: [Genetica.ITreeNode](#)

The root node of the tree to be saved.

*basePath*

Type: [System.String](#)

The path in which to save the given tree

*fileName*

Type: [System.String](#)

The name of the image file to be saved (without extension)

*formatVertex*

Type: [FormatVertexEventHandler](#)([Vertex](#))

The delegate used to format the vertexes.

*formatEdge*

Type: [FormatEdgeAction](#)([Vertex](#), [Edge](#))

The delegate used to format the edges.

*imageType* (Optional)

Type: [GraphvizImageType](#)

The type of image file in which to save the tree.

`timeout` (Optional)

Type: [System.Int32](#)

The maximum time to wait for Graphviz to create the image file.

*Return Value*

Type: [String](#)

The path to file where the tree image file was saved.

*Usage Note*

In Visual Basic and C#, you can call this method as an instance method on any object of type [ITreeNode](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

*See Also*

[Extensions Class](#)

[ToGraphvizFile Overload](#)

[Genetica.Graphviz Namespace](#)

## Vertex Class

Represents a vertex for [ITreeNode](#) types to be used in GraphViz.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Graphviz.Vertex

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public class Vertex
```

[View Source](#)

The **Vertex** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">Vertex</a>	Creates a new <b>Vertex</b> with the given node.

### Properties

	Name	Description
	<a href="#">Node</a>	The <a href="#">ITreeNode</a> associated with this vertex.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Overrides <a href="#">Object.GetHashCode()</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

### See Also

[Genetica.Graphviz Namespace](#)

## Vertex Constructor

Creates a new [Vertex](#) with the given node.

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public Vertex(  
    ITreeNode node  
)
```

[View Source](#)

#### Parameters

*node*

Type: [Genetica.ITreeNode](#)

The node associated with the vertex.

#### See Also

[Vertex Class](#)

[Genetica.Graphviz Namespace](#)

## Vertex.Vertex Properties

The [Vertex](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Node</a>	The <a href="#">ITreeNode</a> associated with this vertex.

### See Also

[Vertex Class](#)

[Genetica.Graphviz Namespace](#)

## Vertex.Node Property

The [ITreeNode](#) associated with this vertex.

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public ITreeNode Node { get; }
```

[View Source](#)

*Property Value*

Type: [ITreeNode](#)

### See Also

[Vertex Class](#)

[Genetica.Graphviz Namespace](#)

## Vertex.Vertex Methods

The [Vertex](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Overrides <a href="#">Object.GetHashCode()</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

### See Also

[Vertex Class](#)

[Genetica.Graphviz Namespace](#)

## Vertex.GetHashCode Method

[Missing <summary> documentation for "M:Genetica.Graphviz.Vertex.GetHashCode"]

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public override int GetHashCode()
```

[View Source](#)

### Return Value

Type: [Int32](#)

[Missing <returns> documentation for "M:Genetica.Graphviz.Vertex.GetHashCode"]

### See Also

[Vertex Class](#)

[Genetica.Graphviz Namespace](#)

## Vertex.ToString Method

[Missing <summary> documentation for "M:Genetica.Graphviz.Vertex.ToString"]

**Namespace:** [Genetica.Graphviz](#)

**Assembly:** Genetica.Graphviz (in Genetica.Graphviz.dll) Version: 1.0.6669.36763

### Syntax

C#

```
public override string ToString()
```

[View Source](#)

**Return Value**

Type: [String](#)

[Missing <returns> documentation for "M:Genetica.Graphviz.Vertex.ToString"]

### See Also

[Vertex Class](#)

[Genetica.Graphviz Namespace](#)

## Genetica.Operators.Crossover Namespace

### Classes

Class	Description
 <a href="#">ContextPreservingCrossover(TProgram, TOutput)</a>	Represents a <a href="#">ICrossoverOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . Creates offspring by choosing a random sub-program of the first parent and replacing with a sub-program of the second parent that has the same index.
 <a href="#">OnePointCrossover(TProgram, TOutput)</a>	Represents a <a href="#">ICrossoverOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . Creates offspring by selecting a random crossover point in the common region between the two <a href="#">ITreeProgram(TOutput)</a> parents and then replacing a subtree of the first parent by the corresponding subtree of the second parent.
 <a href="#">StochasticCrossover(TProgram)</a>	Represents a generic <a href="#">ICrossoverOperator(TProgram)</a> that selects from a list of crossover operators at random.
 <a href="#">SubtreeCrossover(TProgram, TOutput)</a>	Represents a <a href="#">ICrossoverOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . This crossover operator replaces a random function sub-program of the first parent by a random program of the second parent. If the first parent is a leaf, then the second parent is returned.
 <a href="#">UniformCrossover(TProgram, TOutput)</a>	Represents a <a href="#">ICrossoverOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . Creates offspring by visiting the points in the common region between the parents and flipping a coin at each point to decide whether the corresponding offspring sub-program should be picked from the first or the second parent.

### Interfaces

Interface	Description
 <a href="#">ICrossoverOperator(TProgram)</a>	An interface for crossover operators for <a href="#">IProgram(TInput, TOutput)</a> , i.e., operators that take two parent programs and produce a new program that results in some combination of the parent's sub-programs.

## ContextPreservingCrossover(TProgram, TOutput) Class

Represents a [ICrossoverOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). Creates offspring by choosing a random sub-program of the first parent and replacing with a sub-program of the second parent that has the same index.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Crossover.ContextPreservingCrossover(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class ContextPreservingCrossover<TProgram, TOutput> : ICrossoverOperator<TProgram>,  
    IDisposable  
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The ContextPreservingCrossover(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">ContextPreservingCrossover(TProgram, TOutput)</a>	Initializes a new instance of the ContextPreservingCrossover(TProgram, TOutput) class

### Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program by choosing a random sub-program of the first parent and replacing with a sub-program of the second parent that has the same index.
	<a href="#">Dispose</a>	Releases all resources used by the ContextPreservingCrossover(TProgram, TOutput)

 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Crossover Namespace](#)

## ContextPreservingCrossover(TProgram, TOutput) Constructor

Initializes a new instance of the [ContextPreservingCrossover\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ContextPreservingCrossover()
```

[View Source](#)

### See Also

[ContextPreservingCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## ContextPreservingCrossover(TProgram, TOutput).ContextPreservingCrossover(TProgram, TOutput) Methods

The [ContextPreservingCrossover\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

Name	Description
<a href="#">Crossover</a>	Creates a new program by choosing a random sub-program of the first parent and replacing with a sub-program of the second parent that has the same index.
<a href="#">Dispose</a>	
<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
<a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[ContextPreservingCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## ContextPreservingCrossover(TProgram, TOutput).Crossover Method

Creates a new program by choosing a random sub-program of the first parent and replacing with a sub-program of the second parent that has the same index.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Crossover(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: *TProgram*

The first parent program.

*parent2*

Type: *TProgram*

The second parent program.

### Return Value

Type: *TProgram*

A new program resulting from the crossover between the given parent programs.

### Implements

[ICrossoverOperator\(TProgram\).Crossover\(TProgram, TProgram\)](#)

### See Also

[ContextPreservingCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## ContextPreservingCrossover(TProgram, TOutput).Dispose Method

Releases all resources used by the [ContextPreservingCrossover\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[ContextPreservingCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## ContextPreservingCrossover(*TProgram*, *TOutput*).GetAllOffspring Method

Gets a list containing all possible offspring programs resulting from applying this crossover operator.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> GetAllOffspring(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: *TProgram*

The first parent program.

*parent2*

Type: *TProgram*

The second parent program.

### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible offspring programs resulting from applying this crossover operator.

### Implements

[ICrossoverOperator\(TProgram\).GetAllOffspring\(TProgram, TProgram\)](#)

### See Also

[ContextPreservingCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## ICrossoverOperator(TProgram) Interface

An interface for crossover operators for [IProgram\(TInput, TOutput\)](#), i.e., operators that take two parent programs and produce a new program that results in some combination of the parent's sub-programs.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface ICrossoverOperator<TProgram> : IDisposable  
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

The ICrossoverOperator(TProgram) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program resulting from the crossover between the given parent programs.
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.

### See Also

[Genetica.Operators.Crossover Namespace](#)

## [ICrossoverOperator\(TProgram\).ICrossoverOperator\(TProgram\)](#)

### Methods

The [ICrossoverOperator\(TProgram\)](#) generic type exposes the following members.

#### Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program resulting from the crossover between the given parent programs.
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.

#### See Also

[ICrossoverOperator\(TProgram\)Interface](#)

[Genetica.Operators.Crossover Namespace](#)

## [ICrossoverOperator\(TProgram\).Crossover Method](#)

Creates a new program resulting from the crossover between the given parent programs.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
TProgram Crossover(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

#### Parameters

*parent1*

Type: [TProgram](#)

The first parent program.

*parent2*

Type: [TProgram](#)

The second parent program.

#### Return Value

Type: [TProgram](#)

A new program resulting from the crossover between the given parent programs.

#### See Also

[ICrossoverOperator\(TProgram\)Interface](#)

[Genetica.Operators.Crossover Namespace](#)

## **ICrossoverOperator(TProgram).GetAllOffspring Method**

Gets a list containing all possible offspring programs resulting from applying this crossover operator.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
IEnumerable<TProgram> GetAllOffspring(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: [TProgram](#)

The first parent program.

*parent2*

Type: [TProgram](#)

The second parent program.

### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible offspring programs resulting from applying this crossover operator.

### See Also

[ICrossoverOperator\(TProgram\)Interface](#)

[Genetica.Operators.Crossover Namespace](#)

## OnePointCrossover(TProgram, TOutput) Class

Represents a [ICrossoverOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). Creates offspring by selecting a random crossover point in the common region between the two [ITreeProgram\(TOutput\)](#) parents and then replacing a subtree of the first parent by the corresponding subtree of the second parent.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Crossover.OnePointCrossover(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class OnePointCrossover<TProgram, TOutput> : ICrossoverOperator<TProgram>,  
    IDisposable  
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The OnePointCrossover(TProgram, TOutput) type exposes the following members.

#### Constructors

	Name	Description
	<a href="#">OnePointCrossover(TProgram, TOutput)</a>	Initializes a new instance of the OnePointCrossover(TProgram, TOutput) class

#### Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program resulting from the crossover between the given parent programs.
	<a href="#">Dispose</a>	Releases all resources used by the OnePointCrossover(TProgram, TOutput)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Remarks

The common region between the parent programs corresponds to the subtrees where the parents have the same shape.

## See Also

[Genetica.Operators.Crossover Namespace](#)

## OnePointCrossover(TProgram, TOutput) Constructor

Initializes a new instance of the [OnePointCrossover\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public OnePointCrossover()
```

[View Source](#)

### See Also

[OnePointCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## OnePointCrossover(TProgram, TOutput).OnePointCrossover(TProgram, TOutput) Methods

The [OnePointCrossover\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program resulting from the crossover between the given parent programs.
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[OnePointCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## OnePointCrossover(TProgram, TOutput).Crossover Method

Creates a new program resulting from the crossover between the given parent programs.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public TProgram Crossover(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

#### Parameters

*parent1*

Type: [TProgram](#)

The first parent program.

*parent2*

Type: [TProgram](#)

The second parent program.

#### Return Value

Type: [TProgram](#)

A new program resulting from the crossover between the given parent programs.

#### Implements

[ICrossoverOperator\(TProgram\).Crossover\(TProgram, TProgram\)](#)

### See Also

[OnePointCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## OnePointCrossover(TProgram, TOutput).Dispose Method

Releases all resources used by the [OnePointCrossover\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[OnePointCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## OnePointCrossover(TProgram, TOutput).GetAllOffspring Method

Gets a list containing all possible offspring programs resulting from applying this crossover operator.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public IEnumerable<TProgram> GetAllOffspring(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: [TProgram](#)

The first parent program.

*parent2*

Type: [TProgram](#)

The second parent program.

### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible offspring programs resulting from applying this crossover operator.

### Implements

[ICrossoverOperator\(TProgram\).GetAllOffspring\(TProgram, TProgram\)](#)

### See Also

[OnePointCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## StochasticCrossover(TProgram) Class

Represents a generic [ICrossoverOperator\(TProgram\)](#) that selects from a list of crossover operators at random.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Crossover.StochasticCrossover(TProgram)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class StochasticCrossover<TProgram> : ICrossoverOperator<TProgram>,
    IDisposable
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

The StochasticCrossover(TProgram) type exposes the following members.

### Constructors

Name	Description
<a href="#">StochasticCrossover(TProgram)(IDictionary(ICrossoverOperator(TProgram), Double))</a>	Creates a new StochasticCrossover(TProgram) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.
<a href="#">StochasticCrossover(TProgram)(IList(ICrossoverOperator(TProgram)))</a>	Creates a new StochasticCrossover(TProgram) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

## Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program resulting from the crossover between the given parent programs.
	<a href="#">Dispose</a>	Releases all resources used by the StochasticCrossover(TProgram)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Crossover Namespace](#)

## StochasticCrossover(TProgram) Constructor

### Overload List

Name	Description
<a href="#">StochasticCrossover(TProgram)(IDictionary(ICrossoverOperator(TProgram), Double))</a>	Creates a new <a href="#">StochasticCrossover(TProgram)</a> with the given list of operators to choose from. Operators will be selected according to the corresponding probability.
<a href="#">StochasticCrossover(TProgram)(IList(ICrossoverOperator(TProgram)))</a>	Creates a new <a href="#">StochasticCrossover(TProgram)</a> with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

### See Also

[StochasticCrossover\(TProgram\)Class](#)  
[Genetica.Operators.Crossover Namespace](#)

## StochasticCrossover(TProgram) Constructor (IDictionary(ICrossoverOperator(TProgram), Double))

Creates a new [StochasticCrossover\(TProgram\)](#) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public StochasticCrossover(  
    IDictionary<ICrossoverOperator<TProgram>, double> possibleCrossovers  
)
```

[View Source](#)

### Parameters

possibleCrossovers

Type: [System.Collections.Generic.IDictionary\(ICrossoverOperator\(TProgram\), Double\)](#)

A list of crossover operators to be used by this operator and the corresponding selection probabilities.

### See Also

[StochasticCrossover\(TProgram\)Class](#)  
[StochasticCrossover\(TProgram\)Overload](#)  
[Genetica.Operators.Crossover Namespace](#)

## StochasticCrossover(TProgram) Constructor (IList(ICrossoverOperator(TProgram)))

Creates a new [StochasticCrossover\(TProgram\)](#) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public StochasticCrossover(  
    IList<ICrossoverOperator<TProgram>> possibleCrossovers  
)
```

[View Source](#)

### Parameters

possibleCrossovers

Type: [System.Collections.Generic.IList\(ICrossoverOperator\(TProgram\)\)](#)

The list of crossover operators to be used by this operator.

### See Also

[StochasticCrossover\(TProgram\)Class](#)  
[StochasticCrossover\(TProgram\)Overload](#)  
[Genetica.Operators.Crossover Namespace](#)

## StochasticCrossover(TProgram).StochasticCrossover(TProgram) Methods

The [StochasticCrossover\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program resulting from the crossover between the given parent programs.
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[StochasticCrossover\(TProgram\)Class](#)  
[Genetica.Operators.Crossover Namespace](#)

## StochasticCrossover(TProgram).Crossover Method

Creates a new program resulting from the crossover between the given parent programs.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public TProgram Crossover(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

#### Parameters

*parent1*

Type: [TProgram](#)

The first parent program.

*parent2*

Type: [TProgram](#)

The second parent program.

#### Return Value

Type: [TProgram](#)

A new program resulting from the crossover between the given parent programs.

#### Implements

[ICrossoverOperator\(TProgram\).Crossover\(TProgram, TProgram\)](#)

### See Also

[StochasticCrossover\(TProgram\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## [StochasticCrossover\(TProgram\).Dispose Method](#)

Releases all resources used by the [StochasticCrossover\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

Implements

[IDisposable.Dispose\(\)](#)

### See Also

[StochasticCrossover\(TProgram\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## StochasticCrossover(TProgram).GetAllOffspring Method

Gets a list containing all possible offspring programs resulting from applying this crossover operator.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> GetAllOffspring(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: [TProgram](#)

The first parent program.

*parent2*

Type: [TProgram](#)

The second parent program.

### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible offspring programs resulting from applying this crossover operator.

### Implements

[ICrossoverOperator\(TProgram\).GetAllOffspring\(TProgram, TProgram\)](#)

### See Also

[StochasticCrossover\(TProgram\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## SubtreeCrossover(TProgram, TOutput) Class

Represents a [ICrossoverOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). This crossover operator replaces a random function sub-program of the first parent by a random program of the second parent. If the first parent is a leaf, then the second parent is returned.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Crossover.SubtreeCrossover(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SubtreeCrossover<TProgram, TOutput> : ICrossoverOperator<TProgram>,
    IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The SubtreeCrossover(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SubtreeCrossover(TProgram, TOutput)</a>	Initializes a new instance of the SubtreeCrossover(TProgram, TOutput) class

### Methods

	Name	Description
	<a href="#">Crossover</a>	Creates a new program by replacing a random function sub-program of the first parent by a random program of the second parent. If the first parent is a leaf, then the second parent is returned.
	<a href="#">Dispose</a>	Releases all resources used by the SubtreeCrossover(TProgram, TOutput)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllOffspring</a>	
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Crossover Namespace](#)

## [SubtreeCrossover\(TProgram, TOutput\)](#) Constructor

Initializes a new instance of the [SubtreeCrossover\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SubtreeCrossover()
```

[View Source](#)

### See Also

[SubtreeCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## SubtreeCrossover(TProgram, TOutput).SubtreeCrossover(TProgram, TOutput) Methods

The [SubtreeCrossover\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
 <a href="#">Crossover</a>		Creates a new program by replacing a random function sub-program of the first parent by a random program of the second parent. If the first parent is a leaf, then the second parent is returned.
 <a href="#">Dispose</a>		
 <a href="#">Equals</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllOffspring</a>		
 <a href="#">GetHashCode</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>		(Inherited from <a href="#">Object</a> .)

### See Also

[SubtreeCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## SubtreeCrossover(TProgram, TOutput).Crossover Method

Creates a new program by replacing a random function sub-program of the first parent by a random program of the second parent. If the first parent is a leaf, then the second parent is returned.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public TProgram Crossover(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: *TProgram*

The first parent program.

*parent2*

Type: *TProgram*

The second parent program.

### Return Value

Type: *TProgram*

A new program resulting from the crossover between the given parent programs.

### Implements

[ICrossoverOperator\(TProgram\).Crossover\(TProgram, TProgram\)](#)

### See Also

[SubtreeCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## **SubtreeCrossover(TProgram, TOutput).Dispose Method**

Releases all resources used by the [SubtreeCrossover\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[SubtreeCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## SubtreeCrossover(TProgram, TOutput).GetAllOffspring Method

[Missing <summary> documentation for  
"M:Genetica.Operators.Crossover.SubtreeCrossover`2.GetAllOffspring(`o,`o)"]

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public IEnumerable<TProgram> GetAllOffspring(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: *TProgram*

[Missing <param name="parent1"/> documentation for

"M:Genetica.Operators.Crossover.SubtreeCrossover`2.GetAllOffspring(`o,`o)"]

*parent2*

Type: *TProgram*

[Missing <param name="parent2"/> documentation for

"M:Genetica.Operators.Crossover.SubtreeCrossover`2.GetAllOffspring(`o,`o)"]

### Return Value

Type: [IEnumerable](#)(*TProgram*)

[Missing <returns> documentation for

"M:Genetica.Operators.Crossover.SubtreeCrossover`2.GetAllOffspring(`o,`o)"]

### Implements

[ICrossoverOperator\(TProgram\).GetAllOffspring\(TProgram, TProgram\)](#)

### See Also

[SubtreeCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## UniformCrossover(TProgram, TOutput) Class

Represents a [ICrossoverOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). Creates offspring by visiting the points in the common region between the parents and flipping a coin at each point to decide whether the corresponding offspring sub-program should be picked from the first or the second parent.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Crossover.UniformCrossover(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class UniformCrossover<TProgram, TOutput> : ICrossoverOperator<TProgram>,
    IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The UniformCrossover(TProgram, TOutput) type exposes the following members.

#### Constructors

	Name	Description
	<a href="#">UniformCrossover(TProgram, TOutput)</a>	Initializes a new instance of the UniformCrossover(TProgram, TOutput) class

#### Methods

	Name	Description
	<a href="#">Crossover</a>	Iterates the common (structural) region between the two parents, picking a sub-program from one of the parents at random.
	<a href="#">Dispose</a>	Releases all resources used by the UniformCrossover(TProgram, TOutput)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Remarks

The common region between the parent programs corresponds to the subtrees where the parents have the same shape.

## See Also

[Genetica.Operators.Crossover Namespace](#)

## UniformCrossover(TProgram, TOutput) Constructor

Initializes a new instance of the [UniformCrossover\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public UniformCrossover()
```

[View Source](#)

### See Also

[UniformCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## UniformCrossover(TProgram, TOutput).UniformCrossover(TProgram, TOutput) Methods

The [UniformCrossover\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Crossover</a>	Iterates the common (structural) region between the two parents, picking a sub-program from one of the parents at random.
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllOffspring</a>	Gets a list containing all possible offspring programs resulting from applying this crossover operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[UniformCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## UniformCrossover(TProgram, TOutput).Crossover Method

Iterates the common (structural) region between the two parents, picking a sub-program from one of the parents at random.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public TProgram Crossover(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

#### Parameters

*parent1*

Type: *TProgram*

The first parent program.

*parent2*

Type: *TProgram*

The second parent program.

#### Return Value

Type: *TProgram*

A new program resulting from the uniform crossover between the given parent programs.

#### Implements

[ICrossoverOperator\(TProgram\).Crossover\(TProgram, TProgram\)](#)

### See Also

[UniformCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## **UniformCrossover(TProgram, TOutput).Dispose Method**

Releases all resources used by the [UniformCrossover\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[UniformCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## UniformCrossover(TProgram, TOutput).GetAllOffspring Method

Gets a list containing all possible offspring programs resulting from applying this crossover operator.

**Namespace:** [Genetica.Operators.Crossover](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public IEnumerable<TProgram> GetAllOffspring(  
    TProgram parent1,  
    TProgram parent2  
)
```

[View Source](#)

### Parameters

*parent1*

Type: [TProgram](#)

The first parent program.

*parent2*

Type: [TProgram](#)

The second parent program.

### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible offspring programs resulting from applying this crossover operator.

### Implements

[ICrossoverOperator\(TProgram\).GetAllOffspring\(TProgram, TProgram\)](#)

### See Also

[UniformCrossover\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Crossover Namespace](#)

## Genetica.Operators.Generation Namespace

### Classes

Class	Description
 <a href="#">FullProgramGenerator(TProgram, TOutput)</a>	Represents a <a href="#">IProgramGenerator(TProgram, TOutput)</a> that generates programs of a given depth.
 <a href="#">GrowProgramGenerator(TProgram, TOutput)</a>	Represents a <a href="#">IProgramGenerator(TProgram, TOutput)</a> that generates programs with some maximum length.
 <a href="#">StochasticProgramGenerator(TProgram, TOutput)</a>	Represents a generic <a href="#">IProgramGenerator(TProgram, TOutput)</a> that selects from a list of generator operators at random.

### Interfaces

Interface	Description
 <a href="#">IProgramGenerator(TProgram, TOutput)</a>	An interface for generation operators for <a href="#">ITreeProgram(TOutput)</a> , i.e., operators that generate new programs by randomly combining elements from a given <a href="#">PrimitiveSet(TProgram)</a> .

## FullProgramGenerator(TProgram, TOutput) Class

Represents a [IProgramGenerator\(TProgram, TOutput\)](#) that generates programs of a given depth.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Generation.FullProgramGenerator(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class FullProgramGenerator<TProgram, TOutput> : IProgramGenerator<TProgram, TOutput>,  
    IDisposable  
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

**TProgram**

The type of program.

**TOutput**

The type of program output.

The FullProgramGenerator(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
≡	<a href="#">FullProgramGenerator(TProgram, TOutput)</a>	Initializes a new instance of the FullProgramGenerator(TProgram, TOutput) class

### Methods

	Name	Description
≡	<a href="#">Dispose</a>	Releases all resources used by the FullProgramGenerator(TProgram, TOutput)
≡	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Generation Namespace](#)

## FullProgramGenerator(TProgram, TOutput) Constructor

Initializes a new instance of the [FullProgramGenerator\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public FullProgramGenerator()
```

[View Source](#)

### See Also

[FullProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## FullProgramGenerator(TProgram, TOutput).FullProgramGenerator(TProgram, TOutput) Methods

The [FullProgramGenerator\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[FullProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## FullProgramGenerator(TProgram, TOutput).Dispose Method

Releases all resources used by the [FullProgramGenerator\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[FullProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## FullProgramGenerator(TProgram, TOutput).Generate Method

Generates a new program having some given maximum depth by randomly combining elements from the given [PrimitiveSet\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Generate(  
    PrimitiveSet<TProgram> primitives,  
    uint maxDepth  
)
```

[View Source](#)

#### Parameters

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The set from which to select primitives.

*maxDepth*

Type: [System.UInt32](#)

The maximum depth of the program generated.

#### Return Value

Type: *TProgram*

The new generated program.

#### Implements

[IProgramGenerator\(TProgram, TOutput\).Generate\(PrimitiveSet\(TProgram\), UInt32\)](#)

### See Also

[FullProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## GrowProgramGenerator(TProgram, TOutput) Class

Represents a [IProgramGenerator\(TProgram, TOutput\)](#) that generates programs with some maximum length.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Generation.GrowProgramGenerator(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class GrowProgramGenerator<TProgram, TOutput> : IProgramGenerator<TProgram, TOutput>,  
    IDisposable  
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The GrowProgramGenerator(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
≡	<a href="#">GrowProgramGenerator(TProgram, TOutput)</a>	Initializes a new instance of the GrowProgramGenerator(TProgram, TOutput) class

### Methods

	Name	Description
≡	<a href="#">Dispose</a>	Releases all resources used by the GrowProgramGenerator(TProgram, TOutput)
≡	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
💡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>

 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Generation Namespace](#)

## **GrowProgramGenerator(TProgram, TOutput) Constructor**

Initializes a new instance of the [GrowProgramGenerator\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public GrowProgramGenerator()
```

[View Source](#)

### See Also

[GrowProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## **GrowProgramGenerator(TProgram, TOutput).GrowProgramGenerator(TProgram, TOutput) Methods**

The [GrowProgramGenerator\(TProgram, TOutput\)](#) generic type exposes the following members.

### **Methods**

	<b>Name</b>	<b>Description</b>
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### **See Also**

[GrowProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## [GrowProgramGenerator\(TProgram, TOutput\).Dispose Method](#)

Releases all resources used by the [GrowProgramGenerator\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

Implements

[IDisposable.Dispose\(\)](#)

### See Also

[GrowProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## [GrowProgramGenerator\(TProgram, TOutput\).Generate Method](#)

Generates a new program having some given maximum depth by randomly combining elements from the given [PrimitiveSet\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Generate(  
    PrimitiveSet<TProgram> primitives,  
    uint maxDepth  
)
```

[View Source](#)

#### Parameters

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The set from which to select primitives.

*maxDepth*

Type: [System.UInt32](#)

The maximum depth of the program generated.

#### Return Value

Type: *TProgram*

The new generated program.

#### Implements

[IProgramGenerator\(TProgram, TOutput\).Generate\(PrimitiveSet\(TProgram\), UInt32\)](#)

### See Also

[GrowProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## IProgramGenerator(TProgram, TOutput) Interface

An interface for generation operators for [ITreeProgram\(TOutput\)](#), i.e., operators that generate new programs by randomly combining elements from a given [PrimitiveSet\(TProgram\)](#).

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IProgramGenerator<TProgram, TOutput> : IDisposable  
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

**TProgram**

The type of program.

**TOutput**

The type of program output.

The IProgramGenerator(TProgram, TOutput) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>

### See Also

[Genetica.Operators.Generation Namespace](#)

## IProgramGenerator(TProgram, TOutput).IProgramGenerator(TProgram, TOutput) Methods

The [IProgramGenerator\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>

### See Also

[IProgramGenerator\(TProgram, TOutput\)Interface](#)

[Genetica.Operators.Generation Namespace](#)

## IProgramGenerator(TProgram, TOutput).Generate Method

Generates a new program having some given maximum depth by randomly combining elements from the given [PrimitiveSet\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
TProgram Generate(  
    PrimitiveSet<TProgram> primitives,  
    uint maxDepth  
)
```

[View Source](#)

#### Parameters

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The set from which to select primitives.

*maxDepth*

Type: [System.UInt32](#)

The maximum depth of the program generated.

#### Return Value

Type: *TProgram*

The new generated program.

### See Also

[IProgramGenerator\(TProgram, TOutput\)Interface](#)

[Genetica.Operators.Generation Namespace](#)

## StochasticProgramGenerator(TProgram, TOutput) Class

Represents a generic [IProgramGenerator\(TProgram, TOutput\)](#) that selects from a list of generator operators at random.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Generation.StochasticProgramGenerator(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public sealed class StochasticProgramGenerator<TProgram, TOutput> : IProgramGenerator<TProgram, TOutput>,  
    IDisposable  
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The StochasticProgramGenerator(TProgram, TOutput) type exposes the following members.

### Constructors

Name	Description
<a href="#">StochasticProgramGenerator(TProgram, TOutput)(IDictionary(IProgramGenerator(TProgram, TOutput), Double))</a>	Creates a new StochasticProgramGenerator(TProgram, TOutput) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.
<a href="#">StochasticProgramGenerator(TProgram, TOutput)(IList(IProgramGenerator(TProgram, TOutput)))</a>	Creates a new StochasticProgramGenerator(TProgram, TOutput) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

## Methods

	<b>Name</b>	<b>Description</b>
	<a href="#">Dispose</a>	Releases all resources used by the StochasticProgramGenerator(TProgram, TOutput)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Generation Namespace](#)

## StochasticProgramGenerator(TProgram, TOutput) Constructor

### Overload List

Name	Description
 <a href="#">StochasticProgramGenerator(TProgram, TOutput)(IDictionary(IProgramGenerator(TProgram, TOutput), Double))</a>	Creates a new <a href="#">StochasticProgramGenerator(TProgram, TOutput)</a> with the given list of operators to choose from. Operators will be selected according to the corresponding probability.
 <a href="#">StochasticProgramGenerator(TProgram, TOutput)(IList(IProgramGenerator(TProgram, TOutput)))</a>	Creates a new <a href="#">StochasticProgramGenerator(TProgram, TOutput)</a> with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

### See Also

[StochasticProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## **StochasticProgramGenerator(TProgram, TOutput) Constructor (IDictionary(IProgramGenerator(TProgram, TOutput), Double))**

Creates a new [StochasticProgramGenerator\(TProgram, TOutput\)](#) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public StochasticProgramGenerator(  
    IDictionary<IProgramGenerator<TProgram, TOutput>, double> possibleGenerators  
)
```

[View Source](#)

### Parameters

possibleGenerators

Type: [System.Collections.Generic.IDictionary\(IProgramGenerator\(TProgram, TOutput\), Double\)](#)

A list of generator operators to be used by this operator and the corresponding selection probabilities.

### See Also

[StochasticProgramGenerator\(TProgram, TOutput\)Class](#)

[StochasticProgramGenerator\(TProgram, TOutput\)Overload](#)

[Genetica.Operators.Generation Namespace](#)

## StochasticProgramGenerator(TProgram, TOutput) Constructor (IList(IProgramGenerator(TProgram, TOutput)))

Creates a new [StochasticProgramGenerator\(TProgram, TOutput\)](#) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public StochasticProgramGenerator(  
    IList<IProgramGenerator<TProgram, TOutput>> possibleGenerators  
)
```

[View Source](#)

### Parameters

possibleGenerators

Type: [System.Collections.Generic.IList\(IProgramGenerator\(TProgram, TOutput\)\)](#)

The list of generator operators to be used by this operator.

### See Also

[StochasticProgramGenerator\(TProgram, TOutput\)Class](#)

[StochasticProgramGenerator\(TProgram, TOutput\)Overload](#)

[Genetica.Operators.Generation Namespace](#)

## StochasticProgramGenerator(TProgram, TOutput).StochasticProgramGenerator(TProgram, TOutput) Methods

The [StochasticProgramGenerator\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Generate</a>	Generates a new program having some given maximum depth by randomly combining elements from the given <a href="#">PrimitiveSet(TProgram)</a>
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[StochasticProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## [StochasticProgramGenerator\(TProgram, TOutput\).Dispose Method](#)

Releases all resources used by the [StochasticProgramGenerator\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[StochasticProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## StochasticProgramGenerator(TProgram, TOutput).Generate Method

Generates a new program having some given maximum depth by randomly combining elements from the given [PrimitiveSet\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Generation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Generate(  
    PrimitiveSet<TProgram> primitives,  
    uint maxDepth  
)
```

[View Source](#)

#### Parameters

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The set from which to select primitives.

*maxDepth*

Type: [System.UInt32](#)

The maximum depth of the program generated.

#### Return Value

Type: *TProgram*

The new generated program.

#### Implements

[IProgramGenerator\(TProgram, TOutput\).Generate\(PrimitiveSet\(TProgram\), UInt32\)](#)

### See Also

[StochasticProgramGenerator\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Generation Namespace](#)

## Genetica.Operators.Mutation Namespace

### Classes

Class	Description
 <a href="#">FitnessSimplifyMutation</a>	Represents a <a href="#">IMutationOperator(TProgram)</a> for <a href="#">MathProgram</a> . This mutation operator tries to simplify (shorten the expression of) a given program by removing descendant programs that do not affect its fitness by some degree.
 <a href="#">HoistMutation(TProgram, TOutput)</a>	Represents a <a href="#">IMutationOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . This operator selects a random sub-program of a given program.
 <a href="#">PointMutation(TProgram, TOutput)</a>	Represents a <a href="#">IMutationOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . Allows the mutation of programs by randomly replacing each sub-program by one program with the same arity taken from some <a href="#">PrimitiveSet(TProgram)</a> .
 <a href="#">ShrinkMutation(TProgram, TOutput)</a>	Represents a <a href="#">IMutationOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . This mutation operator removes a random descendant node of a given program and replaces it with a random terminal program from a given <a href="#">PrimitiveSet(TProgram)</a> .
 <a href="#">SimplifyMutation</a>	Represents a <a href="#">IMutationOperator(TProgram)</a> for <a href="#">MathProgram</a> . This mutation operator simplifies (tries to shorten the expression of) a random descendant program of a given program.
 <a href="#">StochasticMutation(TProgram)</a>	Represents a generic <a href="#">IMutationOperator(TProgram)</a> that selects from a list of mutation operators at random.
 <a href="#">SubtreeMutation(TProgram, TOutput)</a>	Represents a <a href="#">IMutationOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . Represents a mutation operator that replaces one sub-program of a given program by a new random program generated using some <a href="#">IProgramGenerator(TProgram, TOutput)</a> .
 <a href="#">SwapMutation(TProgram, TOutput)</a>	Represents a <a href="#">IMutationOperator(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . This operator mutates a given program by swapping (reversing the order of) the children of a randomly-selected function sub-program.

### Interfaces

Interface	Description
 <a href="#">IMutationOperator(TProgram)</a>	An interface for mutation operators for <a href="#">IProgram(TInput, TOutput)</a> , i.e., operators that take one program and create a new one by changing some sub-program in a certain way.



## FitnessSimplifyMutation Class

Represents a [IMutationOperator\(TProgram\)](#) for [MathProgram](#). This mutation operator tries to simplify (shorten the expression of) a given program by removing descendant programs that do not affect its fitness by some degree.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.FitnessSimplifyMutation

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class FitnessSimplifyMutation : IMutationOperator<MathProgram>,
    IDisposable
```

[View Source](#)

The **FitnessSimplifyMutation** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">FitnessSimplifyMutation</a>	Creates a new <b>FitnessSimplifyMutation</b> with the given fitness function.

### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the <b>FitnessSimplifyMutation</b>
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Mutate</a>	Mutates the given <a href="#">TProgram</a> by creating a new one based on some change of one of its sub-programs.
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

**See Also**

[Genetica.Operators.Mutation Namespace](#)

## FitnessSimplifyMutation Constructor

Creates a new [FitnessSimplifyMutation](#) with the given fitness function.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public FitnessSimplifyMutation(  
    IFitnessFunction<MathProgram> fitnessFunction,  
    double epsilon  
)
```

[View Source](#)

#### Parameters

*fitnessFunction*

Type: [Genetica.Evaluation.IFitnessFunction\(MathProgram\)](#)

The fitness function used for simplification.

*epsilon*

Type: [System.Double](#)

The acceptable difference between the fitness of the given program and that of a simplified program for them to be considered equivalent.

### See Also

[FitnessSimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## FitnessSimplifyMutation.FitnessSimplifyMutation Methods

The [FitnessSimplifyMutation](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">Dispose</a>	
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by creating a new one based on some change of one of its sub-programs.
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[FitnessSimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## FitnessSimplifyMutation.Dispose Method

Releases all resources used by the [FitnessSimplifyMutation](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[FitnessSimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## FitnessSimplifyMutation.GetAllMutations Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<MathProgram> GetAllMutations(  
    MathProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(MathProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### See Also

[FitnessSimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## FitnessSimplifyMutation.Mutate Method

Mutates the given *TProgram* by creating a new one based on some change of one of its sub-programs.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public MathProgram Mutate(  
    MathProgram program  
)
```

[View Source](#)

### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to mutate.

### Return Value

Type: [MathProgram](#)

A new *TProgram* based on some change of one of the given program's sub-programs.

### Implements

[IMutationOperator\(TProgram\).Mutate\(TProgram\)](#)

### See Also

[FitnessSimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## **HoistMutation(TProgram, TOutput) Class**

Represents a [IMutationOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). This operator selects a random sub-program of a given program.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.HoistMutation(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class HoistMutation<TProgram, TOutput> : IMutationOperator<TProgram>,
    IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The HoistMutation(TProgram, TOutput) type exposes the following members.

#### Constructors

	Name	Description
	<a href="#">HoistMutation(TProgram, TOutput)</a>	Initializes a new instance of the HoistMutation(TProgram, TOutput) class

#### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the HoistMutation(TProgram, TOutput)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by selecting one of its sub-programs.
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Mutation Namespace](#)

## [HoistMutation\(TProgram, TOutput\) Constructor](#)

Initializes a new instance of the [HoistMutation\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public HoistMutation()
```

[View Source](#)

### See Also

[HoistMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## **HoistMutation(TProgram, TOutput).HoistMutation(TProgram, TOutput) Methods**

The [HoistMutation\(TProgram, TOutput\)](#) generic type exposes the following members.

### **Methods**

	<b>Name</b>	<b>Description</b>
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Mutate</a>	Mutates the given <a href="#">TProgram</a> by selecting one of its sub-programs.
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### **See Also**

[HoistMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## **HoistMutation(TProgram, TOutput).Dispose Method**

Releases all resources used by the [HoistMutation\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[HoistMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## [HoistMutation\(TProgram, TOutput\).GetAllMutations Method](#)

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> GetAllMutations(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### See Also

[HoistMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## [HoistMutation\(TProgram, TOutput\).Mutate Method](#)

Mutates the given *TProgram* by selecting one of its sub-programs.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Mutate(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: *TProgram*

A sub-program of the given *TProgram*.

#### Implements

[IMutationOperator\(TProgram\).Mutate\(TProgram\)](#)

### See Also

[HoistMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## IMutationOperator(TProgram) Interface

An interface for mutation operators for [IProgram\(TInput, TOutput\)](#), i.e., operators that take one program and create a new one by changing some sub-program in a certain way.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IMutationOperator<TProgram> : IDisposable  
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

*TProgram*

The type of program.

The IMutationOperator(TProgram) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by creating a new one based on some change of one of its sub-programs.

### See Also

[Genetica.Operators.Mutation Namespace](#)

## [IMutationOperator\(TProgram\).IMutationOperator\(TProgram\)](#) Methods

The [IMutationOperator\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by creating a new one based on some change of one of its sub-programs.

### See Also

[IMutationOperator\(TProgram\)Interface](#)  
[Genetica.Operators.Mutation Namespace](#)

## [IMutationOperator\(TProgram\).GetAllMutations](#) Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
IEnumerable<TProgram> GetAllMutations(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

### See Also

[IMutationOperator\(TProgram\)Interface](#)

[Genetica.Operators.Mutation Namespace](#)

## [IMutationOperator\(TProgram\).Mutate Method](#)

Mutates the given *TProgram* by creating a new one based on some change of one of its sub-programs.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
TProgram Mutate(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: *TProgram*

A new *TProgram* based on some change of one of the given program's sub-programs.

#### See Also

[IMutationOperator\(TProgram\)Interface](#)

[Genetica.Operators.Mutation Namespace](#)

## PointMutation(TProgram, TOutput) Class

Represents a [IMutationOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). Allows the mutation of programs by randomly replacing each sub-program by one program with the same arity taken from some [PrimitiveSet\(TProgram\)](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.PointMutation(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class PointMutation<TProgram, TOutput> : IMutationOperator<TProgram>,
    IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

**TProgram**

The type of program.

**TOutput**

The type of program output.

The PointMutation(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">PointMutation(TProgram, TOutput)</a>	Creates a new PointMutation(TProgram, TOutput) with the given mutation probability and primitive set.

### Properties

	Name	Description
	<a href="#">MutationProbability</a>	Gets or sets the probability of mutating each sub-program.

### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the PointMutation(TProgram, TOutput)

 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by randomly replacing each sub-program by one program with the same arity taken from the defined <a href="#">PrimitiveSet(TProgram)</a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Mutation Namespace](#)

## PointMutation(TProgram, TOutput) Constructor

Creates a new [PointMutation\(TProgram, TOutput\)](#) with the given mutation probability and primitive set.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public PointMutation(  
    PrimitiveSet<TProgram> primitives,  
    double mutationProbability = 0.5  
)
```

[View Source](#)

### Parameters

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The primitive set to be used in mutation operations.

*mutationProbability* (Optional)

Type: [System.Double](#)

The probability of mutating each sub-program.

### See Also

[PointMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## PointMutation(*TProgram*, *TOutput*).PointMutation(*TProgram*, *TOutput*) Properties

The [PointMutation\(\*TProgram\*, \*TOutput\*\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">MutationProbability</a>	Gets or sets the probability of mutating each sub-program.

### See Also

[PointMutation\(\*TProgram\*, \*TOutput\*\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## PointMutation(TProgram, TOutput).MutationProbability Property

Gets or sets the probability of mutating each sub-program.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double MutationProbability { get; set; }
```

[View Source](#)

*Property Value*

Type: [Double](#)

### See Also

[PointMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## PointMutation(*TProgram*, *TOutput*).PointMutation(*TProgram*, *TOutput*) Methods

The [PointMutation\(\*TProgram\*, \*TOutput\*\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by randomly replacing each sub-program by one program with the same arity taken from the defined <a href="#">PrimitiveSet(<i>TProgram</i>)</a> .
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[PointMutation\(\*TProgram\*, \*TOutput\*\)Class](#)  
[Genetica.Operators.Mutation Namespace](#)

## PointMutation(TProgram, TOutput).Dispose Method

Releases all resources used by the [PointMutation\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[PointMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## PointMutation(*TProgram*, *TOutput*).GetAllMutations Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public IEnumerable<TProgram> GetAllMutations(
    TProgram program
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### See Also

[PointMutation\(\*TProgram\*, \*TOutput\*\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## PointMutation(*TProgram*, *TOutput*).Mutate Method

Mutates the given *TProgram* by randomly replacing each sub-program by one program with the same arity taken from the defined [PrimitiveSet\(\*TProgram\*\)](#).

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Mutate(  
    TProgram program  
)
```

[View Source](#)

### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

### Return Value

Type: *TProgram*

A new *TProgram* created by randomly replacing each sub-program by one program with the same arity taken from the defined [PrimitiveSet\(\*TProgram\*\)](#).

### Implements

[IMutationOperator\(\*TProgram\*\).Mutate\(\*TProgram\*\)](#)

### See Also

[PointMutation\(\*TProgram\*, \*TOutput\*\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## ShrinkMutation(TProgram, TOutput) Class

Represents a [IMutationOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). This mutation operator removes a random descendant node of a given program and replaces it with a random terminal program from a given [PrimitiveSet\(TProgram\)](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.ShrinkMutation(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class ShrinkMutation<TProgram, TOutput> : IMutationOperator<TProgram>,
    IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

[Missing <typeparam name="TProgram"/> documentation for  
"T:Genetica.Operators.Mutation.ShrinkMutation`2"]

TOutput

[Missing <typeparam name="TOutput"/> documentation for  
"T:Genetica.Operators.Mutation.ShrinkMutation`2"]

The ShrinkMutation(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
≡	<a href="#">ShrinkMutation(TProgram, TOutput)</a>	Creates a new ShrinkMutation(TProgram, TOutput) with the given primitives.

### Methods

	Name	Description
≡	<a href="#">Dispose</a>	Releases all resources used by the ShrinkMutation(TProgram, TOutput)
≡	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <a href="#">TProgram</a> by removing one of its sub-programs at random and replacing it with a random terminal program from a given <a href="#">PrimitiveSet(TProgram)</a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Mutation Namespace](#)

## ShrinkMutation(TProgram, TOutput) Constructor

Creates a new [ShrinkMutation\(TProgram, TOutput\)](#) with the given primitives.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ShrinkMutation(  
    PrimitiveSet<TProgram> primitives  
)
```

[View Source](#)

#### Parameters

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The primitive set to be used in mutation operations.

#### See Also

[ShrinkMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## ShrinkMutation(*TProgram*, *TOutput*).ShrinkMutation(*TProgram*, *TOutput*) Methods

The [ShrinkMutation\(\*TProgram\*, \*TOutput\*\)](#) generic type exposes the following members.

### Methods

Name	Description
 <a href="#">Dispose</a>	
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by removing one of its sub-programs at random and replacing it with a random terminal program from a given <a href="#">PrimitiveSet(<i>TProgram</i>)</a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[ShrinkMutation\(\*TProgram\*, \*TOutput\*\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## ShrinkMutation(TProgram, TOutput).Dispose Method

Releases all resources used by the [ShrinkMutation\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[ShrinkMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## ShrinkMutation(*TProgram*, *TOutput*).GetAllMutations Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> GetAllMutations(
    TProgram program
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### See Also

[ShrinkMutation\(\*TProgram\*, \*TOutput\*\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## ShrinkMutation(*TProgram*, *TOutput*).Mutate Method

Mutates the given *TProgram* by removing one of its sub-programs at random and replacing it with a random terminal program from a given [PrimitiveSet\(\*TProgram\*\)](#).

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Mutate(  
    TProgram program  
)
```

[View Source](#)

### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

### Return Value

Type: *TProgram*

A new *TProgram* based on some change of one of the given program's sub-programs.

### Implements

[IMutationOperator\(\*TProgram\*\).Mutate\(\*TProgram\*\)](#)

### See Also

[ShrinkMutation\(\*TProgram\*, \*TOutput\*\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SimplifyMutation Class

Represents a [IMutationOperator\(TProgram\)](#) for [MathProgram](#). This mutation operator simplifies (tries to shorten the expression of) a random descendant program of a given program.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.SimplifyMutation

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SimplifyMutation : IMutationOperator<MathProgram>,  
    IDisposable
```

[View Source](#)

The **SimplifyMutation** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SimplifyMutation</a>	Initializes a new instance of the <b>SimplifyMutation</b> class

### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the <b>SimplifyMutation</b>
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Mutate</a>	Mutates the given <a href="#">MathProgram</a> by simplifying a random sub-program of the given <a href="#">MathProgram</a> .
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

**See Also**

[Genetica.Operators.Mutation Namespace](#)

## SimplifyMutation Constructor

Initializes a new instance of the [SimplifyMutation](#) class

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SimplifyMutation()
```

[View Source](#)

### See Also

[SimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SimplifyMutation.SimplifyMutation Methods

The [SimplifyMutation](#) type exposes the following members.

### Methods

Name	Description
 <a href="#">Dispose</a>	
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <a href="#">MathProgram</a> by simplifying a random sub-program of the given <a href="#">MathProgram</a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[SimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SimplifyMutation.Dispose Method

Releases all resources used by the [SimplifyMutation](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[SimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SimplifyMutation.GetAllMutations Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<MathProgram> GetAllMutations(  
    MathProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(MathProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### See Also

[SimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SimplifyMutation.Mutate Method

Mutates the given [MathProgram](#) by simplifying a random sub-program of the given [MathProgram](#).

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public MathProgram Mutate(  
    MathProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: [Genetica.Elements.MathProgram](#)

The program we want to mutate.

#### Return Value

Type: [MathProgram](#)

A new [MathProgram](#) resulting of the simplification of one of the sub-programs.

#### Implements

[IMutationOperator\(TProgram\).Mutate\(TProgram\)](#)

### See Also

[SimplifyMutation Class](#)

[Genetica.Operators.Mutation Namespace](#)

## StochasticMutation(TProgram) Class

Represents a generic [IMutationOperator\(TProgram\)](#) that selects from a list of mutation operators at random.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.StochasticMutation(TProgram)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class StochasticMutation<TProgram> : IMutationOperator<TProgram>,
    IDisposable
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

The StochasticMutation(TProgram) type exposes the following members.

### Constructors

Name	Description
 <a href="#">StochasticMutation(TProgram)(ICollection(IMutationOperator(TProgram)))</a>	Creates a new StochasticMutation(TProgram) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).
 <a href="#">StochasticMutation(TProgram)(IDictionary(IMutationOperator(TProgram), Double))</a>	Creates a new StochasticMutation(TProgram) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

## Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the StochasticMutation(TProgram)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Mutate</a>	Randomly chooses one of the <a href="#">IMutationOperator(TProgram)s</a> to perform the mutation.
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Mutation Namespace](#)

## StochasticMutation(TProgram) Constructor

### Overload List

Name	Description
<a href="#">StochasticMutation(TProgram)(ICollection(IMutationOperator(TProgram)))</a>	Creates a new <a href="#">StochasticMutation(TProgram)</a> with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).
<a href="#">StochasticMutation(TProgram)(IDictionary(IMutationOperator(TProgram), Double))</a>	Creates a new <a href="#">StochasticMutation(TProgram)</a> with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

### See Also

[StochasticMutation\(TProgram\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## **StochasticMutation(TProgram) Constructor (ICollection(IMutationOperator(TProgram)))**

Creates a new [StochasticMutation\(TProgram\)](#) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public StochasticMutation(  
    ICollection<IMutationOperator<TProgram>> possibleMutations  
)
```

[View Source](#)

### Parameters

*possibleMutations*

Type: [System.Collections.Generic.ICollection\(IMutationOperator\(TProgram\)\)](#)

The list of mutation operators to be used by this operator.

### See Also

[StochasticMutation\(TProgram\)Class](#)

[StochasticMutation\(TProgram\)Overload](#)

[Genetica.Operators.Mutation Namespace](#)

## **StochasticMutation(TProgram) Constructor (IDictionary(IMutationOperator(TProgram), Double))**

Creates a new [StochasticMutation\(TProgram\)](#) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public StochasticMutation(  
    IDictionary<IMutationOperator<TProgram>, double> possibleMutations  
)
```

[View Source](#)

### Parameters

*possibleMutations*

Type: [System.Collections.Generic.IDictionary\(IMutationOperator\(TProgram\), Double\)](#)

A list of mutation operators to be used by this operator and the corresponding selection probabilities.

### See Also

[StochasticMutation\(TProgram\)Class](#)

[StochasticMutation\(TProgram\)Overload](#)

[Genetica.Operators.Mutation Namespace](#)

## StochasticMutation(TProgram).StochasticMutation(TProgram) Methods

The [StochasticMutation\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Mutate</a>	Randomly chooses one of the <a href="#">IMutationOperator(TProgram)s</a> to perform the mutation.
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[StochasticMutation\(TProgram\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## [StochasticMutation\(TProgram\).Dispose Method](#)

Releases all resources used by the [StochasticMutation\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

Implements

[IDisposable.Dispose\(\)](#)

### See Also

[StochasticMutation\(TProgram\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## StochasticMutation(TProgram).GetAllMutations Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> GetAllMutations(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### See Also

[StochasticMutation\(TProgram\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## StochasticMutation(TProgram).Mutate Method

Randomly chooses one of the [IMutationOperator\(TProgram\)](#)s to perform the mutation.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Mutate(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: [TProgram](#)

An *program* to be mutated.

#### Return Value

Type: [TProgram](#)

A new *program* corresponding to the given *program* mutated.

#### Implements

[IMutationOperator\(TProgram\).Mutate\(TProgram\)](#)

### See Also

[StochasticMutation\(TProgram\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SubtreeMutation(TProgram, TOutput) Class

Represents a [IMutationOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). Represents a mutation operator that replaces one sub-program of a given program by a new random program generated using some [IProgramGenerator\(TProgram, TOutput\)](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.SubtreeMutation(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SubtreeMutation<TProgram, TOutput> : IMutationOperator<TProgram>,
    IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The SubtreeMutation(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SubtreeMutation(TProgram, TOutput)</a>	Creates anew SubtreeMutation(TProgram, TOutput) with the given arguments.

### Properties

	Name	Description
	<a href="#">MaxDepth</a>	Gets or sets the maximum depth of new random sub-programs.

### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the SubtreeMutation(TProgram, TOutput)

 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by replacing one of its sub-programs by a new random program generated using the defined <a href="#"><i>IProgramGenerator(TProgram, TOutput)</i></a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Mutation Namespace](#)

## SubtreeMutation(TProgram, TOutput) Constructor

Creates anew [SubtreeMutation\(TProgram, TOutput\)](#) with the given arguments.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public SubtreeMutation(  
    IProgramGenerator<TProgram, TOutput> programGenerator,  
    PrimitiveSet<TProgram> primitives,  
    uint maxDepth  
)
```

[View Source](#)

### Parameters

*programGenerator*

Type: [Genetica.Operators.Generation.IProgramGenerator\(TProgram, TOutput\)](#)

The generator for new sub-programs.

*primitives*

Type: [Genetica.Elements.PrimitiveSet\(TProgram\)](#)

The primitive set to be used in mutation operations.

*maxDepth*

Type: [System.UInt32](#)

The maximum depth of new random sub-programs.

### See Also

[SubtreeMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SubtreeMutation(TProgram, TOutput).SubtreeMutation(TProgram, TOutput) Properties

The [SubtreeMutation\(TProgram, TOutput\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">MaxDepth</a>	Gets or sets the maximum depth of new random sub-programs.

### See Also

[SubtreeMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SubtreeMutation(TProgram, TOutput).MaxDepth Property

Gets or sets the maximum depth of new random sub-programs.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint MaxDepth { get; set; }
```

[View Source](#)

**Property Value**

Type: [UInt32](#)

### See Also

[SubtreeMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SubtreeMutation(TProgram, TOutput).SubtreeMutation(TProgram, TOutput) Methods

The [SubtreeMutation\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

Name	Description
 <a href="#">Dispose</a>	
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <a href="#">TProgram</a> by replacing one of its sub-programs by a new random program generated using the defined <a href="#">IProgramGenerator(TProgram, TOutput)</a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[SubtreeMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## **SubtreeMutation(TProgram, TOutput).Dispose Method**

Releases all resources used by the [SubtreeMutation\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[SubtreeMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SubtreeMutation(TProgram, TOutput).GetAllMutations Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> GetAllMutations(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### Remarks

Because there may be a huge number of possible mutations resulting from the use of this operator, only the primitives (functions + terminals) are considered, i.e., new programs of depth 0 or 1.

### See Also

[SubtreeMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SubtreeMutation(*TProgram*, *TOutput*).Mutate Method

Mutates the given *TProgram* by replacing one of its sub-programs by a new random program generated using the defined [IProgramGenerator\(\*TProgram\*, \*TOutput\*\)](#).

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Mutate(  
    TProgram program  
)
```

[View Source](#)

### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

### Return Value

Type: *TProgram*

A new *TProgram* by replacing one of its sub-programs by a new random program generated using the defined [IProgramGenerator\(\*TProgram\*, \*TOutput\*\)](#).

### Implements

[IMutationOperator\(\*TProgram\*\).Mutate\(\*TProgram\*\)](#)

### See Also

[SubtreeMutation\(\*TProgram\*, \*TOutput\*\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SwapMutation(TProgram, TOutput) Class

Represents a [IMutationOperator\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). This operator mutates a given program by swapping (reversing the order of) the children of a randomly-selected function sub-program.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Mutation.SwapMutation(TProgram, TOutput)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SwapMutation<TProgram, TOutput> : IMutationOperator<TProgram>,
    IDisposable
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The SwapMutation(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SwapMutation(TProgram, TOutput)</a>	Initializes a new instance of the SwapMutation(TProgram, TOutput) class

### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the SwapMutation(TProgram, TOutput)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.

 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Mutate</a>	Mutates the given <i>TProgram</i> by swapping (reversing the order of) the children of a randomly-selected function sub-program.
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Mutation Namespace](#)

## SwapMutation(TProgram, TOutput) Constructor

Initializes a new instance of the [SwapMutation\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SwapMutation()
```

[View Source](#)

### See Also

[SwapMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SwapMutation(TProgram, TOutput).SwapMutation(TProgram, TOutput) Methods

The [SwapMutation\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetAllMutations</a>	Gets a list containing all possible programs resulting from applying this mutation operator.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Mutate</a>	Mutates the given <a href="#">TProgram</a> by swapping (reversing the order of) the children of a randomly-selected function sub-program.
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[SwapMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SwapMutation(TProgram, TOutput).Dispose Method

Releases all resources used by the [SwapMutation\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[SwapMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SwapMutation(TProgram, TOutput).GetAllMutations Method

Gets a list containing all possible programs resulting from applying this mutation operator.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> GetAllMutations(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list containing all possible programs resulting from applying this mutation operator.

#### Implements

[IMutationOperator\(TProgram\).GetAllMutations\(TProgram\)](#)

### See Also

[SwapMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## SwapMutation(TProgram, TOutput).Mutate Method

Mutates the given *TProgram* by swapping (reversing the order of) the children of a randomly-selected function sub-program.

**Namespace:** [Genetica.Operators.Mutation](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TProgram Mutate(  
    TProgram program  
)
```

[View Source](#)

### Parameters

*program*

Type: *TProgram*

The program we want to mutate.

### Return Value

Type: *TProgram*

A new *TProgram* by swapping (reversing the order of) the children of a randomly-selected function sub-program.

### Implements

[IMutationOperator\(TProgram\).Mutate\(TProgram\)](#)

### See Also

[SwapMutation\(TProgram, TOutput\)Class](#)

[Genetica.Operators.Mutation Namespace](#)

## Genetica.Operators.Selection Namespace

### Classes

Class	Description
 <a href="#">PopulationSelectors</a>	Provides methods that can be used as <a href="#">PopulationSelector</a> delegates.
 <a href="#">RouletteWheelSelection(TProgram)</a>	Represents a <a href="#">ISelectionOperator(TProgram)</a> that performs fitness proportionate (roulette wheel) selection. The operator works by selecting $n$ programs, where $n$ is the size of the given population. The operator assigns a selection probability to each program (a proportion of the wheel) according to their fitness score, as dictated by some <a href="#">IFitnessFunction(TProgram)</a> . The proportion is decided according to some <a href="#">PopulationSelector</a> scheme. While programs with a higher fitness will be more likely selected, there is still a chance that some weaker programs may survive the selection process.
 <a href="#">StochasticSelection(TProgram)</a>	Represents a generic <a href="#">ISelectionOperator(TProgram)</a> that selects from a list of selection operators at random.
 <a href="#">TournamentSelection(TProgram)</a>	Represents a <a href="#">ISelectionOperator(TProgram)</a> that performs tournament selection. The operator works by performing $n$ tournaments, where $n$ is the size of the given population. For each tournament, the operator selects $m$ programs from the population at random, and the program attaining maximal score wins the tournament and gets picked to the selection group.

### Interfaces

Interface	Description
 <a href="#">ISelectionOperator(TProgram)</a>	An interface for selection operators for <a href="#">IProgram(TInput, TOutput)</a> , i.e., operators that select a certain number of programs from a given <a href="#">IPopulation(TProgram)</a> .

### Delegates

Delegate	Description
 <a href="#">PopulationSelector</a>	Represents a delegate for methods of assigning proportions of selection to some <a href="#">IPopulation(TProgram)</a> to be used in conjunction with some <a href="#">ISelectionOperator(TProgram)</a> .

## ISelectionOperator(TProgram) Interface

An interface for selection operators for [IProgram\(TInput, TOutput\)](#), i.e., operators that select a certain number of programs from a given [IPopulation\(TProgram\)](#).

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface ISelectionOperator<TProgram> : IDisposable  
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

*TProgram*

The type of program.

The ISelectionOperator(TProgram) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .

### See Also

[Genetica.Operators.Selection Namespace](#)

## ISelectionOperator(TProgram).ISelectionOperator(TProgram) Methods

The [ISelectionOperator\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	(Inherited from <a href="#">IDisposable</a> .)
	<a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .

### See Also

[ISelectionOperator\(TProgram\)Interface](#)  
[Genetica.Operators.Selection Namespace](#)

## ISelectionOperator(TProgram).Select Method

Performs the selection operation over the given [IPopulation\(TProgram\)](#).

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
IEnumerable<TProgram> Select(  
    IPopulation<TProgram> population  
)
```

[View Source](#)

#### Parameters

*population*

Type: [Genetica.IPopulation\(TProgram\)](#)

The population over which to perform selection.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list of *TProgram* resulting from the selection operation.

### See Also

[ISelectionOperator\(TProgram\)Interface](#)

[Genetica.Operators.Selection Namespace](#)

## PopulationSelector Delegate

Represents a delegate for methods of assigning proportions of selection to some [IPopulation\(TProgram\)](#) to be used in conjunction with some [ISelectionOperator\(TProgram\)](#).

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public delegate double[] PopulationSelector(  
    uint numPointers  
)
```

#### Parameters

numPointers

Type: [System.UInt32](#)

The number of pointers for the selection.

#### Return Value

Type: [Double\[\]](#)

The array of selection pointers

### See Also

[Genetica.Operators.Selection Namespace](#)

## PopulationSelectors Class

Provides methods that can be used as [PopulationSelector](#) delegates.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Selection.PopulationSelectors

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static class PopulationSelectors
```

[View Source](#)

The **PopulationSelectors** type exposes the following members.

### Methods

	Name	Description
	<a href="#">RandomSelector</a>	Gets an array of pointers that are unevenly (randomly) spread.
	<a href="#">UniformSelector</a>	Gets an array of pointers that are equally spread.

### See Also

[Genetica.Operators.Selection Namespace](#)

## PopulationSelectors.PopulationSelectors Methods

The [PopulationSelectors](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">RandomSelector</a>	Gets an array of pointers that are unevenly (randomly) spread.
	<a href="#">UniformSelector</a>	Gets an array of pointers that are equally spread.

### See Also

[PopulationSelectors Class](#)

[Genetica.Operators.Selection Namespace](#)

## PopulationSelectors.RandomSelector Method

Gets an array of pointers that are unevenly (randomly) spread.

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static double[] RandomSelector(  
    uint numPointers  
)
```

[View Source](#)

#### Parameters

numPointers

Type: [System.UInt32](#)

The number of pointers for the selection.

#### Return Value

Type: [Double\[\]](#)

The array of selection pointers

### See Also

[PopulationSelectors Class](#)

[Genetica.Operators.Selection Namespace](#)

## PopulationSelectors.UniformSelector Method

Gets an array of pointers that are equally spread.

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static double[] UniformSelector(  
    uint numPointers  
)
```

[View Source](#)

#### Parameters

numPointers

Type: [System.UInt32](#)

[Missing <param name="numPointers"/> documentation for  
"M:Genetica.Operators.Selection.PopulationSelectors.UniformSelector(System.UInt32)"]

#### Return Value

Type: [Double](#)[]

The array of selection pointers

### See Also

[PopulationSelectors Class](#)

[Genetica.Operators.Selection Namespace](#)

## RouletteWheelSelection(TProgram) Class

Represents a [ISelectionOperator\(TProgram\)](#) that performs fitness proportionate (roulette wheel) selection. The operator works by selecting  $n$  programs, where  $n$  is the size of the given population. The operator assigns a selection probability to each program (a proportion of the wheel) according to their fitness score, as dictated by some [IFitnessFunction\(TProgram\)](#). The proportion is decided according to some [PopulationSelector](#) scheme. While programs with a higher fitness will be more likely selected, there is still a chance that some weaker programs may survive the selection process.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Selection.RouletteWheelSelection(TProgram)

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public class RouletteWheelSelection<TProgram> : ISelectionOperator<TProgram>,
    IDisposable
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

**TProgram**

The type of program.

The RouletteWheelSelection(TProgram) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">RouletteWheelSelection(TProgram)</a>	Creates a new RouletteWheelSelection(TProgram) with the given arguments.

### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the RouletteWheelSelection(TProgram)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Selection Namespace](#)

## RouletteWheelSelection(TProgram) Constructor

Creates a new [RouletteWheelSelection\(TProgram\)](#) with the given arguments.

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public RouletteWheelSelection(  
    PopulationSelector selector,  
    IFitnessFunction<TProgram> fitnessFunction  
)
```

[View Source](#)

### Parameters

*selector*

Type: [Genetica.Operators.Selection.PopulationSelector](#)

The scheme used to attribute proportion of selection.

*fitnessFunction*

Type: [Genetica.Evaluation.IFitnessFunction\(TProgram\)](#)

The function used to evaluate the programs' fitness.

### See Also

[RouletteWheelSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## RouletteWheelSelection(TProgram).RouletteWheelSelection(TProgram) Methods

The [RouletteWheelSelection\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[RouletteWheelSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## RouletteWheelSelection(TProgram).Dispose Method

Releases all resources used by the [RouletteWheelSelection\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[RouletteWheelSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## RouletteWheelSelection(TProgram).Select Method

Performs the selection operation over the given [IPopulation\(TProgram\)](#).

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> Select(  
    IPopulation<TProgram> population  
)
```

[View Source](#)

#### Parameters

*population*

Type: [Genetica.IPopulation\(TProgram\)](#)

The population over which to perform selection.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list of *TProgram* resulting from the selection operation.

#### Implements

[ISelectionOperator\(TProgram\).Select\(IPopulation\(TProgram\)\)](#)

### See Also

[RouletteWheelSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## StochasticSelection(TProgram) Class

Represents a generic [ISelectionOperator\(TProgram\)](#) that selects from a list of selection operators at random.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Selection.StochasticSelection(TProgram)

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class StochasticSelection<TProgram> : ISelectionOperator<TProgram>,  
    IDisposable  
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

The StochasticSelection(TProgram) type exposes the following members.

### Constructors

Name	Description
 <a href="#">StochasticSelection(TProgram)(ICollection(ISelectionOperator(TProgram)))</a>	Creates a new StochasticSelection(TProgram) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).
 <a href="#">StochasticSelection(TProgram)(IDictionary(ISelectionOperator(TProgram), Double))</a>	Creates a new StochasticSelection(TProgram) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

## Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the StochasticSelection(TProgram)
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Selection Namespace](#)

## StochasticSelection(TProgram) Constructor

### Overload List

Name	Description
 <a href="#">StochasticSelection(TProgram)(ICollection(ISelectionOperator(TProgram)))</a>	Creates a new <a href="#">StochasticSelection(TProgram)</a> with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).
 <a href="#">StochasticSelection(TProgram)(IDictionary(ISelectionOperator(TProgram), Double))</a>	Creates a new <a href="#">StochasticSelection(TProgram)</a> with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

### See Also

[StochasticSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## StochasticSelection(TProgram) Constructor (ICollection(ISelectionOperator(TProgram)))

Creates a new [StochasticSelection\(TProgram\)](#) with the given list of operators to choose from. All operators will have the same probability of being selected (uniform distribution).

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public StochasticSelection(  
    ICollection<ISelectionOperator<TProgram>> possibleSelections  
)
```

[View Source](#)

### Parameters

`possibleSelections`

Type: [System.Collections.Generic.ICollection\(ISelectionOperator\(TProgram\)\)](#)

The list of selection operators to be used by this operator.

### See Also

[StochasticSelection\(TProgram\)Class](#)

[StochasticSelection\(TProgram\)Overload](#)

[Genetica.Operators.Selection Namespace](#)

## **StochasticSelection(TProgram) Constructor (IDictionary(ISelectionOperator(TProgram), Double))**

Creates a new [StochasticSelection\(TProgram\)](#) with the given list of operators to choose from. Operators will be selected according to the corresponding probability.

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public StochasticSelection(  
    IDictionary<ISelectionOperator<TProgram>, double> possibleSelections  
)
```

[View Source](#)

### Parameters

*possibleSelections*

Type: [System.Collections.Generic.IDictionary\(ISelectionOperator\(TProgram\), Double\)](#)

A list of selection operators to be used by this operator and the corresponding selection probabilities.

### See Also

[StochasticSelection\(TProgram\)Class](#)

[StochasticSelection\(TProgram\)Overload](#)

[Genetica.Operators.Selection Namespace](#)

## StochasticSelection(TProgram).StochasticSelection(TProgram) Methods

The [StochasticSelection\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[StochasticSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## [StochasticSelection\(TProgram\).Dispose Method](#)

Releases all resources used by the [StochasticSelection\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

Implements

[IDisposable.Dispose\(\)](#)

### See Also

[StochasticSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## StochasticSelection(TProgram).Select Method

Performs the selection operation over the given [IPopulation\(TProgram\)](#).

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IEnumerable<TProgram> Select(  
    IPopulation<TProgram> population  
)
```

[View Source](#)

#### Parameters

*population*

Type: [Genetica.IPopulation\(TProgram\)](#)

The population over which to perform selection.

#### Return Value

Type: [IEnumerable\(TProgram\)](#)

A list of *TProgram* resulting from the selection operation.

#### Implements

[ISelectionOperator\(TProgram\).Select\(IPopulation\(TProgram\)\)](#)

### See Also

[StochasticSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## TournamentSelection(TProgram) Class

Represents a [ISelectionOperator\(TProgram\)](#) that performs tournament selection. The operator works by performing  $n$  tournaments, where  $n$  is the size of the given population. For each tournament, the operator selects  $m$  programs from the population at random, and the program attaining maximal score wins the tournament and gets picked to the selection group.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Operators.Selection.TournamentSelection(TProgram)

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class TournamentSelection<TProgram> : ISelectionOperator<TProgram>,
    IDisposable
where TProgram : IProgram
```

[View Source](#)

#### Type Parameters

*TProgram*

The type of program.

The TournamentSelection(TProgram) type exposes the following members.

#### Constructors

	Name	Description
	<a href="#">TournamentSelection(TProgram)</a>	Creates a new TournamentSelection(TProgram) with the given elements.

#### Properties

	Name	Description
	<a href="#">TournamentSize</a>	Gets the size (in number of programs selected from the population) of each tournament to be performed.

#### Methods

	Name	Description
	<a href="#">Dispose</a>	Releases all resources used by the TournamentSelection(TProgram)

 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Operators.Selection Namespace](#)

## TournamentSelection(TProgram) Constructor

Creates a new [TournamentSelection\(TProgram\)](#) with the given elements.

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TournamentSelection(  
    IComparer<TProgram> programComparer,  
    uint tournamentSize = 1  
)
```

[View Source](#)

### Parameters

*programComparer*

Type: [System.Collections.Generic.IComparer\(TProgram\)](#)

The comparer used to check the program attaining maximal score.

*tournamentSize* (Optional)

Type: [System.UInt32](#)

The size of each tournament to be performed.

### See Also

[TournamentSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## TournamentSelection(TProgram).TournamentSelection(TProgram) Properties

The [TournamentSelection\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">TournamentSize</a>	Gets the size (in number of programs selected from the population) of each tournament to be performed.

### See Also

[TournamentSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## TournamentSelection(TProgram).TournamentSize Property

Gets the size (in number of programs selected from the population) of each tournament to be performed.

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint TournamentSize { get; }
```

[View Source](#)

### Property Value

Type: [UInt32](#)

### See Also

[TournamentSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## TournamentSelection(TProgram).TournamentSelection(TProgram) Methods

The [TournamentSelection\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Dispose</a>	
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Select</a>	Performs the selection operation over the given <a href="#">IPopulation(TProgram)</a> .
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[TournamentSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## TournamentSelection(TProgram).Dispose Method

Releases all resources used by the [TournamentSelection\(TProgram\)](#)

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Dispose()
```

[View Source](#)

*Implements*

[IDisposable.Dispose\(\)](#)

### See Also

[TournamentSelection\(TProgram\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## TournamentSelection(*TProgram*).Select Method

Performs the selection operation over the given [IPopulation\(\*TProgram\*\)](#).

**Namespace:** [Genetica.Operators.Selection](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public IEnumerable<TProgram> Select(  
    IPopulation<TProgram> population  
)
```

[View Source](#)

#### Parameters

*population*

Type: [Genetica.IPopulation\(\*TProgram\*\)](#)

The population over which to perform selection.

#### Return Value

Type: [IEnumerable\(\*TProgram\*\)](#)

A list of *TProgram* resulting from the selection operation.

#### Implements

[ISelectionOperator\(\*TProgram\*\).Select\(IPopulation\(\*TProgram\*\)\)](#)

### See Also

[TournamentSelection\(\*TProgram\*\)Class](#)

[Genetica.Operators.Selection Namespace](#)

## Genetica.Similarity Namespace

### Classes

Class	Description
 <a href="#">AverageSimilarity(TProgram)</a>	Represents a <a href="#">ISimilarityMeasure(TProgram)</a> that computes the average similarity computed by other several similarity measures.
 <a href="#">CommonRegionSimilarity(TProgram, TOutput)</a>	Measures the similarity of two <a href="#">ITreeProgram(TOutput)</a> based on the common-region of their expressions' trees. Common-region similarity is given by the division of the number of sub-programs in the common-region and the max number of sub-programs between the two given programs.
 <a href="#">LeafSimilarity(TProgram, TOutput)</a>	Represents a <a href="#">ISimilarityMeasure(TProgram)</a> for <a href="#">ITreeProgram(TOutput)</a> . It measures the similarity of programs based on the leaf nodes of their expressions. Leaf similarity is given by the division between the number of common terminals and the number of all terminals of the two given programs.
 <a href="#">NormalNotationEditSimilarity</a>	Measures the similarity of two <a href="#">IProgram</a> based on the edit or Levenshtein string distance between the expressions of the programs in normal notation, i.e., based on the number of edits needed to transform the expression of one program into the one of the other program.
 <a href="#">PrefixNotationEditSimilarity</a>	Measures the similarity of two <a href="#">MathProgram</a> based on the edit or Levenshtein string distance between the expressions of the programs in prefix notation, i.e., based on the number of edits needed to transform the expression of one program into the one of the other program.
 <a href="#">PrimitiveSimilarity(TProgram, TOutput)</a>	Measures the similarity of <a href="#">ITreeProgram(TOutput)</a> based on the primitives in their expressions. Primitive similarity is given by the division between the number of common primitives and the number of all primitives of the two given programs.
 <a href="#">SubCombinationSimilarity(TProgram, TOutput)</a>	Measures the similarity of <a href="#">ITreeProgram(TOutput)</a> based on their sub-combinations. Sub-combination similarity is given by the division of the number of common sub-combinations in the programs and the total number of sub-combinations.
 <a href="#">SubProgramSimilarity(TProgram, TOutput)</a>	Measures the similarity of <a href="#">ITreeProgram(TOutput)</a> based on their sub-combinations. Sub-program similarity is

		given by the division of the number of common sub-programs of the programs and the total number of sub-programs.
 <a href="#">SymbolTreeSimilarity</a>		Measures the similarity of two <a href="#">ITreeProgram</a> based on the similarity of their <a href="#">SymbolTree(TProgram)</a> .
 <a href="#">TreeEditSimilarity(TProgram, TOutput)</a>		Measures the similarity of two <a href="#">ITreeProgram(TOutput)</a> based on the edit or Levenshtein distance, i.e., based on the number of edits---transformations, additions and deletions---needed to transform the syntactic tree of one program into the one of the other program.
 <a href="#">ValueSimilarity</a>		Measures the similarity of two <a href="#">MathProgram</a> in terms of the values that they can compute when we replace their <a href="#">Constant</a> programs by random variables.

## Interfaces

Interface	Description
 <a href="#">ISimilarityMeasure(TProgram)</a>	Represents an interface for similarity measures between two programs, e.g., based on their tree structure.

## AverageSimilarity(TProgram) Class

Represents a [ISimilarityMeasure\(TProgram\)](#) that computes the average similarity computed by other several similarity measures.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.AverageSimilarity(TProgram)

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class AverageSimilarity<TProgram> : ISimilarityMeasure<TProgram>
where TProgram : IProgram
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

The AverageSimilarity(TProgram) type exposes the following members.

#### Constructors

	Name	Description
≡	<a href="#">AverageSimilarity(TProgram)</a>	Creates a new AverageSimilarity(TProgram) with the given measures.

#### Methods

	Name	Description
≡	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
≡	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

**See Also**

[Genetica.Similarity Namespace](#)

## AverageSimilarity(TProgram) Constructor

Creates a new [AverageSimilarity\(TProgram\)](#) with the given measures.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public AverageSimilarity(  
    IEnumerable<ISimilarityMeasure<TProgram>> measures  
)
```

[View Source](#)

### Parameters

*measures*

Type: [System.Collections.Generic.IEnumerable\(ISimilarityMeasure\(TProgram\)\)](#)

The similarity measures used to compute the average similarity.

### See Also

[AverageSimilarity\(TProgram\)Class](#)

[Genetica.Similarity Namespace](#)

## AverageSimilarity(TProgram).AverageSimilarity(TProgram) Methods

The [AverageSimilarity\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
      	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
      	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[AverageSimilarity\(TProgram\)Class](#)

[Genetica.Similarity Namespace](#)

## AverageSimilarity(TProgram).Calculate Method

Calculates the similarity (or inverse distance) between two TProgram.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: [TProgram](#)

The first program of the comparison.

*prog2*

Type: [TProgram](#)

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between [0](#) and [1](#) representing the calculated similarity between the given programs. A value near [1](#) indicates a high similarity (or low distance) between the two programs, while a value near [0](#) represents a low similarity (or high distance) between the programs. If the programs are the same, it returns [1](#), if any of the programs is [null](#), it returns [0](#).

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[AverageSimilarity\(TProgram\)Class](#)

[Genetica.Similarity Namespace](#)

## CommonRegionSimilarity(TProgram, TOutput) Class

Measures the similarity of two [ITreeProgram\(TOutput\)](#) based on the common-region of their expressions' trees. Common-region similarity is given by the division of the number of sub-programs in the common-region and the max number of sub-programs between the two given programs.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.CommonRegionSimilarity(TProgram, TOutput)

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class CommonRegionSimilarity<TProgram, TOutput> : ISimilarityMeasure<TProgram>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The CommonRegionSimilarity(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">CommonRegionSimilarity(TProgram, TOutput)</a>	Initializes a new instance of the CommonRegionSimilarity(TProgram, TOutput) class

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)

	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Similarity Namespace](#)

## CommonRegionSimilarity(TProgram, TOutput) Constructor

Initializes a new instance of the [CommonRegionSimilarity\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public CommonRegionSimilarity()
```

[View Source](#)

### See Also

[CommonRegionSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## CommonRegionSimilarity(TProgram, TOutput).CommonRegionSimilarity(TProgram, TOutput) Methods

The [CommonRegionSimilarity\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[CommonRegionSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## CommonRegionSimilarity(TProgram, TOutput).Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: *TProgram*

The first program of the comparison.

*prog2*

Type: *TProgram*

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between [0](#) and [1](#) representing the calculated similarity between the given programs. A value near [1](#) indicates a high similarity (or low distance) between the two programs, while a value near [0](#) represents a low similarity (or high distance) between the programs. If the programs are the same, it returns [1](#), if any of the programs is [null](#), it returns [0](#).

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[CommonRegionSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## ISimilarityMeasure(TProgram) Interface

Represents an interface for similarity measures between two programs, e.g., based on their tree structure.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface ISimilarityMeasure<in TProgram>
where TProgram : IProgram
```

[View Source](#)

### Type Parameters

*TProgram*

The type of program.

The ISimilarityMeasure(TProgram) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two <i>TProgram</i> .

### See Also

[Genetica.Similarity Namespace](#)

## [ISimilarityMeasure\(TProgram\).ISimilarityMeasure\(TProgram\)](#)

### Methods

The [ISimilarityMeasure\(TProgram\)](#) generic type exposes the following members.

#### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.

### See Also

[ISimilarityMeasure\(TProgram\)Interface](#)

[Genetica.Similarity Namespace](#)

## [ISimilarityMeasure\(TProgram\).Calculate Method](#)

Calculates the similarity (or inverse distance) between two TProgram.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: [TProgram](#)

The first program of the comparison.

*prog2*

Type: [TProgram](#)

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between 0 and 1 representing the calculated similarity between the given programs. A value near 1 indicates a high similarity (or low distance) between the two programs, while a value near 0 represents a low similarity (or high distance) between the programs. If the programs are the same, it returns 1, if any of the programs is [null](#), it returns 0.

### See Also

[ISimilarityMeasure\(TProgram\)Interface](#)

[Genetica.Similarity Namespace](#)

## LeafSimilarity(TProgram, TOutput) Class

Represents a [ISimilarityMeasure\(TProgram\)](#) for [ITreeProgram\(TOutput\)](#). It measures the similarity of programs based on the leaf nodes of their expressions. Leaf similarity is given by the division between the number of common terminals and the number of all terminals of the two given programs.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.LeafSimilarity(TProgram, TOutput)

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class LeafSimilarity<TProgram, TOutput> : ISimilarityMeasure<TProgram>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

**TProgram**

The type of program.

**TOutput**

The type of program output.

The LeafSimilarity(TProgram, TOutput) type exposes the following members.

#### Constructors

	Name	Description
	<a href="#">LeafSimilarity(TProgram, TOutput)</a>	Initializes a new instance of the LeafSimilarity(TProgram, TOutput) class

#### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)

	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Similarity Namespace](#)

## [LeafSimilarity\(TProgram, TOutput\)](#) Constructor

Initializes a new instance of the [LeafSimilarity\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public LeafSimilarity()
```

[View Source](#)

### See Also

[LeafSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## [LeafSimilarity\(TProgram, TOutput\).LeafSimilarity\(TProgram, TOutput\)](#) Methods

The [LeafSimilarity\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
 <a href="#">Calculate</a>		Calculates the similarity (or inverse distance) between two <code>TProgram</code> .
 <a href="#">Equals</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>		(Inherited from <a href="#">Object</a> .)

### See Also

[LeafSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## [LeafSimilarity\(TProgram, TOutput\).Calculate Method](#)

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: *TProgram*

The first program of the comparison.

*prog2*

Type: *TProgram*

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between [0](#) and [1](#) representing the calculated similarity between the given programs. A value near [1](#) indicates a high similarity (or low distance) between the two programs, while a value near [0](#) represents a low similarity (or high distance) between the programs. If the programs are the same, it returns [1](#), if any of the programs is [null](#), it returns [0](#).

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[LeafSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## NormalNotationEditSimilarity Class

Measures the similarity of two [IProgram](#) based on the edit or Levenshtein string distance between the expressions of the programs in normal notation, i.e., based on the number of edits needed to transform the expression of one program into the one of the other program.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.NormalNotationEditSimilarity

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class NormalNotationEditSimilarity : ISimilarityMeasure<IProgram>
```

[View Source](#)

The **NormalNotationEditSimilarity** type exposes the following members.

### Constructors

	Name	Description
≡	<a href="#">NormalNotationEditSimilarity</a>	Initializes a new instance of the <b>NormalNotationEditSimilarity</b> class

### Methods

	Name	Description
≡	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two <b>TProgram</b> .
≡	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
≡	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[Genetica.Similarity Namespace](#)

## NormalNotationEditSimilarity Constructor

Initializes a new instance of the [NormalNotationEditSimilarity](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public NormalNotationEditSimilarity()
```

[View Source](#)

### See Also

[NormalNotationEditSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## NormalNotationEditSimilarity.NormalNotationEditSimilarity Methods

The [NormalNotationEditSimilarity](#) type exposes the following members.

### Methods

	Name	Description
       	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
      	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
     	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
    	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
   	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[NormalNotationEditSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## NormalNotationEditSimilarity.Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Calculate(  
    IProgram prog1,  
    IProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: [Genetica.Elements.IProgram](#)

The first program of the comparison.

*prog2*

Type: [Genetica.Elements.IProgram](#)

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between 0 and 1 representing the calculated similarity between the given programs. A value near 1 indicates a high similarity (or low distance) between the two programs, while a value near 0 represents a low similarity (or high distance) between the programs. If the programs are the same, it returns 1, if any of the programs is *null*, it returns 0.

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[NormalNotationEditSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## PrefixNotationEditSimilarity Class

Measures the similarity of two [MathProgram](#) based on the edit or Levenshtein string distance between the expressions of the programs in prefix notation, i.e., based on the number of edits needed to transform the expression of one program into the one of the other program.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.PrefixNotationEditSimilarity

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class PrefixNotationEditSimilarity : ISimilarityMeasure<MathProgram>
```

[View Source](#)

The **PrefixNotationEditSimilarity** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">PrefixNotationEditSimilarity</a>	Initializes a new instance of the <b>PrefixNotationEditSimilarity</b> class

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two <b>TProgram</b> .
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[Genetica.Similarity Namespace](#)

## PrefixNotationEditSimilarity Constructor

Initializes a new instance of the [PrefixNotationEditSimilarity](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public PrefixNotationEditSimilarity()
```

[View Source](#)

### See Also

[PrefixNotationEditSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## PrefixNotationEditSimilarity.PrefixNotationEditSimilarity Methods

The [PrefixNotationEditSimilarity](#) type exposes the following members.

### Methods

	Name	Description
      	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
      	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
      	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[PrefixNotationEditSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## PrefixNotationEditSimilarity.Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Calculate(  
    MathProgram prog1,  
    MathProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: [Genetica.Elements.MathProgram](#)

The first program of the comparison.

*prog2*

Type: [Genetica.Elements.MathProgram](#)

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between 0 and 1 representing the calculated similarity between the given programs. A value near 1 indicates a high similarity (or low distance) between the two programs, while a value near 0 represents a low similarity (or high distance) between the programs. If the programs are the same, it returns 1, if any of the programs is *null*, it returns 0.

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[PrefixNotationEditSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## PrimitiveSimilarity(TProgram, TOutput) Class

Measures the similarity of [ITreeProgram\(TOutput\)](#) based on the primitives in their expressions.

Primitive similarity is given by the division between the number of common primitives and the number of all primitives of the two given programs.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.PrimitiveSimilarity(TProgram, TOutput)

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class PrimitiveSimilarity<TProgram, TOutput> : ISimilarityMeasure<TProgram>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The PrimitiveSimilarity(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">PrimitiveSimilarity(TProgram, TOutput)</a>	Initializes a new instance of the PrimitiveSimilarity(TProgram, TOutput) class

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)

	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Similarity Namespace](#)

## PrimitiveSimilarity(TProgram, TOutput) Constructor

Initializes a new instance of the [PrimitiveSimilarity\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public PrimitiveSimilarity()
```

[View Source](#)

### See Also

[PrimitiveSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## PrimitiveSimilarity(TProgram, TOutput).PrimitiveSimilarity(TProgram, TOutput) Methods

The [PrimitiveSimilarity\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[PrimitiveSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## PrimitiveSimilarity(TProgram, TOutput).Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: *TProgram*

The first program of the comparison.

*prog2*

Type: *TProgram*

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between [0](#) and [1](#) representing the calculated similarity between the given programs. A value near [1](#) indicates a high similarity (or low distance) between the two programs, while a value near [0](#) represents a low similarity (or high distance) between the programs. If the programs are the same, it returns [1](#), if any of the programs is [null](#), it returns [0](#).

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[PrimitiveSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## SubCombinationSimilarity(TProgram, TOutput) Class

Measures the similarity of [ITreeProgram\(TOutput\)](#) based on their sub-combinations. Sub-combination similarity is given by the division of the number of common sub-combinations in the programs and the total number of sub-combinations.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.SubCombinationSimilarity(TProgram, TOutput)

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SubCombinationSimilarity<TProgram, TOutput> : ISimilarityMeasure<TProgram>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The SubCombinationSimilarity(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SubCombinationSimilarity(TProgram, TOutput)</a>	Initializes a new instance of the SubCombinationSimilarity(TProgram, TOutput) class

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)

	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Similarity Namespace](#)

## **SubCombinationSimilarity(TProgram, TOutput) Constructor**

Initializes a new instance of the [SubCombinationSimilarity\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SubCombinationSimilarity()
```

[View Source](#)

### See Also

[SubCombinationSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## **SubCombinationSimilarity(TProgram, TOutput).SubCombinationSimilarity(TProgram, TOutput) Methods**

The [SubCombinationSimilarity\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	<b>Name</b>	<b>Description</b>
 <a href="#">Calculate</a>		Calculates the similarity (or inverse distance) between two <code>TProgram</code> .
 <a href="#">Equals</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetHashCode</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetType</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>		(Inherited from <a href="#">Object</a> .)

### See Also

[SubCombinationSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## SubCombinationSimilarity(TProgram, TOutput).Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: *TProgram*

The first program of the comparison.

*prog2*

Type: *TProgram*

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between [0](#) and [1](#) representing the calculated similarity between the given programs. A value near [1](#) indicates a high similarity (or low distance) between the two programs, while a value near [0](#) represents a low similarity (or high distance) between the programs. If the programs are the same, it returns [1](#), if any of the programs is [null](#), it returns [0](#).

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[SubCombinationSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## SubProgramSimilarity(TProgram, TOutput) Class

Measures the similarity of [ITreeProgram\(TOutput\)](#) based on their sub-combinations. Sub-program similarity is given by the division of the number of common sub-programs of the programs and the total number of sub-programs.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.SubProgramSimilarity(TProgram, TOutput)

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SubProgramSimilarity<TProgram, TOutput> : ISimilarityMeasure<TProgram>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The SubProgramSimilarity(TProgram, TOutput) type exposes the following members.

#### Constructors

	Name	Description
	<a href="#">SubProgramSimilarity(TProgram, TOutput)</a>	Initializes a new instance of the SubProgramSimilarity(TProgram, TOutput) class

#### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)

	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Similarity Namespace](#)

## SubProgramSimilarity(TProgram, TOutput) Constructor

Initializes a new instance of the [SubProgramSimilarity\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SubProgramSimilarity()
```

[View Source](#)

### See Also

[SubProgramSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## SubProgramSimilarity(TProgram, TOutput).SubProgramSimilarity(TProgram, TOutput) Methods

The [SubProgramSimilarity\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
  	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
  	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[SubProgramSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## SubProgramSimilarity(TProgram, TOutput).Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: *TProgram*

The first program of the comparison.

*prog2*

Type: *TProgram*

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between [0](#) and [1](#) representing the calculated similarity between the given programs. A value near [1](#) indicates a high similarity (or low distance) between the two programs, while a value near [0](#) represents a low similarity (or high distance) between the programs. If the programs are the same, it returns [1](#), if any of the programs is [null](#), it returns [0](#).

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[SubProgramSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## SymbolTreeSimilarity Class

Measures the similarity of two [ITreeProgram](#) based on the similarity of their [SymbolTree\(TProgram\)](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.SymbolTreeSimilarity

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SymbolTreeSimilarity : ISimilarityMeasure<ITreeProgram>
```

[View Source](#)

The **SymbolTreeSimilarity** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SymbolTreeSimilarity</a>	Initializes a new instance of the <b>SymbolTreeSimilarity</b> class

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two <i>TProgram</i> .
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[Genetica.Similarity Namespace](#)

## SymbolTreeSimilarity Constructor

Initializes a new instance of the [SymbolTreeSimilarity](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SymbolTreeSimilarity()
```

[View Source](#)

### See Also

[SymbolTreeSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## SymbolTreeSimilarity.SymbolTreeSimilarity Methods

The [SymbolTreeSimilarity](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[SymbolTreeSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## SymbolTreeSimilarity.Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Calculate(  
    ITreeProgram prog1,  
    ITreeProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: [Genetica.Elements.ITreeProgram](#)

The first program of the comparison.

*prog2*

Type: [Genetica.Elements.ITreeProgram](#)

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between 0 and 1 representing the calculated similarity between the given programs. A value near 1 indicates a high similarity (or low distance) between the two programs, while a value near 0 represents a low similarity (or high distance) between the programs. If the programs are the same, it returns 1, if any of the programs is *null*, it returns 0.

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[SymbolTreeSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## TreeEditSimilarity(TProgram, TOutput) Class

Measures the similarity of two [ITreeProgram\(TOutput\)](#) based on the edit or Levenshtein distance, i.e., based on the number of edits--transformations, additions and deletions--needed to transform the syntactic tree of one program into the one of the other program.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.TreeEditSimilarity(TProgram, TOutput)

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class TreeEditSimilarity<TProgram, TOutput> : ISimilarityMeasure<TProgram>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of program output.

The TreeEditSimilarity(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">TreeEditSimilarity(TProgram, TOutput)</a>	Initializes a new instance of the TreeEditSimilarity(TProgram, TOutput) class

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)

	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## See Also

[Genetica.Similarity Namespace](#)

## TreeEditSimilarity(TProgram, TOutput) Constructor

Initializes a new instance of the [TreeEditSimilarity\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public TreeEditSimilarity()
```

[View Source](#)

### See Also

[TreeEditSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## TreeEditSimilarity(TProgram, TOutput).TreeEditSimilarity(TProgram, TOutput) Methods

The [TreeEditSimilarity\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[TreeEditSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## TreeEditSimilarity(TProgram, TOutput).Calculate Method

Calculates the similarity (or inverse distance) between two *TProgram*.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public double Calculate(  
    TProgram prog1,  
    TProgram prog2  
)
```

[View Source](#)

### Parameters

*prog1*

Type: *TProgram*

The first program of the comparison.

*prog2*

Type: *TProgram*

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between [0](#) and [1](#) representing the calculated similarity between the given programs. A value near [1](#) indicates a high similarity (or low distance) between the two programs, while a value near [0](#) represents a low similarity (or high distance) between the programs. If the programs are the same, it returns [1](#), if any of the programs is [null](#), it returns [0](#).

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[TreeEditSimilarity\(TProgram, TOutput\)Class](#)

[Genetica.Similarity Namespace](#)

## ValueSimilarity Class

Measures the similarity of two [MathProgram](#) in terms of the values that they can compute when we replace their [Constant](#) programs by random variables.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Similarity.ValueSimilarity

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class ValueSimilarity : ISimilarityMeasure<MathProgram>
```

[View Source](#)

The **ValueSimilarity** type exposes the following members.

### Constructors

	Name	Description
	<a href="#">ValueSimilarity</a>	Initializes a new instance of the <b>ValueSimilarity</b> class

### Methods

	Name	Description
	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two <a href="#">TProgram</a> .
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[Genetica.Similarity Namespace](#)

## ValueSimilarity Constructor

Initializes a new instance of the [ValueSimilarity](#) class

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ValueSimilarity()
```

[View Source](#)

### See Also

[ValueSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## ValueSimilarity.ValueSimilarity Methods

The [ValueSimilarity](#) type exposes the following members.

### Methods

	Name	Description
  	<a href="#">Calculate</a>	Calculates the similarity (or inverse distance) between two TProgram.
  	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
  	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

### See Also

[ValueSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## ValueSimilarity.Calculate Method

Calculates the similarity (or inverse distance) between two TProgram.

**Namespace:** [Genetica.Similarity](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double Calculate(  
    MathProgram prog1,  
    MathProgram prog2  
)
```

[View Source](#)

### Parameters

prog1

Type: [Genetica.Elements.MathProgram](#)

The first program of the comparison.

prog2

Type: [Genetica.Elements.MathProgram](#)

The second program of the comparison.

### Return Value

Type: [Double](#)

A number between 0 and 1 representing the calculated similarity between the given programs. A value near 1 indicates a high similarity (or low distance) between the two programs, while a value near 0 represents a low similarity (or high distance) between the programs. If the programs are the same, it returns 1, if any of the programs is null, it returns 0.

### Implements

[ISimilarityMeasure\(TProgram\).Calculate\(TProgram, TProgram\)](#)

### See Also

[ValueSimilarity Class](#)

[Genetica.Similarity Namespace](#)

## Genetica.Trees Namespace

### Classes

Class	Description
 <a href="#">InformationTree(TProgram)</a>	Implementation of the information tree (iTTree) data structure in [1].
 <a href="#">InfoTreeExtensions</a>	Contains a set of extension methods for all kinds of <a href="#">IInformationTree(TProgram)</a> .
 <a href="#">OrderedSymbolTree(TProgram)</a>	Modified version of the symbol-tree data structure in [1] where parent nodes are created for each sub-program corresponding to the position in which they appear in their parent's children list. In other words, this tree allows the separation of function program's children according to the argument position in which they appear.
 <a href="#">OrderedSymbolTree(TProgram).ArgumentNode</a>	
 <a href="#">OrderedSymbolTree(TProgram).SymbolNode</a>	
 <a href="#">SubProgramTree(TProgram, TOutput)</a>	Modified version of the <a href="#">OrderedSymbolTree(TProgram)</a> data structure where nodes are created for each sub-program of an added <a href="#">ITreeProgram(TOutput)</a> .
 <a href="#">SymbolTree(TProgram)</a>	Implementation of the symbol-tree data structure in [1].

### Interfaces

Interface	Description
 <a href="#">IArgInfoTreeNode</a>	An interface for <a href="#">IInformationTree(TProgram)</a> that represent arguments.
 <a href="#">IInformationTree(TProgram)</a>	Represents an interface for trees representing information about a collection of <a href="#">ITreePrograms</a> . Usually an <a href="#">IInformationTree(TProgram)</a> represents the "average" semantic and/or syntactical structure of the set of <a href="#">ITreeNode</a> considered.
 <a href="#">IInformationTreeNode</a>	Represents a specialization of <a href="#">ITreeNode</a> for information trees.

## IArgInfoTreeNode Interface

An interface for [IInformationTree\(TProgram\)](#) that represent arguments.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IArgInfoTreeNode : IInformationTreeNode,
    ITreeNode
```

The **IArgInfoTreeNode** type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in. (Inherited from <a href="#">IInformationTreeNode</a> .)
	<a href="#">Value</a>	Gets or sets the value associated with the node. (Inherited from <a href="#">IInformationTreeNode</a> .)

### Extension Methods

	Name	Description
	<a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[Genetica.Trees Namespace](#)



## IArgInfoTreeNode.IArgInfoTreeNode Properties

The [IArgInfoTreeNode](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in. (Inherited from <a href="#">IInformationTreeNode</a> .)
	<a href="#">Value</a>	Gets or sets the value associated with the node. (Inherited from <a href="#">IInformationTreeNode</a> .)

### See Also

[IArgInfoTreeNode Interface](#)

[Genetica.Trees Namespace](#)

## IArgInfoTreeNode.IArgInfoTreeNode Methods

The [IArgInfoTreeNode](#) type exposes the following members.

### Extension Methods

Name	Description
 <a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[IArgInfoTreeNode Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram) Interface

Represents an interface for trees representing information about a collection of [ITreePrograms](#).

Usually an IInformationTree(TProgram) represents the "average" semantic and/or syntactical structure of the set of [ITreeNode](#) considered.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IInformationTree<in TProgram>
where TProgram : ITreeProgram
```

[View Source](#)

#### Type Parameters

TProgram

The type of program.

The IInformationTree(TProgram) type exposes the following members.

### Properties

	Name	Description
	<a href="#">RootNode</a>	Gets the root node of this information tree.

### Methods

	Name	Description
	<a href="#">AddProgram</a>	Adds a program to this information tree.
	<a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.
	<a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.
	<a href="#">GetCount</a>	Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.
	<a href="#">GetNodeCount</a>	Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.
	<a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.

## Extension Methods

	<b>Name</b>	<b>Description</b>
 <a href="#">GetFullness(TProgram)</a>		Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile(TProgram)</a>		Saves the given IInformationTree(TProgram) to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(TProgram)</a>		Saves the given IInformationTree(TProgram) to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).IInformationTree(TProgram) Properties

The [IInformationTree\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">RootNode</a>	Gets the root node of this information tree.

### See Also

[IInformationTree\(TProgram\)Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).RootNode Property

Gets the root node of this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
IInformationTreeNode RootNode { get; }
```

[View Source](#)

**Property Value**

Type: [IInformationTreeNode](#)

### See Also

[IInformationTree\(TProgram\)Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).IInformationTree(TProgram) Methods

The [IInformationTree\(TProgram\)](#) generic type exposes the following members.

### Methods

Name	Description
 <a href="#">AddProgram</a>	Adds a program to this information tree.
 <a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.
 <a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.
 <a href="#">GetCount</a>	Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.
 <a href="#">GetNodeCount</a>	Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.
 <a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.

### Extension Methods

Name	Description
 <a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[IInformationTree\(TProgram\)Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).AddProgram Method

Adds an program to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
void AddProgram(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be added to the tree.

#### See Also

[IInformationTree\(TProgram\)Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).AddPrograms Method

Adds a collection of programs to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
void AddPrograms(  
    IEnumerable<TProgram> programs  
)
```

[View Source](#)

### Parameters

*programs*

Type: [System.Collections.Generic.IEnumerable\(TProgram\)](#)

The programs to be added to the tree.

### See Also

[IInformationTree\(TProgram\)Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).Clear Method

Clears the tree by removing all information collected from the added programs.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
void Clear()
```

[View Source](#)

### See Also

[IInformationTree\(TProgram\) Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).GetCount Method

Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
uint GetCount()
```

[View Source](#)

### Return Value

Type: [UInt32](#)

The structure-unique node count.

### See Also

[IInformationTree\(TProgram\)Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).GetNodeCount Method

Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
uint GetNodeCount()
```

[View Source](#)

### Return Value

Type: [UInt32](#)

The node count.

### See Also

[IInformationTree\(TProgram\) Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTree(TProgram).Prune Method

Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
void Prune(  
    double valueThreshold  
)
```

[View Source](#)

#### Parameters

*valueThreshold*

Type: [System.Double](#)

The threshold value used to prune the nodes.

#### See Also

[IInformationTree\(TProgram\)Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTreeNode Interface

Represents a specialization of [ITreeNode](#) for information trees.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public interface IInformationTreeNode : ITreeNode
```

[View Source](#)

The **IInformationTreeNode** type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in.
	<a href="#">Value</a>	Gets or sets the value associated with the node.

### Extension Methods

	Name	Description
	<a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[Genetica.Trees Namespace](#)



## IInformationTreeNode.IInformationTreeNode Properties

The [IInformationTreeNode](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	Gets the list of child nodes associated with this node. (Inherited from <a href="#">ITreeNode</a> .)
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in.
	<a href="#">Value</a>	Gets or sets the value associated with the node.

### See Also

[IInformationTreeNode Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTreeNode.RootNode Property

Gets the root node of tree in which this node is in.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
IInformationTreeNode RootNode { get; }
```

[View Source](#)

*Property Value*

Type: [IInformationTreeNode](#)

### See Also

[IInformationTreeNode Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTreeNode.Value Property

Gets or sets the value associated with the node.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
uint Value { get; set; }
```

[View Source](#)

*Property Value*

Type: [UInt32](#)

### See Also

[IInformationTreeNode Interface](#)

[Genetica.Trees Namespace](#)

## IInformationTreeNode.IInformationTreeNode Methods

The [IInformationTreeNode](#) type exposes the following members.

### Extension Methods

Name	Description
 <a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[IInformationTreeNode Interface](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram) Class

Implementation of the information tree (iTree) data structure in [1].

### Inheritance Hierarchy

[System.Object](#)

Genetica.Trees.InformationTree(TProgram)

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class InformationTree<TProgram> : IInformationTree<TProgram>
where TProgram : ITreeProgram
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

The InformationTree(TProgram) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">InformationTree(TProgram)</a>	Initializes a new instance of the InformationTree(TProgram) class

### Properties

	Name	Description
	<a href="#">RootNode</a>	Gets the root node of this information tree.

### Methods

	Name	Description
	<a href="#">AddProgram</a>	Adds an program to this information tree.
	<a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.
	<a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">GetCount</a>	Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetNodeCount</a>	Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Extension Methods

Name	Description
 <a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## Remarks

[1] Ekárt, A. and Gustafson, S. (2004, April). A data structure for improved GP analysis via efficient computation and visualization of population measures. In European Conference on Genetic Programming (pp. 35-46). Springer Berlin Heidelberg.

## See Also

[Genetica.Trees Namespace](#)

## InformationTree(TProgram) Constructor

Initializes a new instance of the [InformationTree\(TProgram\)](#) class

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public InformationTree()
```

[View Source](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).InformationTree(TProgram) Properties

The [InformationTree\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">RootNode</a>	Gets the root node of this information tree.

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).RootNode Property

Gets the root node of this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IInformationTreeNode RootNode { get; }
```

[View Source](#)

*Property Value*

Type: [IInformationTreeNode](#)

*Implements*

[IInformationTree\(TProgram\).RootNode](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).InformationTree(TProgram) Methods

The [InformationTree\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
 <a href="#">AddProgram</a>	Adds an program to this information tree.	
 <a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.	
 <a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.	
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">GetCount</a>	Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.	
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">GetNodeCount</a>	Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.	
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.	
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)	

### Extension Methods

	Name	Description
 <a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)	
 <a href="#">ToD3JsonFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)	
 <a href="#">ToGraphvizFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file. (Defined by <a href="#">Extensions</a> .)	

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).AddProgram Method

Adds an program to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void AddProgram(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be added to the tree.

#### Implements

[IInformationTree\(TProgram\).AddProgram\(TProgram\)](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).AddPrograms Method

Adds a collection of programs to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void AddPrograms(  
    IEnumerable<TProgram> programs  
)
```

[View Source](#)

#### Parameters

*programs*

Type: [System.Collections.Generic.IEnumerable\(TProgram\)](#)

The programs to be added to the tree.

#### Implements

[IInformationTree\(TProgram\).AddPrograms\(IEnumerable\(TProgram\)\)](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).Clear Method

Clears the tree by removing all information collected from the added programs.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Clear()
```

[View Source](#)

*Implements*

[IInformationTree\(TProgram\).Clear\(\)](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).GetCount Method

Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint GetCount()
```

[View Source](#)

### Return Value

Type: [UInt32](#)

The structure-unique node count.

### Implements

[IInformationTree\(TProgram\).GetCount\(\)](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).GetNodeCount Method

Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint GetNodeCount()
```

[View Source](#)

### Return Value

Type: [UInt32](#)

The node count.

### Implements

[IInformationTree\(TProgram\).GetNodeCount\(\)](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InformationTree(TProgram).Prune Method

Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Prune(  
    double valueThreshold  
)
```

[View Source](#)

#### Parameters

*valueThreshold*

Type: [System.Double](#)

The threshold value used to prune the nodes.

#### Implements

[IInformationTree\(TProgram\).Prune\(Double\)](#)

### See Also

[InformationTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## InfoTreeExtensions Class

Contains a set of extension methods for all kinds of [IInformationTree\(TProgram\)](#).

### Inheritance Hierarchy

[System.Object](#)

Genetica.Trees.InfoTreeExtensions

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static class InfoTreeExtensions
```

[View Source](#)

The **InfoTreeExtensions** type exposes the following members.

### Methods

	Name	Description
 	<a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree.
 	<a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node.

### See Also

[Genetica.Trees Namespace](#)

## InfoTreeExtensions.InfoTreeExtensions Methods

The [InfoTreeExtensions](#) type exposes the following members.

### Methods

	Name	Description
 <a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree.	
 <a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node.	

### See Also

[InfoTreeExtensions Class](#)

[Genetica.Trees Namespace](#)

## InfoTreeExtensions.GetFullness(TProgram) Method

Gets the degree of fullness of the tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static double GetFullness<TProgram>(  
    this IInformationTree<TProgram> tree  
)  
where TProgram : ITreeProgram
```

[View Source](#)

### Parameters

tree

Type: [Genetica.Trees.IInformationTree\(TProgram\)](#)

[Missing <param name="tree"/> documentation for

"M:Genetica.Trees.InfoTreeExtensions.GetFullness`1(Genetica.Trees.IInformationTree{`o})"]

### Type Parameters

TProgram

[Missing <typeparam name="TProgram"/> documentation for

"M:Genetica.Trees.InfoTreeExtensions.GetFullness`1(Genetica.Trees.IInformationTree{`o})"]

### Return Value

Type: [Double](#)

The degree of fullness of the tree.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IInformationTree\(TProgram\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[InfoTreeExtensions Class](#)

[Genetica.Trees Namespace](#)

## InfoTreeExtensions.GetRelativeValue Method

Calculates the value of a node relative to the associated root node.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static double GetRelativeValue(  
    this IInformationTreeNode node  
)
```

[View Source](#)

#### Parameters

*node*

Type: [Genetica.Trees.IInformationTreeNode](#)

The node whose relative value we want to calculate.

#### Return Value

Type: [Double](#)

The value of a node relative to the associated root node.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IInformationTreeNode](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[InfoTreeExtensions Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram) Class

Modified version of the symbol-tree data structure in [1] where parent nodes are created for each sub-program corresponding to the position in which they appear in their parent's children list. In other words, this tree allows the separation of function program's children according to the argument position in which they appear.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Trees.OrderedSymbolTree(TProgram)

[Genetica.Trees.SubProgramTree\(TProgram, TOutput\)](#)

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class OrderedSymbolTree<TProgram> : IInformationTree<TProgram>
where TProgram : ITreeProgram
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

The OrderedSymbolTree(TProgram) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">OrderedSymbolTree(TProgram)</a>	Initializes a new instance of the OrderedSymbolTree(TProgram) class

### Properties

	Name	Description
	<a href="#">RootNode</a>	

### Methods

	Name	Description
	<a href="#">AddElement</a>	

 <a href="#">AddProgram</a>	Adds a program to this information tree.
 <a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.
 <a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetCount</a>	Gets the number of nodes in the symbol-tree.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetNodeCount</a>	Gets the total number of program nodes (genetic nodes) used to build the tree [1].
 <a href="#">GetNodeRatio</a>	Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol-Tree and the number of generation nodes.
 <a href="#">GetSimilarity</a>	Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub-trees.
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Fields

	Name	Description
 <a href="#">rootNode</a>		

## Extension Methods

	Name	Description
 <a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)	
 <a href="#">ToD3JsonFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)	
 <a href="#">ToGraphvizFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file. (Defined by <a href="#">Extensions</a> .)	

## Remarks

[1] Foster, M. A. (2005). The program structure of genetic programming trees (Master's thesis, School of Computer Science and Information Technology, RMIT University Melbourne Australia).

**See Also**

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram) Constructor

Initializes a new instance of the [OrderedSymbolTree\(TProgram\)](#) class

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public OrderedSymbolTree()
```

[View Source](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).OrderedSymbolTree(TProgram) Properties

The [OrderedSymbolTree\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">RootNode</a>	

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).RootNode Property

[Missing <summary> documentation for "P:Genetica.Trees.OrderedSymbolTree`1.RootNode"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IInformationTreeNode RootNode { get; }
```

[View Source](#)

**Property Value**

Type: [IInformationTreeNode](#)

*Implements*

[IInformationTree\(TProgram\).RootNode](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).OrderedSymbolTree(TProgram)

### Methods

The [OrderedSymbolTree\(TProgram\)](#) generic type exposes the following members.

#### Methods

	Name	Description
	<a href="#">AddElement</a>	
	<a href="#">AddProgram</a>	Adds an program to this information tree.
	<a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.
	<a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetCount</a>	Gets the number of nodes in the symbol-tree.
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetNodeCount</a>	Gets the total number of program nodes (genetic nodes) used to build the tree [1].
	<a href="#">GetNodeRatio</a>	Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol–Tree and the number of generation nodes.
	<a href="#">GetSimilarity</a>	Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub–trees.
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.
	<a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

#### Extension Methods

	Name	Description
	<a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)
	<a href="#">ToD3JsonFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file. (Defined by <a href="#">Extensions</a> .)

**See Also**

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).AddElement Method

[Missing <summary> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.AddElement(`o,Genetica.Trees.OrderedSymbolTree{`o}.ArgumentNode,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode,System.Collections.Generic.HashSet{Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode})"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected static void AddElement(  
    TProgram program,  
    OrderedSymbolTree(TProgram).ArgumentNode node,  
    OrderedSymbolTree(TProgram).SymbolNode rootNode,  
    HashSet<OrderedSymbolTree(TProgram).SymbolNode> visited  
)
```

[View Source](#)

### Parameters

program

Type: TProgram

[Missing <param name="program"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.AddElement(`o,Genetica.Trees.OrderedSymbolTree{`o}.ArgumentNode,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode,System.Collections.Generic.HashSet{Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode})"]

node

Type: [Genetica.Trees.OrderedSymbolTree\(TProgram\).ArgumentNode](#)

[Missing <param name="node"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.AddElement(`o,Genetica.Trees.OrderedSymbolTree{`o}.ArgumentNode,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode,System.Collections.Generic.HashSet{Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode})"]

rootNode

Type: [Genetica.Trees.OrderedSymbolTree\(TProgram\).SymbolNode](#)

[Missing <param name="rootNode"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.AddElement(`o,Genetica.Trees.OrderedSymbolTree{`o}.ArgumentNode,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode,System.Collections.Generic.HashSet{Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode})"]

visited

Type: [System.Collections.Generic.HashSet\(OrderedSymbolTree\(TProgram\).SymbolNode\)](#)

[Missing <param name="visited"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.AddElement(`o,Genetica.Trees.OrderedSymbolTree{`o}.ArgumentNode,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode,System.Collections.Generic.HashSet{Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode})"]

**rgumentNode,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode,System.Collections.Generic.Ha  
shSet{Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode})"]**

[See Also](#)

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).AddProgram Method

Adds an program to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public virtual void AddProgram(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be added to the tree.

#### Implements

[IInformationTree\(TProgram\).AddProgram\(TProgram\)](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).AddPrograms Method

Adds a collection of programs to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void AddPrograms(  
    IEnumerable<TProgram> programs  
)
```

[View Source](#)

#### Parameters

*programs*

Type: [System.Collections.Generic.IEnumerable\(TProgram\)](#)

The programs to be added to the tree.

#### Implements

[IInformationTree\(TProgram\).AddPrograms\(IEnumerable\(TProgram\)\)](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## [OrderedSymbolTree\(TProgram\).Clear Method](#)

Clears the tree by removing all information collected from the added programs.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Clear()
```

[View Source](#)

*Implements*

[IInformationTree\(TProgram\).Clear\(\)](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).GetCount Method

Gets the number of nodes in the symbol-tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint GetCount()
```

[View Source](#)

#### Return Value

Type: [UInt32](#)

The tree node count.

#### Implements

[IInformationTree\(TProgram\).GetCount\(\)](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).GetNodeCount Method

Gets the total number of program nodes (genetic nodes) used to build the tree [1].

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint GetNodeCount()
```

[View Source](#)

### Return Value

Type: [UInt32](#)

The total node count.

### Implements

[IInformationTree\(TProgram\).GetNodeCount\(\)](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).GetNodeRatio Method

Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol–Tree and the number of generation nodes.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double GetNodeRatio()
```

[View Source](#)

### Return Value

Type: [Double](#)

The node ratio.

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).GetSimilarity Method

Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub-trees.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double GetSimilarity(  
    OrderedSymbolTree<TProgram> other  
)
```

[View Source](#)

### Parameters

other

Type: [Genetica.Trees.OrderedSymbolTree\(TProgram\)](#)

The other tree we want to calculate the similarity with.

### Return Value

Type: [Double](#)

The similarity between this and the given symbol-tree.

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).Prune Method

Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Prune(  
    double frequencyThreshold  
)
```

[View Source](#)

#### Parameters

*frequencyThreshold*

Type: [System.Double](#)

[Missing <param name="frequencyThreshold"/> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.Prune(System.Double)"]

#### Implements

[IInformationTree\(TProgram\).Prune\(Double\)](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).OrderedSymbolTree(TProgram) Fields

The [OrderedSymbolTree\(TProgram\)](#) generic type exposes the following members.

### Fields

	Name	Description
	<a href="#">rootNode</a>	

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).rootNode Field

[Missing <summary> documentation for "F:Genetica.Trees.OrderedSymbolTree`1.rootNode"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected OrderedSymbolTree(TProgram).SymbolNode rootNode
```

[View Source](#)

### Field Value

Type: [OrderedSymbolTree\(TProgram\).SymbolNode](#)

### See Also

[OrderedSymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).ArgumentNode Class

[Missing <summary> documentation for "T:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode"]

### Inheritance Hierarchy

[System.Object](#)

Genetica.Trees.OrderedSymbolTree(TProgram).ArgumentNode

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected class ArgumentNode : IArgInfoTreeNode,  
    IInformationTreeNode, ITreeNode
```

[View Source](#)

The OrderedSymbolTree(TProgram).ArgumentNode generic type exposes the following members.

### Constructors

	Name	Description
	<a href="#">OrderedSymbolTree(TProgram).ArgumentNode</a>	Initializes a new instance of the OrderedSymbolTree(TProgram).ArgumentNode class

### Properties

	Name	Description
	<a href="#">Children</a>	
	<a href="#">Parent</a>	
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in.
	<a href="#">Value</a>	Gets or sets the value associated with the node.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetCount</a>	
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

## Extension Methods

Name	Description
 <a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).ArgumentNode Constructor

Initializes a new instance of the [OrderedSymbolTree\(TProgram\).ArgumentNode](#) class

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ArgumentNode(  
    int index,  
    OrderedSymbolTree(TProgram).SymbolNode parent  
)
```

[View Source](#)

### Parameters

*index*

Type: [System.Int32](#)

[Missing <param name="index"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.#ctor(System.Int32,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode)" ]

*parent*

Type: [Genetica.Trees.OrderedSymbolTree\(TProgram\).SymbolNode](#)

[Missing <param name="parent"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.#ctor(System.Int32,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode)" ]

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## ArgumentNode.ArgumentNode Properties

The [OrderedSymbolTree\(TProgram\).ArgumentNode](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	
	<a href="#">Parent</a>	
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in.
	<a href="#">Value</a>	Gets or sets the value associated with the node.

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).ArgumentNode.Children Property

[Missing <summary> documentation for  
"P:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.Children"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IDictionary<string, OrderedSymbolTree(TProgram).SymbolNode> Children { get; }
```

[View Source](#)

### Property Value

Type: [IDictionary\(String, OrderedSymbolTree\(TProgram\).SymbolNode\)](#)

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).ArgumentNode.Parent Property

[Missing <summary> documentation for  
"P:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.Parent"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public OrderedSymbolTree(TProgram).SymbolNode Parent { get; }
```

[View Source](#)

### Property Value

Type: [OrderedSymbolTree\(TProgram\).SymbolNode](#)

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## [OrderedSymbolTree\(TProgram\).ArgumentNode.RootNode Property](#)

Gets the root node of tree in which this node is in.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IInformationTreeNode RootNode { get; }
```

[View Source](#)

**Property Value**

Type: [IInformationTreeNode](#)

*Implements*

[IInformationTreeNode.RootNode](#)

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).ArgumentNode.Value Property

Gets or sets the value associated with the node.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint Value { get; set; }
```

[View Source](#)

*Property Value*

Type: [UInt32](#)

*Implements*

[IInformationTreeNode.Value](#)

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## ArgumentNode.ArgumentNode Methods

The [OrderedSymbolTree\(TProgram\).ArgumentNode](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetCount</a>	
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

### Extension Methods

	Name	Description
	<a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).ArgumentNode.GetCount Method

[Missing <summary> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.GetCount"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ushort GetCount()
```

[View Source](#)

**Return Value**

Type: [UInt16](#)

[Missing <returns> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.GetCount"]

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).ArgumentNode.ToString Method

[Missing <summary> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.ToString"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string ToString()
```

[View Source](#)

### Return Value

Type: [String](#)

[Missing <returns> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.ArgumentNode.ToString"]

### See Also

[OrderedSymbolTree\(TProgram\).ArgumentNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).SymbolNode Class

[Missing <summary> documentation for "T:Genetica.Trees.OrderedSymbolTree`1.SymbolNode"]

### Inheritance Hierarchy

[System.Object](#)

Genetica.Trees.OrderedSymbolTree(TProgram).SymbolNode

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
protected class SymbolNode : IInformationTreeNode,  
    ITreeNode
```

[View Source](#)

The OrderedSymbolTree(TProgram).SymbolNode generic type exposes the following members.

### Constructors

	Name	Description
	<a href="#">OrderedSymbolTree(TProgram).SymbolNode</a>	Initializes a new instance of the OrderedSymbolTree(TProgram).SymbolNode class

### Properties

	Name	Description
	<a href="#">Children</a>	
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in.
	<a href="#">Value</a>	Gets or sets the value associated with the node.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetCount</a>	
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)
--	---

## Extension Methods

Name	Description
 <a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Vertex, Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## See Also

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).SymbolNode Constructor

Initializes a new instance of the [OrderedSymbolTree\(TProgram\).SymbolNode](#) class

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SymbolNode(  
    string symbol,  
    int count,  
    OrderedSymbolTree(TProgram).SymbolNode rootNode  
)
```

[View Source](#)

### Parameters

symbol

Type: [System.String](#)

[Missing <param name="symbol"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.#ctor(System.String,System.Int32,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode)"

count

Type: [System.Int32](#)

[Missing <param name="count"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.#ctor(System.String,System.Int32,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode)"

rootNode

Type: [Genetica.Trees.OrderedSymbolTree\(TProgram\).SymbolNode](#)

[Missing <param name="rootNode"/> documentation for

"M:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.#ctor(System.String,System.Int32,Genetica.Trees.OrderedSymbolTree{`o}.SymbolNode)"

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## SymbolNode.SymbolNode Properties

The [OrderedSymbolTree\(TProgram\).SymbolNode](#) type exposes the following members.

### Properties

	Name	Description
	<a href="#">Children</a>	
	<a href="#">RootNode</a>	Gets the root node of tree in which this node is in.
	<a href="#">Value</a>	Gets or sets the value associated with the node.

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).SymbolNode.Children Property

[Missing <summary> documentation for  
"P:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.Children"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public OrderedSymbolTree(TProgram).ArgumentNode[] Children { get; }
```

[View Source](#)

### Property Value

Type: [OrderedSymbolTree\(TProgram\).ArgumentNode\[\]](#)

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## [OrderedSymbolTree\(TProgram\).SymbolNode.RootNode Property](#)

Gets the root node of tree in which this node is in.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IInformationTreeNode RootNode { get; }
```

[View Source](#)

### Property Value

Type: [IInformationTreeNode](#)

### Implements

[IInformationTreeNode.RootNode](#)

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## [OrderedSymbolTree\(TProgram\).SymbolNode.Value Property](#)

Gets or sets the value associated with the node.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint Value { get; set; }
```

[View Source](#)

*Property Value*

Type: [UInt32](#)

*Implements*

[IInformationTreeNode.Value](#)

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## SymbolNode.SymbolNode Methods

The [OrderedSymbolTree\(TProgram\).SymbolNode](#) type exposes the following members.

### Methods

	Name	Description
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetCount</a>	
	<a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">ToString</a>	(Overrides <a href="#">Object.ToString()</a> .)

### Extension Methods

	Name	Description
	<a href="#">GetRelativeValue</a>	Calculates the value of a node relative to the associated root node. (Defined by <a href="#">InfoTreeExtensions</a> .)
	<a href="#">ToD3JsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToD3TreeJsonFile</a>	Saves the tree of the given <a href="#">ITreeNode</a> to a d3.js tree file using a small tree layout. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)
	<a href="#">ToGraphvizFile(String, String, FormatVertexEventHandler(Vertex), FormatEdgeAction(Edge), GraphvizImageType, Int32)</a>	Overloaded. Saves the given <a href="#">ITreeNode</a> to an image file. (Defined by <a href="#">Extensions</a> .)

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).SymbolNode.GetCount Method

[Missing <summary> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.GetCount"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public ushort GetCount()
```

[View Source](#)

### Return Value

Type: [UInt16](#)

[Missing <returns> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.GetCount"]

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## OrderedSymbolTree(TProgram).SymbolNode.ToString Method

[Missing <summary> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.ToString"]

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override string ToString()
```

[View Source](#)

### Return Value

Type: [String](#)

[Missing <returns> documentation for  
"M:Genetica.Trees.OrderedSymbolTree`1.SymbolNode.ToString"]

### See Also

[OrderedSymbolTree\(TProgram\).SymbolNode Class](#)

[Genetica.Trees Namespace](#)

## SubProgramTree(TProgram, TOutput) Class

Modified version of the [OrderedSymbolTree\(TProgram\)](#) data structure where nodes are created for each sub-program of an added [ITreeProgram\(TOutput\)](#).

### Inheritance Hierarchy

[System.Object](#)

[Genetica.Trees.OrderedSymbolTree\(TProgram\)](#)

Genetica.Trees.SubProgramTree(TProgram, TOutput)

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public class SubProgramTree<TProgram, TOutput> : OrderedSymbolTree<TProgram>
where TProgram : Object, ITreeProgram<TOutput>
```

[View Source](#)

### Type Parameters

TProgram

The type of program.

TOutput

The type of output.

The SubProgramTree(TProgram, TOutput) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SubProgramTree(TProgram, TOutput)</a>	Initializes a new instance of the SubProgramTree(TProgram, TOutput) class

### Properties

	Name	Description
	<a href="#">RootNode</a>	(Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)

### Methods

	Name	Description
	<a href="#">AddProgram</a>	Adds an program to this information tree. (Overrides <a href="#">OrderedSymbolTree(TProgram).AddProgram(TProgram)</a> .)

 <a href="#">AddPrograms</a>	Adds a collection of programs to this information tree. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetCount</a>	Gets the number of nodes in the symbol-tree. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetNodeCount</a>	Gets the total number of program nodes (genetic nodes) used to build the tree [1]. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetNodeRatio</a>	Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol-Tree and the number of generation nodes. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetSimilarity</a>	Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub-trees. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Fields

	Name	Description
 <a href="#">rootNode</a>	(Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)	

## See Also

[Genetica.Trees Namespace](#)

## [SubProgramTree\(TProgram, TOutput\)](#) Constructor

Initializes a new instance of the [SubProgramTree\(TProgram, TOutput\)](#) class

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SubProgramTree()
```

[View Source](#)

### See Also

[SubProgramTree\(TProgram, TOutput\)Class](#)

[Genetica.Trees Namespace](#)

## SubProgramTree(TProgram, TOutput).SubProgramTree(TProgram, TOutput) Properties

The [SubProgramTree\(TProgram, TOutput\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">RootNode</a>	(Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)

### See Also

[SubProgramTree\(TProgram, TOutput\)Class](#)

[Genetica.Trees Namespace](#)

## SubProgramTree(TProgram, TOutput).SubProgramTree(TProgram, TOutput) Methods

The [SubProgramTree\(TProgram, TOutput\)](#) generic type exposes the following members.

### Methods

	Name	Description
 <a href="#">AddProgram</a>		Adds an program to this information tree. (Overrides <a href="#">OrderedSymbolTree(TProgram).AddProgram(TProgram)</a> .)
 <a href="#">AddPrograms</a>		Adds a collection of programs to this information tree. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">Clear</a>		Clears the tree by removing all information collected from the added programs. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">Equals</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">Finalize</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetCount</a>		Gets the number of nodes in the symbol-tree. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetHashCode</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">GetNodeCount</a>		Gets the total number of program nodes (genetic nodes) used to build the tree [1]. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetNodeRatio</a>		Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol–Tree and the number of generation nodes. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetSimilarity</a>		Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub–trees. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">GetType</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>		(Inherited from <a href="#">Object</a> .)
 <a href="#">Prune</a>		Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold. (Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)
 <a href="#">ToString</a>		(Inherited from <a href="#">Object</a> .)

### See Also

[SubProgramTree\(TProgram, TOutput\)Class](#)

[Genetica.Trees Namespace](#)

## [SubProgramTree\(TProgram, TOutput\).AddProgram](#) Method

Adds an program to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public override void AddProgram(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be added to the tree.

#### Implements

[IInformationTree\(TProgram\).AddProgram\(TProgram\)](#)

### See Also

[SubProgramTree\(TProgram, TOutput\)Class](#)

[Genetica.Trees Namespace](#)

## SubProgramTree(TProgram, TOutput).SubProgramTree(TProgram, TOutput) Fields

The [SubProgramTree\(TProgram, TOutput\)](#) generic type exposes the following members.

### Fields

	Name	Description
	<a href="#">rootNode</a>	(Inherited from <a href="#">OrderedSymbolTree(TProgram)</a> .)

### See Also

[SubProgramTree\(TProgram, TOutput\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram) Class

Implementation of the symbol-tree data structure in [1].

### Inheritance Hierarchy

[System.Object](#)

Genetica.Trees.SymbolTree(TProgram)

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public class SymbolTree<TProgram> : IInformationTree<TProgram>
where TProgram : ITreeProgram
```

[View Source](#)

### Type Parameters

*TProgram*

The type of program.

The SymbolTree(TProgram) type exposes the following members.

### Constructors

	Name	Description
	<a href="#">SymbolTree(TProgram)</a>	Initializes a new instance of the SymbolTree(TProgram) class

### Properties

	Name	Description
	<a href="#">RootNode</a>	Gets the root node of this information tree.

### Methods

	Name	Description
	<a href="#">AddProgram</a>	Adds an program to this information tree.
	<a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.
	<a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.
	<a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)
	<a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)

 <a href="#">GetCount</a>	Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">GetNodeCount</a>	Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.
 <a href="#">GetNodeRatio</a>	Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol–Tree and the number of generation nodes.
 <a href="#">GetSimilarity</a>	Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub–trees.
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)
 <a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)

## Extension Methods

Name	Description
 <a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)
 <a href="#">ToD3JsonFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)
 <a href="#">ToGraphvizFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file. (Defined by <a href="#">Extensions</a> .)

## Remarks

[1] Foster, M. A. (2005). The program structure of genetic programming trees (Master's thesis, School of Computer Science and Information Technology, RMIT University Melbourne Australia).

## See Also

[Genetica.Trees Namespace](#)

## [SymbolTree\(TProgram\) Constructor](#)

Initializes a new instance of the [SymbolTree\(TProgram\)](#) class

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public SymbolTree()
```

[View Source](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram).SymbolTree(TProgram) Properties

The [SymbolTree\(TProgram\)](#) generic type exposes the following members.

### Properties

	Name	Description
	<a href="#">RootNode</a>	Gets the root node of this information tree.

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## [SymbolTree\(TProgram\).RootNode Property](#)

Gets the root node of this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public IInformationTreeNode RootNode { get; }
```

[View Source](#)

*Property Value*

Type: [IInformationTreeNode](#)

*Implements*

[IInformationTree\(TProgram\).RootNode](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram).SymbolTree(TProgram) Methods

The [SymbolTree\(TProgram\)](#) generic type exposes the following members.

### Methods

	Name	Description
 <a href="#">AddProgram</a>	Adds an program to this information tree.	
 <a href="#">AddPrograms</a>	Adds a collection of programs to this information tree.	
 <a href="#">Clear</a>	Clears the tree by removing all information collected from the added programs.	
 <a href="#">Equals</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">Finalize</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">GetCount</a>	Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.	
 <a href="#">GetHashCode</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">GetNodeCount</a>	Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.	
 <a href="#">GetNodeRatio</a>	Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol–Tree and the number of generation nodes.	
 <a href="#">GetSimilarity</a>	Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub–trees.	
 <a href="#">GetType</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">MemberwiseClone</a>	(Inherited from <a href="#">Object</a> .)	
 <a href="#">Prune</a>	Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.	
 <a href="#">ToString</a>	(Inherited from <a href="#">Object</a> .)	

### Extension Methods

	Name	Description
 <a href="#">GetFullness(TProgram)</a>	Gets the degree of fullness of the tree. (Defined by <a href="#">InfoTreeExtensions</a> .)	
 <a href="#">ToD3JsonFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to a d3.js tree file. (Defined by <a href="#">Extensions</a> .)	
 <a href="#">ToGraphvizFile(TProgram)</a>	Saves the given <a href="#">IInformationTree(TProgram)</a> to an image file. (Defined by <a href="#">Extensions</a> .)	

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)



## [SymbolTree\(TProgram\).AddProgram](#) Method

Adds an program to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void AddProgram(  
    TProgram program  
)
```

[View Source](#)

#### Parameters

*program*

Type: *TProgram*

The program to be added to the tree.

#### Implements

[IInformationTree\(TProgram\).AddProgram\(TProgram\)](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## [SymbolTree\(TProgram\).AddPrograms Method](#)

Adds a collection of programs to this information tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void AddPrograms(  
    IEnumerable<TProgram> programs  
)
```

[View Source](#)

#### Parameters

*programs*

Type: [System.Collections.Generic.IEnumerable\(TProgram\)](#)

The programs to be added to the tree.

#### Implements

[IInformationTree\(TProgram\).AddPrograms\(IEnumerable\(TProgram\)\)](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## [SymbolTree\(TProgram\).Clear Method](#)

Clears the tree by removing all information collected from the added programs.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Clear()
```

[View Source](#)

*Implements*

[IInformationTree\(TProgram\).Clear\(\)](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram).GetCount Method

Gets the number of node positions sampled in the tree search space, i.e., the structure-unique node count.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint GetCount()
```

[View Source](#)

### Return Value

Type: [UInt32](#)

The structure-unique node count.

### Implements

[IInformationTree\(TProgram\).GetCount\(\)](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram).GetNodeCount Method

Gets the total number of terminal nodes (genetic nodes) of the programs added to the tree.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public uint GetNodeCount()
```

[View Source](#)

### Return Value

Type: [UInt32](#)

The node count.

### Implements

[IInformationTree\(TProgram\).GetNodeCount\(\)](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram).GetNodeRatio Method

Gets the ratio between the number of tree nodes and the total number of nodes inserted into the tree. The rationale is to discover correlations between the size of the Symbol–Tree and the number of generation nodes.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double GetNodeRatio()
```

[View Source](#)

### Return Value

Type: [Double](#)

The node ratio.

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram).GetSimilarity Method

Gets the similarity between this and another symbol-tree. It counts the number of genetic nodes that have similar sub-trees.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public double GetSimilarity(  
    SymbolTree<TProgram> other  
)
```

[View Source](#)

### Parameters

other

Type: [Genetica.Trees.SymbolTree\(TProgram\)](#)

The other tree we want to calculate the similarity with.

### Return Value

Type: [Double](#)

The similarity between this and the given symbol-tree.

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## SymbolTree(TProgram).Prune Method

Prunes the tree by removing all nodes stored in this tree whose value falls below the given threshold.

**Namespace:** [Genetica.Trees](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public void Prune(  
    double frequencyThreshold  
)
```

[View Source](#)

#### Parameters

*frequencyThreshold*

Type: [System.Double](#)

[Missing <param name="frequencyThreshold"/> documentation for  
"M:Genetica.Trees.SymbolTree`1.Prune(System.Double)"]

#### Implements

[IInformationTree\(TProgram\).Prune\(Double\)](#)

### See Also

[SymbolTree\(TProgram\)Class](#)

[Genetica.Trees Namespace](#)

## Genetica.Util Namespace

### Classes

	<b>Class</b>	<b>Description</b>
	<a href="#">CollectionUtil</a>	Provides several utility methods to create and update collection objects.
	<a href="#">StringUtil</a>	

## CollectionUtil Class

Provides several utility methods to create and update collection objects.

### Inheritance Hierarchy

[System.Object](#)

Genetica.Util.CollectionUtil

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static class CollectionUtil
```

[View Source](#)

The **CollectionUtil** type exposes the following members.

### Methods

	Name	Description
 	<a href="#">AddRange(T)(ISet(T), IEnumerable(T))</a>	Adds the given values to the given set.
 	<a href="#">AddRange(TKey, TValue)(IDictionary(TKey, TValue), IDictionary(TKey, TValue))</a>	Adds the given values to the dictionary.
 	<a href="#">AddRange(TKey, TValue)(IDictionary(TKey, TValue), IEnumerable(KeyValuePair(TKey, TValue)))</a>	Adds the given values to the dictionary.
 	<a href="#">AddSorted(T)</a>	Adds the given item to the sorted list at the correct index.
 	<a href="#">Align(T)</a>	Modifies the given lists by aligning their items so that equal items have the same index.
 	<a href="#">Dispose(T)</a>	Disposes of all items in the given collection.
 	<a href="#">GetAllCombinations(T)</a>	Gets an IList representing all combinations of T items from the provided lists. Combinations are created in the following form: [item from the 1st list, item from the 2nd list , ..., item from the nth list].
 	<a href="#">GetRandomItem(T)(IDictionary(T, Double))</a>	Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the

		dictionary. Uses a static random number generator.
 <a href="#">GetRandomItem(T)(IList(T))</a>		Gets a random item from the list. Uses a static random number generator.
 <a href="#">GetRandomItem(T)(IDictionary(T, Double), Random)</a>		Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the dictionary.
 <a href="#">GetRandomItem(T)(IList(T), Random)</a>		Gets a random item from the list.
 <a href="#">GetSubset(T)</a>		Gets a subset of the given list between the given indexes.
 <a href="#">GetUniqueItems(T)</a>		Gets a list of items which is equivalent to a given list (in terms Equals and GetHashCode) but where the items are replaced by the corresponding objects referenced in the given dictionary.
 <a href="#">Shuffle(T)(IList(T))</a>		Shuffles the given list so that the items are sorted in a random way. Uses a static random number generator.
 <a href="#">Shuffle(T)(IList(T), Random)</a>		Shuffles the given list so that the items are sorted in a random way.

## See Also

[Genetica.Util Namespace](#)

## CollectionUtil.CollectionUtil Methods

The [CollectionUtil](#) type exposes the following members.

### Methods

	Name	Description
 	<a href="#">AddRange(T)(ISet(T), IEnumerable(T))</a>	Adds the given values to the given set.
 	<a href="#">AddRange(TKey, TValue)(IDictionary(TKey, TValue), IDictionary(TKey, TValue))</a>	Adds the given values to the dictionary.
 	<a href="#">AddRange(TKey, TValue)(IDictionary(TKey, TValue), IEnumerable(KeyValuePair(TKey, TValue)))</a>	Adds the given values to the dictionary.
 	<a href="#">AddSorted(T)</a>	Adds the given item to the sorted list at the correct index.
 	<a href="#">Align(T)</a>	Modifies the given lists by aligning their items so that equal items have the same index.
 	<a href="#">Dispose(T)</a>	Disposes of all items in the given collection.
 	<a href="#">GetAllCombinations(T)</a>	Gets an IList representing all combinations of <i>T</i> items from the provided lists. Combinations are created in the following form: [item from the 1st list, item from the 2nd list , ..., item from the nth list].
 	<a href="#">GetRandomItem(T)(IDictionary(T, Double))</a>	Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the dictionary. Uses a static random number generator.
 	<a href="#">GetRandomItem(T)(IList(T))</a>	Gets a random item from the list. Uses a static random number generator.
 	<a href="#">GetRandomItem(T)(IDictionary(T, Double), Random)</a>	Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the dictionary.
 	<a href="#">GetRandomItem(T)(IList(T), Random)</a>	Gets a random item from the list.
 	<a href="#">GetSubset(T)</a>	Gets a subset of the given list between the given indexes.
 	<a href="#">GetUniqueItems(T)</a>	Gets a list of items which is equivalent to a given list (in terms Equals and GetHashCode) but where the items are replaced by the corresponding objects referenced in the given dictionray.

 	<a href="#"><u>Shuffle(T)(IList(T))</u></a>	Shuffles the given list so that the items are sorted in a random way. Uses a static random number generator.
 	<a href="#"><u>Shuffle(T)(IList(T), Random)</u></a>	Shuffles the given list so that the items are sorted in a random way.

## See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.AddRange Method

### Overload List

	Name	Description
 	<a href="#">AddRange(TKey, TValue)(IDictionary(TKey, TValue), IDictionary(TKey, TValue))</a>	Adds the given values to the dictionary.
 	<a href="#">AddRange(TKey, TValue)(Dictionary(TKey, TValue), IEnumerable(KeyValuePair(TKey, TValue)))</a>	Adds the given values to the dictionary.
 	<a href="#">AddRange(T)(ISet(T), IEnumerable(T))</a>	Adds the given values to the given set.

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## [CollectionUtil.AddRange\(TKey, TValue\) Method \(IDictionary\(TKey, TValue\), IDictionary\(TKey, TValue\)\)](#)

Adds the given values to the dictionary.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static void AddRange<TKey, TValue>(
    this IDictionary<TKey, TValue> set,
    IDictionary<TKey, TValue> values
)
```

[View Source](#)

### Parameters

*set*

Type: [System.Collections.Generic.IDictionary\(TKey, TValue\)](#)

The dictionary we want to add the values to.

*values*

Type: [System.Collections.Generic.IDictionary\(TKey, TValue\)](#)

The values to be added to the dictionary.

### Type Parameters

*TKey*

The type of key stored in the dictionary.

*TValue*

The type of value stored in the dictionary.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IDictionary\(TKey, TValue\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[AddRange Overload](#)

[Genetica.Util Namespace](#)

## [CollectionUtil.AddRange\(TKey, TValue\) Method \(IDictionary\(TKey, TValue\), IEnumerable\(KeyValuePair\(TKey, TValue\)\)\)](#)

Adds the given values to the dictionary.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static void AddRange<TKey, TValue>(
    this IDictionary<TKey, TValue> set,
    IEnumerable<KeyValuePair<TKey, TValue>> values
)
```

[View Source](#)

### Parameters

**set**

Type: [System.Collections.Generic.IDictionary\(TKey, TValue\)](#)

The dictionary we want to add the values to.

**values**

Type: [System.Collections.Generic.IEnumerable\(KeyValuePair\(TKey, TValue\)\)](#)

The values to be added to the dictionary.

### Type Parameters

**TKey**

The type of key stored in the dictionary.

**TValue**

The type of value stored in the dictionary.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IDictionary\(TKey, TValue\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[AddRange Overload](#)

[Genetica.Util Namespace](#)

## CollectionUtil.AddRange(*T*) Method (ISet(*T*), IEnumerable(*T*))

Adds the given values to the given set.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static void AddRange<T>(  
    this ISet<T> set,  
    IEnumerable<T> values  
)
```

[View Source](#)

### Parameters

*set*

Type: [System.Collections.Generic.ISet\(\*T\*\)](#)

The set we want to add the values to.

*values*

Type: [System.Collections.Generic.IEnumerable\(\*T\*\)](#)

The values to be added to the set.

### Type Parameters

*T*

The type of items in the collections.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [ISet\(\*T\*\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[AddRange Overload](#)

[Genetica.Util Namespace](#)

## CollectionUtil.AddSorted(*T*) Method

Adds the given item to the sorted list at the correct index.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public static int AddSorted<T>(
    this List<T> list,
    T item
)
where T : Object, IComparable<T>
```

[View Source](#)

### Parameters

*list*

Type: [System.Collections.Generic.List\(\*T\*\)](#)

The sorted list onto which we want to add the given item.

*item*

Type: [\*T\*](#)

The item to add to list.

### Type Parameters

*T*

The type of items in the list

### Return Value

Type: [Int32](#)

The index where the given item was added.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [List\(\*T\*\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### Remarks

This method assumes the given list is already sorted. <https://stackoverflow.com/a/22801345>

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.Align(*T*) Method

Modifies the given lists by aligning their items so that equal items have the same index.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public static void Align<T>(  
    IList<T> list1,  
    IList<T> list2  
)
```

[View Source](#)

### Parameters

*list1*

Type: [System.Collections.Generic.IList\(T\)](#)

The first list.

*list2*

Type: [System.Collections.Generic.IList\(T\)](#)

The second list.

### Type Parameters

*T*

The type of items stored in the given lists.

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.Dispose(T) Method

Disposes of all items in the given collection.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static void Dispose<T>(
    this IEnumerable<T> collection
)
where T : IDisposable
```

[View Source](#)

### Parameters

*collection*

Type: [System.Collections.Generic.IEnumerable\(T\)](#)

The collection whose items we want to dispose.

### Type Parameters

*T*

The type of items in the collection, has to implement [IDisposable](#).

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IEnumerable\(T\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetAllCombinations(*T*) Method

Gets an *IList* representing all combinations of *T* items from the provided lists. Combinations are created in the following form: [item from the 1st list, item from the 2nd list , ..., item from the *n*th list].

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public static IEnumerable<IList<T>> GetAllCombinations<T>(
    this IList<IEnumerable<T>> list
)
```

[View Source](#)

### Parameters

*list*

Type: [System.Collections.Generic.IList\(IEnumerable\(T\)\)](#)

The list containing the lists of items to combine.

### Type Parameters

*T*

The type of items in the lists.

### Return Value

Type: [IEnumerable\(IList\(T\)\)](#)

A list containing all combinations of items.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type *IList(IEnumerable(T))*. When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetRandomItem Method

### Overload List

	Name	Description
 	<a href="#">GetRandomItem(T)(IDictionary(T, Double))</a>	Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the dictionary. Uses a static random number generator.
 	<a href="#">GetRandomItem(T)(IList(T))</a>	Gets a random item from the list. Uses a static random number generator.
 	<a href="#">GetRandomItem(T)(IDictionary(T, Double), Random)</a>	Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the dictionary.
 	<a href="#">GetRandomItem(T)(IList(T), Random)</a>	Gets a random item from the list.

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetRandomItem(*T*) Method (IDictionary(*T*, Double))

Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the dictionary. Uses a static random number generator.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static T GetRandomItem<T>(
    this IDictionary<T, double> dict
)
```

[View Source](#)

### Parameters

*dict*

Type: [System.Collections.Generic.IDictionary\(T, Double\)](#)

The dictionary we want to get a random item from. Assumes *dict.Values.Sum() == 1*.

### Type Parameters

*T*

The type of items stored in the dictionary keys.

### Return Value

Type: *T*

The random item.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IDictionary\(T, Double\)](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[GetRandomItem Overload](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetRandomItem(*T*) Method (IList(*T*))

Gets a random item from the list. Uses a static random number generator.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static T GetRandomItem<T>(
    this IList<T> list
)
```

[View Source](#)

### Parameters

*list*

Type: [System.Collections.Generic.IList\(\*T\*\)](#)

The list we want to get a random item from.

### Type Parameters

*T*

The type of items stored in the list.

### Return Value

Type: ***T***

The random item.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IList\(\*T\*\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[GetRandomItem Overload](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetRandomItem(*T*) Method (*IDictionary*(*T*, Double), Random)

Gets an item from the given dictionary keys, randomly sampled according to the probabilities defined by the corresponding values in the dictionary.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static T GetRandomItem<T>(
    this IDictionary<T, double> dict,
    Random random
)
```

[View Source](#)

### Parameters

*dict*

Type: [System.Collections.Generic.IDictionary](#)(*T*, Double)

The dictionary we want to get a random item from. Assumes *dict.Values.Sum()* == 1.

*random*

Type: [System.Random](#)

The random source used in this operation.

### Type Parameters

*T*

The type of items stored in the dictionary keys.

### Return Value

Type: *T*

The random item.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IDictionary](#)(*T*, Double). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[GetRandomItem Overload](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetRandomItem(*T*) Method (IList(*T*), Random)

Gets a random item from the list.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static T GetRandomItem<T>(
    this IList<T> list,
    Random random
)
```

[View Source](#)

### Parameters

*list*

Type: [System.Collections.Generic.IList\(\*T\*\)](#)

The list we want to get a random item from.

*random*

Type: [System.Random](#)

The random number generator to be used in the item selection.

### Type Parameters

*T*

The type of items stored in the list.

### Return Value

Type: *T*

The random item.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IList\(\*T\*\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[GetRandomItem Overload](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetSubset(*T*) Method

Gets a subset of the given list between the given indexes.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static IList<T> GetSubset<T>(
    this IList<T> collection,
    uint startIndex = 0,
    uint finalIndex = 4294967295
)
```

[View Source](#)

### Parameters

*collection*

Type: [System.Collections.Generic.IList\(T\)](#)

The original collection that we want to get a sub-set.

*startIndex* (Optional)

Type: [System.UInt32](#)

The initial index of the collection sub-set.

*finalIndex* (Optional)

Type: [System.UInt32](#)

The final index of the collection sub-set.

### Type Parameters

*T*

The type of items stored in the given collection.

### Return Value

Type: [IList\(T\)](#)

The subset of the given list.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IList\(T\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.GetUniqueItems(T) Method

Gets a list of items which is equivalent to a given list (in terms Equals and GetHashCode) but where the items are replaced by the corresponding objects referenced in the given dictionary.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

#### C#

```
public static IList<T> GetUniqueItems<T>(
    this IList<T> list,
    IDictionary<T, T> allItems
)
```

[View Source](#)

#### Parameters

*list*

Type: [System.Collections.Generic.IList\(T\)](#)

The list containing the collection of items.

*allItems*

Type: [System.Collections.Generic.IDictionary\(T, T\)](#)

A table where items can be fetched by hashcode.

#### Type Parameters

*T*

The type of items stored in the collection.

#### Return Value

Type: [IList\(T\)](#)

A list of items which is equivalent to a given list.

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IList\(T\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### Remarks

This method is useful when we want to make sure we don't have useless copies of the same objects (as dictated by GetHashCode() and Equals()) stored in different places.

#### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.Shuffle Method

### Overload List

	Name	Description
 	<a href="#">Shuffle(T)(IList(T))</a>	Shuffles the given list so that the items are sorted in a random way. Uses a static random number generator.
 	<a href="#">Shuffle(T)(IList(T), Random)</a>	Shuffles the given list so that the items are sorted in a random way.

### See Also

[CollectionUtil Class](#)

[Genetica.Util Namespace](#)

## CollectionUtil.Shuffle(*T*) Method (*IList(T)*)

Shuffles the given list so that the items are sorted in a random way. Uses a static random number generator.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static void Shuffle<T>(
    this IList<T> list
)
```

[View Source](#)

### Parameters

*list*

Type: [System.Collections.Generic.IList\(T\)](#)

The original list that we want to shuffle.

### Type Parameters

*T*

The type of items stored in the given collection.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IList\(T\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### Remarks

Based on the Fisher-Yates shuffle as stated in the solution:

<https://stackoverflow.com/questions/273313/randomize-a-listt>

### See Also

[CollectionUtil Class](#)

[Shuffle Overload](#)

[Genetica.Util Namespace](#)

## CollectionUtil.Shuffle(*T*) Method (*IList(T)*, *Random*)

Shuffles the given list so that the items are sorted in a random way.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

**C#**

```
public static void Shuffle<T>(
    this IList<T> list,
    Random random
)
```

[View Source](#)

### Parameters

*list*

Type: [System.Collections.Generic.IList\(T\)](#)

The original list that we want to shuffle.

*random*

Type: [System.Random](#)

The random number generator used in the shuffling process.

### Type Parameters

*T*

The type of items stored in the given collection.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [IList\(T\)](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### Remarks

Based on the Fisher-Yates shuffle as stated in the solution:

<https://stackoverflow.com/questions/273313/randomize-a-listt>

### See Also

[CollectionUtil Class](#)

[Shuffle Overload](#)

[Genetica.Util Namespace](#)

## StringUtil Class

[Missing <summary> documentation for "T:Genetica.Util.StringUtil"]

### Inheritance Hierarchy

[System.Object](#)

Genetica.Util.StringUtil

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static class StringUtil
```

[View Source](#)

The **StringUtil** type exposes the following members.

### Methods

	Name	Description
 	<a href="#">LevenshteinDistance</a>	Gets the edit or Levenshtein distance between two strings. The Damereau-Levenshtein Distance algorithm calculates the number of letter additions, subtractions, substitutions, and transpositions (swaps) necessary to convert one string to another. The lower the score, the more similar they are.
 	<a href="#">Repeat(Char, UInt32)</a>	
 	<a href="#">Repeat(String, UInt32)</a>	
 	<a href="#">ToLiteral</a>	

### See Also

[Genetica.Util Namespace](#)

## StringUtil.StringUtil Methods

The [StringUtil](#) type exposes the following members.

### Methods

	Name	Description
 	<a href="#">LevenshteinDistance</a>	Gets the edit or Levenshtein distance between two strings. The Damereau-Levenshtein Distance algorithm calculates the number of letter additions, subtractions, substitutions, and transpositions (swaps) necessary to convert one string to another. The lower the score, the more similar they are.
 	<a href="#">Repeat(Char, UInt32)</a>	
 	<a href="#">Repeat(String, UInt32)</a>	
 	<a href="#">ToLiteral</a>	

### See Also

[StringUtil Class](#)

[Genetica.Util Namespace](#)

## StringUtil.LevenshteinDistance Method

Gets the edit or Levenshtein distance between two strings. The Damereau-Levenshtein Distance algorithm calculates the number of letter additions, subtractions, substitutions, and transpositions (swaps) necessary to convert one string to another. The lower the score, the more similar they are.

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static int LevenshteinDistance(  
    this string a,  
    string b  
)
```

[View Source](#)

### Parameters

a

Type: [System.String](#)

The first string.

b

Type: [System.String](#)

The second string.

### Return Value

Type: [Int32](#)

The edit or Levenshtein distance between the given strings.

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [String](#).

When you use instance method syntax to call this method, omit the first parameter. For more

information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### Remarks

Code based on: <https://stackoverflow.com/questions/9453731/how-to-calculate-distance-similarity-measure-of-given-2-strings>

### See Also

[StringUtil Class](#)

[Genetica.Util Namespace](#)

## StringUtil.Repeat Method

### Overload List

	Name	Description
 <b>S</b>	<a href="#">Repeat(Char, UInt32)</a>	
 <b>S</b>	<a href="#">Repeat(String, UInt32)</a>	

### See Also

[StringUtil Class](#)

[Genetica.Util Namespace](#)

## StringUtil.Repeat Method (Char, UInt32)

[Missing <summary> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.Char,System.UInt32)"]

Namespace: [Genetica.Util](#)

Assembly: Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static string Repeat(  
    this char c,  
    uint num  
)
```

[View Source](#)

### Parameters

c

Type: [System.Char](#)

[Missing <param name="c"/> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.Char,System.UInt32)"]

num

Type: [System.UInt32](#)

[Missing <param name="num"/> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.Char,System.UInt32)"]

### Return Value

Type: [String](#)

[Missing <returns> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.Char,System.UInt32)"]

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [Char](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[StringUtil Class](#)

[Repeat Overload](#)

[Genetica.Util Namespace](#)

## StringUtil.Repeat Method (String, UInt32)

[Missing <summary> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.String,System.UInt32)"]

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static string Repeat(  
    this string str,  
    uint num  
)
```

[View Source](#)

### Parameters

str

Type: [System.String](#)

[Missing <param name="str"/> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.String,System.UInt32)"]

num

Type: [System.UInt32](#)

[Missing <param name="num"/> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.String,System.UInt32)"]

### Return Value

Type: [String](#)

[Missing <returns> documentation for  
"M:Genetica.Util.StringUtil.Repeat(System.String,System.UInt32)"]

### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [String](#). When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

### See Also

[StringUtil Class](#)

[Repeat Overload](#)

[Genetica.Util Namespace](#)

## StringUtil.ToString Method

[Missing <summary> documentation for "M:Genetica.Util.StringUtil.ToString(System.String)"]

**Namespace:** [Genetica.Util](#)

**Assembly:** Genetica (in Genetica.dll) Version: 1.0.6669.36762

### Syntax

C#

```
public static string ToString(  
    this string input  
)
```

[View Source](#)

#### Parameters

*input*

Type: [System.String](#)

[Missing <param name="input"/> documentation for "M:Genetica.Util.StringUtil.ToString(System.String)"]

#### Return Value

Type: [String](#)

[Missing <returns> documentation for "M:Genetica.Util.StringUtil.ToString(System.String)"]

#### Usage Note

In Visual Basic and C#, you can call this method as an instance method on any object of type [String](#).

When you use instance method syntax to call this method, omit the first parameter. For more information, see [Extension Methods \(Visual Basic\)](#) or [Extension Methods \(C# Programming Guide\)](#).

#### See Also

[StringUtil Class](#)

[Genetica.Util Namespace](#)