System.Numerics.Multiprecision Library Reference

August 7, 2016

Contents

Description

Multiprecision is a multiple precision arithmetic library. The library is implemented using GNU Multiple Precision Arithmetic Library (http://gmplib.org/).

Namespaces

Namespace	Description
System.Numerics.Multiprecision	Contains arbitrary precision integer type BigInt, arbitrary precision rational type BigRational and arbitrary precision floating point type BigFloat.

1 Usage

1.0.1 Referencing the System.Numerics.Multiprecision library

Right-click a project node in IDE | Project References... | Add System Extension Library Reference... | enable *System.Numerics.Multiprecision* check box

or add following line to your project's .cmp file:

```
reference <ext/System.Numerics.Multiprecision/System.Numerics.Multiprecision.cml>;
```

2 Installation

The System.Numerics.Multiprecision library uses the GNU MP library that is precompiled for the following platforms:

- 32-bit Windows (x86)
- 64-bit Windows (x64)
- 32-bit PC Linux (i686)
- 64-bit PC Linux (x86_64)

Current version of GNU MP library at the time of writing is 6.1.0.
If you want to install newer version of the GNU MP library or if you have different platform, you can compile the GNU MP library from sources by using the following instructions.

2.0.1 Compiling the GNU MP library from sources in Windows

- Install MinGW-w64 GCC to a path that does not contain spaces. I have mine installed in `C:\mingw-w64`.

  Installer can be obtained from
  http://sourceforge.net/projects/mingw-w64/files/Toolchains%20targetting%20Win32/Personal%20Builds/mingw-builds/installer/mingw-w64-install.exe/download

- Install MSYS2. Installation instructions can be found in
  http://sourceforge.net/p/msys2/wiki/MSYS2%20installation/

  I have mine installed in `C:\msys64`.

- Start MSYS2 shell.

- In the MSYS2 shell obtain **tar** by executing

  ```
  pacman -S tar
  ```

- Obtain **make** by executing

  ```
  pacman -S make
  ```

- Mount your MinGW-directory (mine is in `C:/mingw-w64`) by executing

  ```
  mkdir /mingw
  mount C:/mingw-w64 /mingw
  ```

- Insert your MinGW bin-directory in front of the PATH by executing

  ```
  export PATH=/mingw/mingw64/bin:$PATH
  ```

- Now if you ask which GCC is used by executing

  ```
  which gcc
  ```

  you should get **/mingw/mingw64/bin/gcc**.

- Download the GNU MP library from http://gmplib.org/#DOWNLOAD in bz2 format and place it to your MSYS2 home directory (mine is in `C:\msys64\home\Seppo`).

- Extract the GNU MP library by executing

  ```
  tar xjf gmp-6.1.0.tar.bz2
  ```

- Change to gmp-directory by executing

  ```
  cd gmp-6.1.0
  ```

- Configure the library for 64-bit Windows by executing

  ```
  ./configure --enable-static --disable-shared --host=x86_64-w64-mingw32
  ```

  For 32-bit Windows the command is

  ```
  ./configure --enable-static --disable-shared --host=i686-w64-mingw32
  ```

  Shared libraries are not used in Cmajor so we are disabling them.

- Make the library by executing

  ```
  make
  ```

- Install the library by executing

  ```
  make install
  ```

- Optionally test the library by executing

  ```
  make check
  ```

- Now the header file **gmp.h** should be in `C:\msys64\usr\local\include` directory and the library file **libgmp.a** should be in `C:\msys64\usr\local\lib` directory. Copy gmp.h and libgmp.a under the Cmajor System.Numerics.GmpIntf extension library to directory

  ```
  %APPDATA%\Cmajor\system\ext\System.Numerics.GmpIntf\gmp\windows\x64
  ```

  for 64-bit Windows, or to

  ```
  %APPDATA%\Cmajor\system\ext\System.Numerics.GmpIntf\gmp\windows\x86
  ```

  for 32-bit Windows.

- That's all.

2.0.2 Compiling the GNU MP library from sources in Linux

- Download the GNU MP library from http://gmplib.org/#DOWNLOAD in bz2 format and place it to your home directory.
- Extract the GNU MP library by executing

  ```
  tar xjf gmp-6.1.0.tar.bz2
  ```

- Change to gmp-directory by executing

  ```
  cd gmp-6.1.0
  ```

- Configure the library by executing

  ```
  ./configure --enable-static --disable-shared
  ```

  Shared libraries are not used in Cmajor so we are disabling them.
- Make the library by executing

  ```
  make
  ```

- Install the library by executing

  ```
  make install
  ```

- Optionally test the library by executing

  ```
  make check
  ```

- Now the header file **gmp.h** should be in `/usr/local/include` directory and the library file **libgmp.a** should be in `/usr/local/lib` directory. Copy gmp.h and libgmp.a under the Cmajor System.Numerics.GmpIntf extension library to

  ```
  <your cmajor directory>/system/ext/System.Numerics.GmpIntf/gmp/linux/x86_64
  ```

  directory for 64-bit Linux or to

  ```
  <your cmajor directory>/system/ext/System.Numerics.GmpIntf/gmp/linux/i686
  ```

  directory for 32-bit Linux.
- That's all.

3 System.Numerics.Multiprecision Namespace

Contains arbitrary precision integer type BigInt, arbitrary precision rational type BigRational and arbitrary precision floating point type BigFloat.

3.1 Classes

Class	Description
BigFloat	An arbitrary precision floating point type.
BigInt	An arbitrary precision signed integer type.
BigRational	An arbitrary precision rational nunber type.
Precision	Represents a precision of given number of digits.

3.1.1 BigFloat Class

An arbitrary precision floating point type.

Syntax

```
public class BigFloat;
```

3.1.1.1 Member Functions

Member Function	Description
BigFloat()	Default constructor. Creates an instance of arbitrary precision floating point type and initializes it to zero.
BigFloat(const System.Numerics.Multiprecision.-BigFloat&)	Copy constructor.
operator=(const System.Numerics.-Multiprecision.BigFloat&)	Copy assignment.
BigFloat(System.Numerics.Multiprecision.-BigFloat&&)	Move constructor.
operator=(System.Numerics.Multiprecision.-BigFloat&&)	Move assignment.
BigFloat(const System.Numerics.Multiprecision.-BigInt&)	Constructor. Creates an instance of arbitrary precision floating point type and initializes it from the given arbitrary precision integer type.
BigFloat(const System.Numerics.Multiprecision.-BigRational&)	Constructor. Creates an instance of arbitrary precision floating point type and initializes it from the given arbitrary precision rational type.
BigFloat(const System.Numerics.Multiprecision.-Precision&)	Constructor. Constructs an arbitrary precision floating point value with given precision.
BigFloat(const System.String<char>&)	Constructor. Constructs an arbitrary precision floating point value from given decimal digits.
BigFloat(const System.String<char>&, int)	Constructor. Constructs an arbitrary precision floating point value from given digits that are in given base.
BigFloat(double)	Constructor. Constructs an arbitrary precision floating point value from given double precision value.
BigFloat(int)	Constructor. Constructs an arbitrary precision floating point value from given integer value.

BigFloat(uint)	Constructor. Constructs an arbitrary precision floating point value from given unsigned integer value.
Handle() const	Returns a handle to the GNU MP library arbitrary precision floating point number representation.
ToDouble() const	Converts the arbitrary precision floating point number to a double precision value by truncating it.
ToString() const	Returns the value of the **BigFloat** as a string.
ToString(int) const	Returns the value of the **BigFloat** as a string using given base.
ToString(int, uint) const	Returns the value of the **BigFloat** as a string using given base and given number of digits.
operator=(const System.Numerics.Multiprecision.BigInt&)	Assigns the value of the **BigFloat** from the given arbitrary precision integer.
operator=(const System.Numerics.Multiprecision.BigRational&)	Assigns the value of the **BigFloat** from the given arbitrary precision rational.
operator=(double)	Assigns the value of the **BigFloat** from the given double precision value.
operator=(int)	Assigns the value of the **BigFloat** from the given integer value.
operator=(uint)	Assigns the value of the **BigFloat** from the given unsigned integer value.
~BigFloat()	Frees memory occupied by the **BigFloat** instance.

BigFloat() Member Function

Default constructor. Creates an instance of arbitrary precision floating point type and initializes it to zero.

Syntax

```
public BigFloat();
```

BigFloat(const System.Numerics.Multiprecision.BigFloat&) Member Function

Copy constructor.

Syntax

```
public BigFloat(const System.Numerics.Multiprecision.BigFloat& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigFloat&	A BigFloat to copy from.

operator=(const System.Numerics.Multiprecision.BigFloat&) Member Function

Copy assignment.

Syntax

```
public void operator=(const System.Numerics.Multiprecision.BigFloat& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigFloat&	A BigFloat to assign from.

BigFloat(System.Numerics.Multiprecision.BigFloat&&) Member Function

Move constructor.

Syntax

```
public BigFloat(System.Numerics.Multiprecision.BigFloat&& that);
```

Parameters

Name	Type	Description
that	System.Numerics.Multiprecision.BigFloat&&	A BigFloat to move from.

operator=(System.Numerics.Multiprecision.BigFloat&&) Member Function

Move assignment.

Syntax

```
public void operator=(System.Numerics.Multiprecision.BigFloat&& __parameter0);
```

Parameters

Name	Type	Description
__parameter0	System.Numerics.Multiprecision.BigFloat&&	A BigFloat to assign from.

BigFloat(const System.Numerics.Multiprecision.BigInt&) Member Function

Constructor. Creates an instance of arbitrary precision floating point type and initializes it from the given arbitrary precision integer type.

Syntax

```
public BigFloat(const System.Numerics.Multiprecision.BigInt& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigInt&	An arbitrary precision integer type value to construct from.

BigFloat(const System.Numerics.Multiprecision.BigRational&) Member Function

Constructor. Creates an instance of arbitrary precision floating point type and initializes it from the given arbitrary precision rational type.

Syntax

```
public BigFloat(const System.Numerics.Multiprecision.BigRational& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigRational&	An arbitrary precision rational type value to construct from.

BigFloat(const System.Numerics.Multiprecision.Precision&) Member Function

Constructor. Constructs an arbitrary precision floating point value with given precision.

Syntax

```
public BigFloat(const System.Numerics.Multiprecision.Precision& prec);
```

Parameters

Name	Type	Description
prec	const System.Numerics.Multiprecision.Precision&	Precision.

BigFloat(const System.String<char>&) Member Function

Constructor. Constructs an arbitrary precision floating point value from given decimal digits.

Syntax

```
public BigFloat(const System.String<char>& str);
```

Parameters

Name	Type	Description
str	const System.String<char>&	Digit string.

BigFloat(const System.String<char>&, int) Member Function

Constructor. Constructs an arbitrary precision floating point value from given digits that are in given base.

Syntax

```
public BigFloat(const System.String<char>& str, int base_);
```

Parameters

Name	Type	Description
str	const System.String<char>&	Digit string.
base_	int	Base of digits.

BigFloat(double) Member Function

Constructor. Constructs an arbitrary precision floating point value from given double precision value.

Syntax

```
public BigFloat(double that);
```

Parameters

Name	Type	Description
that	double	A double precision value to initialize from.

BigFloat(int) Member Function

Constructor. Constructs an arbitrary precision floating point value from given integer value.

Syntax

```
public BigFloat(int that);
```

Parameters

Name	Type	Description
that	int	An integer value to initialize from.

BigFloat(uint) Member Function

Constructor. Constructs an arbitrary precision floating point value from given unsigned integer value.

Syntax

```
public BigFloat(uint that);
```

Parameters

Name	Type	Description
that	uint	An unsigned integer value to initialize from.

Handle() const Member Function

Returns a handle to the GNU MP library arbitrary precision floating point number representation.

Syntax

```
public void* Handle() const;
```

Returns

void*

Returns a handle to the GNU MP library arbitrary precision floating point number representation.

ToDouble() const Member Function

Converts the arbitrary precision floating point number to a double precision value by truncating it.

Syntax

```
public double ToDouble() const;
```

Returns

double

Returns the arbitrary precision floating point number converted to a double precision value by truncating it.

ToString() const Member Function

Returns the value of the BigFloat as a string.

Syntax

```
public System.String<char> ToString() const;
```

Returns

System.String<char>

Returns the value of the BigFloat as a string.

ToString(int) const Member Function

Returns the value of the BigFloat as a string using given base.

Syntax

```
public System.String<char> ToString(int base_) const;
```

Parameters

Name	Type	Description
base_	int	Base.

Returns

System.String<char>

Returns the value of the BigFloat as a string using given base.

ToString(int, uint) const Member Function

Returns the value of the BigFloat as a string using given base and given number of digits.

Syntax

```
public System.String<char> ToString(int base_, uint numDigits) const;
```

Parameters

Name	Type	Description
base_	int	Base.
numDigits	uint	Number of digits.

Returns

System.String<char>

Returns the value of the BigFloat as a string using given base and given number of digits.

operator=(const System.Numerics.Multiprecision.BigInt&) Member Function

Assigns the value of the BigFloat from the given arbitrary precision integer.

Syntax

```
public void operator=(const System.Numerics.Multiprecision.BigInt& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigInt&	An arbitrary precision integer to assign from.

operator=(const System.Numerics.Multiprecision.BigRational&) Member Function

Assigns the value of the BigFloat from the given arbitrary precision rational.

Syntax

```
public void operator=(const System.Numerics.Multiprecision.BigRational& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigRational&	An arbitrary precision rational to assign from.

operator=(double) Member Function

Assigns the value of the BigFloat from the given double precision value.

Syntax

```
public void operator=(double that);
```

Parameters

Name	Type	Description
that	double	A double precision value to assign from.

operator=(int) Member Function

Assigns the value of the BigFloat from the given integer value.

Syntax

```
public void operator=(int that);
```

Parameters

Name	Type	Description
that	int	An integer value to assign from.

operator=(uint) Member Function

Assigns the value of the BigFloat from the given unsigned integer value.

Syntax

```
public void operator=(uint that);
```

Parameters

Name	Type	Description
that	uint	An unsigned integer value to assign from.

~BigFloat() Member Function

Frees memory occupied by the BigFloat instance.

Syntax

```
public ~BigFloat();
```

3.1.1.2 Nonmember Functions

Function	Description
Abs(const System.Numerics.Multiprecision.BigFloat&)	Returns the absolute value of the given **BigFloat** value.
Ceil(const System.Numerics.Multiprecision.BigFloat&)	Returns the **BigFloat** rounded up to the next arbitrary precision integer value.
Floor(const System.Numerics.Multiprecision.BigFloat&)	Returns the **BigFloat** rounded down to the previous arbitrary precision integer value.
Sqrt(const System.Numerics.Multiprecision.BigFloat&)	Returns the square root of the given **BigFloat** number.
Trunc(const System.Numerics.Multiprecision.BigFloat&)	Returns the **BigFloat** truncated towards zero.
operator*(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.BigFloat&)	Returns the product of given **BigFloat** value multiplied by another.
operator+(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.BigFloat&)	Returns the sum of given **BigFloat** value added to another.
operator-(const System.Numerics.Multiprecision.BigFloat&)	Returns the negation of **BigFloat** .
operator-(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.BigFloat&)	Returns the difference of given **BigFloat** value subtracted from another.
operator/(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.BigFloat&)	Returns the quotient when given **BigFloat** is divided by another.
operator<(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.BigFloat&)	Returns true if the first **BigFloat** is less than the second **BigFloat** , false otherwise.
operator==(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.BigFloat&)	Returns true if the first **BigFloat** is equal to the second **BigFloat** , false otherwise.

Abs(const System.Numerics.Multiprecision.BigFloat&) Function

Returns the absolute value of the given BigFloat value.

Syntax

```
public System.Numerics.Multiprecision.BigFloat Abs(const System.Numerics.Multiprecision.BigFloat&
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigFloat&	Arbitrary precision floating point number.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the absolute value of the given BigFloat value.

Ceil(const System.Numerics.Multiprecision.BigFloat&) Function

Returns the BigFloat rounded up to the next arbitrary precision integer value.

Syntax

```
public System.Numerics.Multiprecision.BigFloat Ceil(const System.Numerics.Multiprecision.BigFloat
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigFloat&	BigFloat.

Returns

System.Numerics.Multiprecision.BigFloat

BigFloat

Floor(const System.Numerics.Multiprecision.BigFloat&) Function

Returns the BigFloat rounded down to the previous arbitrary precision integer value.

Syntax

```
public System.Numerics.Multiprecision.BigFloat Floor(const System.Numerics.Multiprecision.BigFloa
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigFloat&	BigFloat.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the BigFloat rounded down to the previous arbitrary precision integer value.

Sqrt(const System.Numerics.Multiprecision.BigFloat&) Function

Returns the square root of the given BigFloat number.

Syntax

```
public System.Numerics.Multiprecision.BigFloat Sqrt(const System.Numerics.Multiprecision.BigFloat
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigFloat&	BigFloat.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the square root of the given BigFloat number.

Trunc(const System.Numerics.Multiprecision.BigFloat&) Function

Returns the BigFloat truncated towards zero.

Syntax

```
public System.Numerics.Multiprecision.BigFloat Trunc(const System.Numerics.Multiprecision.BigFloa
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigFloat&	BigFloat.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the BigFloat truncated towards zero.

operator*(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.B Function

Returns the product of given BigFloat value multiplied by another.

Syntax

```
public System.Numerics.Multiprecision.BigFloat operator*(const System.Numerics.Multiprecision.Big
left, const System.Numerics.Multiprecision.BigFloat& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigFloat&	Left operand.
right	const System.Numerics.Multiprecision.BigFloat&	Right operand.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the product of given BigFloat value multiplied by another.

operator+(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.B Function

Returns the sum of given BigFloat value added to another.

Syntax

```
public System.Numerics.Multiprecision.BigFloat operator+(const System.Numerics.Multiprecision.Big
left, const System.Numerics.Multiprecision.BigFloat& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigFloat&	Left operand.
right	const System.Numerics.Multiprecision.BigFloat&	Right operand.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the sum of given BigFloat value added to another.

operator-(const System.Numerics.Multiprecision.BigFloat&) Function

Returns the negation of BigFloat.

Syntax

```
public System.Numerics.Multiprecision.BigFloat operator-(const System.Numerics.Multiprecision.Big
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigFloat&	A value.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the negation of BigFloat.

operator-(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.B Function

Returns the difference of given BigFloat value subtracted from another.

Syntax

```
public System.Numerics.Multiprecision.BigFloat operator-(const System.Numerics.Multiprecision.Big
left, const System.Numerics.Multiprecision.BigFloat& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigFloat&	Left operand.
right	const System.Numerics.Multiprecision.BigFloat&	Right operand.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the difference of given BigFloat value subtracted from another.

operator/(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.B Function

Returns the quotient when given BigFloat is divided by another.

Syntax

```
public System.Numerics.Multiprecision.BigFloat operator/(const System.Numerics.Multiprecision.Big
left, const System.Numerics.Multiprecision.BigFloat& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigFloat&	Left operand.
right	const System.Numerics.Multiprecision.BigFloat&	Right operand.

Returns

System.Numerics.Multiprecision.BigFloat

Returns the quotient when given BigFloat is divided by another.

operator<(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision.B Function

Returns true if the first BigFloat is less than the second BigFloat, false otherwise.

Syntax

```
public bool operator<(const System.Numerics.Multiprecision.BigFloat& left, const System.Numerics
right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigFloat&	Left operand.
right	const System.Numerics.Multiprecision.BigFloat&	Right operand.

Returns

bool

Returns true if the first BigFloat is less than the second BigFloat, false otherwise.

operator==(const System.Numerics.Multiprecision.BigFloat&, const System.Numerics.Multiprecision
Function

Returns true if the first BigFloat is equal to the second BigFloat, false otherwise.

Syntax

```
public bool operator==(const System.Numerics.Multiprecision.BigFloat& left, const
System.Numerics.Multiprecision.BigFloat& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigFloat&	Left operand.
right	const System.Numerics.Multiprecision.BigFloat&	Right operand.

Returns

bool

Returns true if the first BigFloat is equal to the second BigFloat, false otherwise.

3.1.2 BigInt Class

An arbitrary precision signed integer type.

Syntax

```
public class BigInt;
```

3.1.2.1 Member Functions

Member Function	Description
BigInt()	Default constructor. Creates an instance of arbitrary precision signed integer and initializes it to zero.
BigInt(const System.Numerics.Multiprecision.BigInt&)	Copy constructor.
operator=(const System.Numerics.Multiprecision.BigInt&)	Copy assignment.
BigInt(System.Numerics.Multiprecision.BigInt&&)	Move constructor.
operator=(System.Numerics.Multiprecision.BigInt&&)	Move assignment.
BigInt(const System.String<char>&)	Constructor. Constructs an arbitrary precision floating point value from given decimal digits.
BigInt(const System.String<char>&, int)	Constructor. Constructs an arbitrary precision floating point value from given digits of given base.
BigInt(int)	Creates an instance of arbitrary precision signed integer and initializes it to given signed integer value.
BigInt(uint)	Creates an instance of arbitrary precision signed integer and initializes it to given unsigned integer value.
Handle() const	Returns a handle to the GNU MP library arbitrary precision integer representation.
ToString() const	Returns the value of the **BigInt** as a string of decimal digits prefixed by minus sign if the value is negative.
ToString(int) const	Returns the value of the **BigInt** as a string of digits in given base prefixed by minus sign if the value is negative.

operator=(int)	Assigns the value of the **BigInt** to given signed integer value.
operator=(uint)	Assigns the value of the **BigInt** to given unsigned integer value.
~BigInt()	Frees memory occupied by the **BigInt** instance.

BigInt() Member Function

Default constructor. Creates an instance of arbitrary precision signed integer and initializes it to zero.

Syntax

```
public BigInt();
```

BigInt(const System.Numerics.Multiprecision.BigInt&) Member Function

Copy constructor.

Syntax

```
public BigInt(const System.Numerics.Multiprecision.BigInt& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigInt&	A BigInt to copy from.

operator=(const System.Numerics.Multiprecision.BigInt&) Member Function

Copy assignment.

Syntax

```
public void operator=(const System.Numerics.Multiprecision.BigInt& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigInt&	A BigInt to assign.

BigInt(System.Numerics.Multiprecision.BigInt&&) Member Function

Move constructor.

Syntax

```
public BigInt(System.Numerics.Multiprecision.BigInt&& that);
```

Parameters

Name	Type	Description
that	System.Numerics.Multiprecision.BigInt&&	A BigInt to move from.

operator=(System.Numerics.Multiprecision.BigInt&&) Member Function

Move assignment.

Syntax

```
public void operator=(System.Numerics.Multiprecision.BigInt&& that);
```

Parameters

Name	Type	Description
that	System.Numerics.Multiprecision.BigInt&&	A BigInt to assign.

BigInt(const System.String<char>&) Member Function

Constructor. Constructs an arbitrary precision floating point value from given decimal digits.

Syntax

```
public BigInt(const System.String<char>& str);
```

Parameters

Name	Type	Description
str	const System.String<char>&	Decimal digit string.

BigInt(const System.String<char>&, int) Member Function

Constructor. Constructs an arbitrary precision floating point value from given digits of given base.

Syntax

```
public BigInt(const System.String<char>& str, int base_);
```

Parameters

Name	Type	Description
str	const System.String<char>&	Digit string.
base_	int	Base of digits.

BigInt(int) Member Function

Creates an instance of arbitrary precision signed integer and initializes it to given signed integer value.

Syntax

```
public BigInt(int that);
```

Parameters

Name	Type	Description
that	int	A signed integer value.

BigInt(uint) Member Function

Creates an instance of arbitrary precision signed integer and initializes it to given unsigned integer value.

Syntax

```
public BigInt(uint that);
```

Parameters

Name	Type	Description
that	uint	An unsigned integer value.

Handle() const Member Function

Returns a handle to the GNU MP library arbitrary precision integer representation.

Syntax

```
public void* Handle() const;
```

Returns

void*

Returns a handle to the GNU MP library arbitrary precision integer representation.

ToString() const Member Function

Returns the value of the BigInt as a string of decimal digits prefixed by minus sign if the value is negative.

Syntax

```
public System.String<char> ToString() const;
```

Returns

System.String<char>

Returns the value of the BigInt as a string of decimal digits prefixed by minus sign if the value is negative.

ToString(int) const Member Function

Returns the value of the BigInt as a string of digits in given base prefixed by minus sign if the value is negative.

Syntax

```
public System.String<char> ToString(int base_) const;
```

Parameters

Name	Type	Description
base_	int	Base of digits. Base may vary from 2 to 62.

Returns

System.String<char>

Returns the value of the BigInt as a string of digits in given base prefixed by minus sign if the value is negative.

operator=(int) Member Function

Assigns the value of the BigInt to given signed integer value.

Syntax

```
public void operator=(int that);
```

Parameters

Name	Type	Description
that	int	A signed integer value.

operator=(uint) Member Function

Assigns the value of the BigInt to given unsigned integer value.

Syntax

```
public void operator=(uint that);
```

Parameters

Name	Type	Description
that	uint	An unsigned integer value.

~BigInt() Member Function

Frees memory occupied by the BigInt instance.

Syntax

```
public ~BigInt();
```

3.1.2.2 Nonmember Functions

Function	Description
Abs(const System.Numerics.Multiprecision.-BigInt&)	Returns absolute value of given **BigInt** .
ClearBit(System.Numerics.Multiprecision.-BigInt&, uint)	Clear given bit of given **BigInt** .
SetBit(System.Numerics.Multiprecision.BigInt&, uint)	Set given bit of given **BigInt** .
TestBit(System.Numerics.Multiprecision.-BigInt&, uint)	Returns true if given bit of given **BigInt** is set, false otherwise.
ToggleBit(System.Numerics.Multiprecision.-BigInt&, uint)	Toggle given bit of given **BigInt** .
operator%(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns the remainder when given **BigInt** is divided by another.
operator&(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns bitwise AND of two **BigInt** values.
operator*(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns the product of given **BigInt** value multiplied by another.
operator+(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns the sum of given **BigInt** value added to another.
operator-(const System.Numerics.-Multiprecision.BigInt&)	Returns the negation of **BigInt** .
operator-(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns the difference of given **BigInt** value subtracted from another.
operator/(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns the quotient when given **BigInt** is divided by another.
operator<(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns true if the first **BigInt** is less than the second **BigInt** , false otherwise.
operator<<(System.IO.OutputStream&, const System.Numerics.Multiprecision.BigInt&)	Puts the value of the given **BigInt** to the given output stream as string of decimal digits prefixed by minus sign if the value is negative.

operator==(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns true if the first **BigInt** is equal to the second **BigInt** , false otherwise.
operator^(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns bitwise XOR of two **BigInt** values.
operator—(const System.Numerics.-Multiprecision.BigInt&, const System.Numerics.-Multiprecision.BigInt&)	Returns bitwise inclusive OR of two **BigInt** values.
operator~(const System.Numerics.-Multiprecision.BigInt&)	Returns bitwise complement of **BigInt** value.

Abs(const System.Numerics.Multiprecision.BigInt&) Function

Returns absolute value of given BigInt.

Syntax

```
public System.Numerics.Multiprecision.BigInt Abs(const System.Numerics.Multiprecision.BigInt&
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigInt&	A BigInt.

Returns

System.Numerics.Multiprecision.BigInt

Returns absolute value of given BigInt.

ClearBit(System.Numerics.Multiprecision.BigInt&, uint) Function

Clear given bit of given BigInt.

Syntax

```
public void ClearBit(System.Numerics.Multiprecision.BigInt& x, uint bitIndex);
```

Parameters

Name	Type	Description
x	System.Numerics.Multiprecision.BigInt&	A reference to a BigInt.
bitIndex	uint	Index of bit to clear.

SetBit(System.Numerics.Multiprecision.BigInt&, uint) Function

Set given bit of given BigInt.

Syntax

```
public void SetBit(System.Numerics.Multiprecision.BigInt& x, uint bitIndex);
```

Parameters

Name	Type	Description
x	System.Numerics.Multiprecision.BigInt&	A reference to a BigInt.
bitIndex	uint	Index of bit to set.

TestBit(System.Numerics.Multiprecision.BigInt&, uint) Function

Returns true if given bit of given BigInt is set, false otherwise.

Syntax

```
public bool TestBit(System.Numerics.Multiprecision.BigInt& x, uint bitIndex);
```

Parameters

Name	Type	Description
x	System.Numerics.Multiprecision.BigInt&	A reference to a BigInt.
bitIndex	uint	Index of bit to test.

Returns

bool

Returns true if given bit of given BigInt is set, false otherwise.

ToggleBit(System.Numerics.Multiprecision.BigInt&, uint) Function

Toggle given bit of given BigInt.

Syntax

```
public void ToggleBit(System.Numerics.Multiprecision.BigInt& x, uint bitIndex);
```

Parameters

Name	Type	Description
x	System.Numerics.Multiprecision.BigInt&	A reference to a BigInt.
bitIndex	uint	Index of bit to toggle.

operator%(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Bi
Function

Returns the remainder when given BigInt is divided by another.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator%(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Divisor.
right	const System.Numerics.Multiprecision.BigInt&	Dividend.

Returns

System.Numerics.Multiprecision.BigInt

Returns the remainder when left is divided by right.

operator&(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Big
Function

Returns bitwise AND of two BigInt values.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator&(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

System.Numerics.Multiprecision.BigInt

Returns bitwise AND of given BigInt values.

operator*(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Big
Function

Returns the product of given BigInt value multiplied by another.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator*(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

System.Numerics.Multiprecision.BigInt

Returns the product of given BigInt value multiplied by another.

operator+(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Big
Function

Returns the sum of given BigInt value added to another.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator+(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

System.Numerics.Multiprecision.BigInt

Returns the sum of given BigInt value added to another.

operator-(const System.Numerics.Multiprecision.BigInt&) Function

Returns the negation of BigInt.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator-(const System.Numerics.Multiprecision.BigIn
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigInt&	A value.

Returns

System.Numerics.Multiprecision.BigInt

Returns the negation of BigInt.

operator-(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Bigl Function

Returns the difference of given BigInt value subtracted from another.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator-(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

System.Numerics.Multiprecision.BigInt

Returns the difference of given BigInt value subtracted from another.

operator/(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Big
Function

Returns the quotient when given BigInt is divided by another.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator/(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Divisor.
right	const System.Numerics.Multiprecision.BigInt&	Dividend.

Returns

System.Numerics.Multiprecision.BigInt

Returns the quotient when given BigInt is divided by another.

Remarks

Quotient is rounded towards zero.

operator<(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Big
Function

Returns true if the first BigInt is less than the second BigInt, false otherwise.

Syntax

```
public bool operator<(const System.Numerics.Multiprecision.BigInt& left, const System.Numerics.M
right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

bool

Returns true if the first BigInt is less than the second BigInt, false otherwise.

operator<<(System.IO.OutputStream&, const System.Numerics.Multiprecision.BigInt&) Function

Puts the value of the given BigInt to the given output stream as string of decimal digits prefixed by minus sign if the value is negative.

Syntax

```
public System.IO.OutputStream& operator<<(System.IO.OutputStream& s, const System.Numerics.Multi
x);
```

Parameters

Name	Type	Description
s	System.IO.OutputStream&	An output stream.
x	const System.Numerics.Multiprecision.BigInt&	A BigInt value.

Returns

System.IO.OutputStream&

Returns a reference to the output stream.

operator==(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.B
Function

Returns true if the first BigInt is equal to the second BigInt, false otherwise.

Syntax

```
public bool operator==(const System.Numerics.Multiprecision.BigInt& left, const System.Numerics.M
right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

bool

Returns true if the first BigInt is equal to the second BigInt, false otherwise.

operator^(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.Big
Function

Returns bitwise XOR of two BigInt values.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator^(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

System.Numerics.Multiprecision.BigInt

Returns bitwise XOR of two BigInt values.

operator|(const System.Numerics.Multiprecision.BigInt&, const System.Numerics.Multiprecision.BigI Function

Returns bitwise inclusive OR of two BigInt values.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator|(const System.Numerics.Multiprecision.BigIn
left, const System.Numerics.Multiprecision.BigInt& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigInt&	Left operand.
right	const System.Numerics.Multiprecision.BigInt&	Right operand.

Returns

System.Numerics.Multiprecision.BigInt

Returns bitwise inclusive OR of two BigInt values.

operator~(const System.Numerics.Multiprecision.BigInt&) Function

Returns bitwise complement of BigInt value.

Syntax

```
public System.Numerics.Multiprecision.BigInt operator~(const System.Numerics.Multiprecision.BigI
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigInt&	Operand.

Returns

System.Numerics.Multiprecision.BigInt

Returns bitwise complement of BigInt value.

3.1.3 BigRational Class

An arbitrary precision rational nunber type.

Syntax

```
public class BigRational;
```

3.1.3.1 Member Functions

Member Function	Description
BigRational()	Default constructor. Creates an instance of arbitrary precision rational number and initializes it to zero.
BigRational(const System.Numerics.Multiprecision.BigRational&)	Copy constructor.
operator=(const System.Numerics.Multiprecision.BigRational&)	Copy assignment.
BigRational(System.Numerics.Multiprecision.BigRational&&)	Move constructor.
operator=(System.Numerics.Multiprecision.BigRational&&)	Move assignment.
BigRational(const System.Numerics.Multiprecision.BigInt&)	Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given arbitrary precision integer type.
BigRational(const System.String<char>&)	Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given string.
BigRational(const System.String<char>&, int)	Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given string of given base.
BigRational(int)	Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given integer value.
BigRational(uint)	Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given unsigned integer value.
Denominator() const	Returns the denominator of the arbitrary precision ration number.
Handle() const	Returns a handle to the GNU MP library arbitrary precision rational representation.

Numerator() const	Returns the numerator of the arbitrary precision ration number.
ToString() const	Returns the value of the **BigRational** as a string.
ToString(int) const	Returns the value of the **BigRational** as a string of given base.
~BigRational()	Frees memory occupied by the **BigRational** instance.

BigRational() Member Function

Default constructor. Creates an instance of arbitrary precision rational number and initializes it to zero.

Syntax

```
public BigRational();
```

BigRational(const System.Numerics.Multiprecision.BigRational&) Member Function

Copy constructor.

Syntax

```
public BigRational(const System.Numerics.Multiprecision.BigRational& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigRational&	A BigRational to copy from.

operator=(const System.Numerics.Multiprecision.BigRational&) Member Function

Copy assignment.

Syntax

```
public void operator=(const System.Numerics.Multiprecision.BigRational& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigRational&	A BigRational to assign from.

BigRational(System.Numerics.Multiprecision.BigRational&&) Member Function

Move constructor.

Syntax

```
public BigRational(System.Numerics.Multiprecision.BigRational&& that);
```

Parameters

Name	Type	Description
that	System.Numerics.Multiprecision.BigRational&&	A BigRational to move from.

operator=(System.Numerics.Multiprecision.BigRational&&) Member Function

Move assignment.

Syntax

```
public void operator=(System.Numerics.Multiprecision.BigRational&& that);
```

Parameters

Name	Type	Description
that	System.Numerics.Multiprecision.BigRational&&	A BigRational to assign from.

BigRational(const System.Numerics.Multiprecision.BigInt&) Member Function

Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given arbitrary precision integer type.

Syntax

```
public BigRational(const System.Numerics.Multiprecision.BigInt& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.BigInt&	An arbitrary precision integer type value to initialize from.

BigRational(const System.String<char>&) Member Function

Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given string.

Syntax

```
public BigRational(const System.String<char>& str);
```

Parameters

Name	Type	Description
str	const System.String<char>&	A string to initialize from.

BigRational(const System.String<char>&, int) Member Function

Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given string of given base.

Syntax

```
public BigRational(const System.String<char>& str, int base_);
```

Parameters

Name	Type	Description
str	const System.String<char>&	A string to initialize from.
base_	int	Base of string.

BigRational(int) Member Function

Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given integer value.

Syntax

```
public BigRational(int that);
```

Parameters

Name	Type	Description
that	int	An integer value to initalize from.

BigRational(uint) Member Function

Constructor. Creates an instance of arbitrary precision rational type and initializes it from the given unsigned integer value.

Syntax

```
public BigRational(uint that);
```

Parameters

Name	Type	Description
that	uint	An unsigned integer value to initalize from.

Denominator() const Member Function

Returns the denominator of the arbitrary precision ration number.

Syntax

```
public System.Numerics.Multiprecision.BigInt Denominator() const;
```

Returns

System.Numerics.Multiprecision.BigInt

Returns the denominator of the arbitrary precision ration number.

Handle() const Member Function

Returns a handle to the GNU MP library arbitrary precision rational representation.

Syntax

```
public void* Handle() const;
```

Returns

void*

Returns a handle to the GNU MP library arbitrary precision rational representation.

Numerator() const Member Function

Returns the numerator of the arbitrary precision ration number.

Syntax

```
public System.Numerics.Multiprecision.BigInt Numerator() const;
```

Returns

System.Numerics.Multiprecision.BigInt

Returns the numerator of the arbitrary precision ration number.

ToString() const Member Function

Returns the value of the BigRational as a string.

Syntax

```
public System.String<char> ToString() const;
```

Returns

System.String<char>

Returns the value of the BigRational as a string.

ToString(int) const Member Function

Returns the value of the BigRational as a string of given base.

Syntax

```
public System.String<char> ToString(int base_) const;
```

Parameters

Name	Type	Description
base_	int	Base.

Returns

System.String<char>

Returns the value of the BigRational as a string of given base.

~BigRational() Member Function

Frees memory occupied by the BigRational instance.

Syntax

```
public ~BigRational();
```

3.1.3.2 Nonmember Functions

Function	Description
Abs(const System.Numerics.Multiprecision.BigRational&)	Returns the absolute value of the given **BigRational** value.
operator*(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecision.BigRational&)	Returns the product of given **BigRational** value multiplied by another.
operator+(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecision.BigRational&)	Returns the sum of given **BigRational** value added to another.
operator-(const System.Numerics.Multiprecision.BigRational&)	Returns the negation of **BigRational** .
operator-(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecision.BigRational&)	Returns the difference of given **BigRational** value subtracted from another.
operator/(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecision.BigRational&)	Returns the quotient when given **BigRational** is divided by another.
operator<(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecision.BigRational&)	Returns true if the first **BigRational** is less than the second **BigRational** , false otherwise.
operator<<(System.IO.OutputStream&, const System.Numerics.Multiprecision.BigRational&)	Puts the value of the given **BigRational** to the given output stream as string.
operator==(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecision.BigRational&)	Returns true if the first **BigRational** is equal to the second **BigRational** , false otherwise.

Abs(const System.Numerics.Multiprecision.BigRational&) Function

Returns the absolute value of the given BigRational value.

Syntax

```
public System.Numerics.Multiprecision.BigRational Abs(const System.Numerics.Multiprecision.BigRat
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigRational&	BigRational.

Returns

System.Numerics.Multiprecision.BigRational

Returns the absolute value of the given BigRational value.

operator*(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecisio Function

Returns the product of given BigRational value multiplied by another.

Syntax

```
public System.Numerics.Multiprecision.BigRational operator*(const System.Numerics.Multiprecision.
left, const System.Numerics.Multiprecision.BigRational& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigRational&	Left operand.
right	const System.Numerics.Multiprecision.BigRational&	Right operand.

Returns

System.Numerics.Multiprecision.BigRational

Returns the product of given BigRational value multiplied by another.

operator+(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecisi
Function

Returns the sum of given BigRational value added to another.

Syntax

```
public System.Numerics.Multiprecision.BigRational operator+(const System.Numerics.Multiprecision.
left, const System.Numerics.Multiprecision.BigRational& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigRational&	Left operand.
right	const System.Numerics.Multiprecision.BigRational&	Right operand.

Returns

System.Numerics.Multiprecision.BigRational

Returns the sum of given BigRational value added to another.

operator-(const System.Numerics.Multiprecision.BigRational&) Function

Returns the negation of BigRational.

Syntax

```
public System.Numerics.Multiprecision.BigRational operator-(const System.Numerics.Multiprecision.
x);
```

Parameters

Name	Type	Description
x	const System.Numerics.Multiprecision.BigRational&	BigRational.

Returns

System.Numerics.Multiprecision.BigRational

Returns the negation of BigRational.

operator-(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecisio
Function

Returns the difference of given BigRational value subtracted from another.

Syntax

```
public System.Numerics.Multiprecision.BigRational operator-(const System.Numerics.Multiprecision.
left, const System.Numerics.Multiprecision.BigRational& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigRational&	Left operand.
right	const System.Numerics.Multiprecision.BigRational&	Right operand.

Returns

System.Numerics.Multiprecision.BigRational

Returns the difference of given BigRational value subtracted from another.

operator/(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecisio Function

Returns the quotient when given BigRational is divided by another.

Syntax

```
public System.Numerics.Multiprecision.BigRational operator/(const System.Numerics.Multiprecision.
left, const System.Numerics.Multiprecision.BigRational& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigRational&	Divisor.
right	const System.Numerics.Multiprecision.BigRational&	Dividend.

Returns

System.Numerics.Multiprecision.BigRational

Returns the quotient when given BigRational is divided by another.

operator<(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multiprecisio
Function

Returns true if the first BigRational is less than the second BigRational, false otherwise.

Syntax

```
public bool operator<(const System.Numerics.Multiprecision.BigRational& left, const
System.Numerics.Multiprecision.BigRational& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigRational&	Left operand.
right	const System.Numerics.Multiprecision.BigRational&	Right operand.

Returns

bool

Returns true if the first BigRational is less than the second BigRational, false otherwise.

operator<<(System.IO.OutputStream&, const System.Numerics.Multiprecision.BigRational&) Function

Puts the value of the given BigRational to the given output stream as string.

Syntax

```
public System.IO.OutputStream& operator<<(System.IO.OutputStream& s, const System.Numerics.Multi
x);
```

Parameters

Name	Type	Description
s	System.IO.OutputStream&	An output stream.
x	const System.Numerics.Multiprecision.BigRational&	A BigRational value.

Returns

System.IO.OutputStream&

Returns a reference to the output stream.

operator==(const System.Numerics.Multiprecision.BigRational&, const System.Numerics.Multipreci
Function

Returns true if the first BigRational is equal to the second BigRational, false otherwise.

Syntax

```
public bool operator==(const System.Numerics.Multiprecision.BigRational& left, const
System.Numerics.Multiprecision.BigRational& right);
```

Parameters

Name	Type	Description
left	const System.Numerics.Multiprecision.BigRational&	Left operand.
right	const System.Numerics.Multiprecision.BigRational&	Right operand.

Returns

bool

Returns true if the first BigRational is equal to the second BigRational, false otherwise.

3.1.4 Precision Class

Represents a precision of given number of digits.

Syntax

```
public class Precision;
```

3.1.4.1 Member Functions

Member Function	Description
Precision()	Default constructor. Initializes the precision with zero digits.
Precision(const System.Numerics.Multiprecision.-Precision&)	Copy constructor.
operator=(const System.Numerics.-Multiprecision.Precision&)	Copy assignment.
Precision(System.Numerics.Multiprecision.-Precision&&)	Move constructor.
operator=(System.Numerics.Multiprecision.-Precision&&)	Move assignment.
Precision(uint)	Constructor. Initializes the precision with given number of digits.
operator_uint() const	Returns the number of digits of this precision.

Precision() Member Function

Default constructor. Initializes the precision with zero digits.

Syntax

```
public Precision();
```

Precision(const System.Numerics.Multiprecision.Precision&) Member Function

Copy constructor.

Syntax

```
public Precision(const System.Numerics.Multiprecision.Precision& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.Precision&	Argument to copy.

operator=(const System.Numerics.Multiprecision.Precision&) Member Function

Copy assignment.

Syntax

```
public void operator=(const System.Numerics.Multiprecision.Precision& that);
```

Parameters

Name	Type	Description
that	const System.Numerics.Multiprecision.Precision&	Argument to assign.

Precision(System.Numerics.Multiprecision.Precision&&) Member Function

Move constructor.

Syntax

```
public Precision(System.Numerics.Multiprecision.Precision&& that);
```

Parameters

Name	Type	Description
that	System.Numerics.Multiprecision.Precision&&	Argument to move from.

operator=(System.Numerics.Multiprecision.Precision&&) Member Function

Move assignment.

Syntax

```
public void operator=(System.Numerics.Multiprecision.Precision&& that);
```

Parameters

Name	Type	Description
that	System.Numerics.Multiprecision.Precision&&	Argument to assign from.

Precision(uint) Member Function

Constructor. Initializes the precision with given number of digits.

Syntax

```
public Precision(uint prec_);
```

Parameters

Name	Type	Description
prec_	uint	Number of digits.

operator_uint() const Member Function

Returns the number of digits of this precision.

Syntax

```
public uint operator_uint() const;
```

Returns

uint

Returns the number of digits of this precision.

3.2 Functions

Function	Description