Infinite Precision Library Documentation

Sathvik Reddy Bhavanam

CS23BTECH11056

April 21, 2024

Contents

1 Integer Library

1.1 Introduction

In this section, describe the purpose and functionalities of your Integer library. Explain how it provides infinite precision arithmetic for integers.

1.2 API Reference

Here, list and explain the functions and methods provided by the Integer library. Use clear and concise explanations.

1.2.1 Constructors

`Integer()`

This constructor creates a vector initializes it to `{0}` and initializes `isNegative` to `false`.

```
#include "Integer.h"
#include "Float.h"

int main()
{
  LOG(InfiniteArithmetic::Integer());
  // Output = 0
}
```

`Integer(std::string num)`

This constructor sets the member variables to appropriate values based on the string. It calls the `VerifyString` function internally to check if the string provided is valid or not.

```

  LOG(InfiniteArithmetic::Integer("212"));
  // Output = 212
.
```

`Integer(const Integer &obj)`

It is a copy constructor that replicates the values of the object given.

```

  namespace InfiniteArithmetic = IA;
  LOG(IA::Integer(IA::Integer("110")));
  // Output = 110
.
```

1.2.2 Destructor

`~Integer()`

The destructor deletes the vector and the boolean variable `isNegative` explicitly.

1.2.3 Integer parse(const std::string &)

The parse function returns an instance of the `Integer` class.

1.2.4 void Assign(Integer)

The `Assign` function assigns the value of one integer to the other.

```
  IA::Integer num1 ("102");
  IA::Integer num2;
  num2.Assign(num1);
  LOG(num2);
  // Output = 102
.
```

The `= operator` is an other function that has been overloaded to assign one variable to another.

1.2.5 Print

The `Print` is a private function that is used to display the contents of the vector.

However, the << (insertion operator) has been overloaded.

1.2.6 Input

Input can be taken through >> (extraction operator).

1.2.7 Add

1.2.8 Add2

1.2.9 Subtract

1.2.10 Multiply

1.2.11 MultiplyByDigit

1.2.12 Divide

1.2.13 Mod

1.2.14 Compare

1.2.15 Complement

1.2.16 Negate

1.2.17 isZero

1.2.18 MatchDigits

1.2.19 VerifyString

1.2.20 PopZero

1.3 Examples

Provide code examples demonstrating the usage of the Integer library functions.

```
>>> a = Integer(10)
>>> b = Integer(5)
>>> c = a.add(b)
>>> print(c.toString())  # Output: 15

>>> d = a.divide(b)
>>> print(d.toString())  # Output: 2
```

2 Float Library

2.1 Introduction

Describe the purpose and functionalities of your Float library here. Explain how it provides infinite precision arithmetic for floating-point numbers.

2.2 API Reference

Document the functions and methods provided by the Float library, similar to the Integer library section.

2.2.1 Constructors

2.2.2 Destructor

2.2.3 parse

2.2.4 Assign

2.2.5 Print

2.2.6 Input

2.2.7 Add

2.2.8 Subtract

2.2.9 Multiply

2.2.10 MultiplyByDigit

2.2.11 Divide

2.2.12 SetPrecision

2.2.13 Compare

2.2.14 Complement

2.2.15 Negate

2.2.16 isZero

2.2.17 MatchDigits

2.2.18 ResizeEnds

2.2.19 VerifyString

2.2.20 PopZero

- **add(a, b)**: Adds two floats ‘a‘ and ‘b‘ and returns the result as a Float object.
- **subtract(a, b)**: Subtracts float ‘b‘ from ‘a‘ and returns the result as a Float object.
- **multiply(a, b)**: Multiplies two floats ‘a‘ and ‘b‘ and returns the result as a Float object.
- **divide(a, b)**: Divides float ‘a‘ by ‘b‘ and returns the result as a Float object. (Handle division by zero case)
- **abs(a)**: Returns the absolute value of float ‘a‘ as a Float object.
- **toString()**: Converts the Float object to a string representation.

2.3 Examples

Include code examples demonstrating the usage of the Float library functions.

```
>>> a = Float(3.14)
>>> b = Float(2.72)
>>> c = a.add(b)
>>> print(c.toString())  # Output: 5.86

>>> d = a.divide(b)
>>> print(d.toString())  # Output: 1.15... (show limited precision)
```