

# Report

Sai Rishi Kasturi cs23btech11024  
Muvvala Sairam Suhas cs23bbtech11038  
Team 20

April 21, 2024

## Contents

<b>1</b>	<b>Design</b>	<b>2</b>
1.1	Integer . . . . .	2
1.2	Float . . . . .	2
1.3	UML diagram . . . . .	3
<b>2</b>	<b>Readme</b>	<b>3</b>
2.1	Arbitrary size . . . . .	3
2.2	Private variables . . . . .	3
2.3	Outstream . . . . .	4
2.4	Operator overloading . . . . .	4
2.5	Constructor . . . . .	4
2.5.1	Integers . . . . .	4
2.5.2	Floats . . . . .	4
<b>3</b>	<b>Limitations</b>	<b>5</b>
3.1	Performance . . . . .	5
3.2	Exception handling . . . . .	5
3.3	Overloaded operators . . . . .	5
3.4	Type conversions . . . . .	5
<b>4</b>	<b>Verification approach</b>	<b>5</b>
4.1	Constructors . . . . .	5
4.2	Operators . . . . .	5
4.3	Short circuiting . . . . .	6
<b>5</b>	<b>Keylearnings</b>	<b>6</b>
<b>6</b>	<b>Git Snapshots</b>	<b>6</b>

# 1 Design

We have written the code such that it is nicely encapsulated by using the private members in the class and have written it in an object oriented way as you can see after reading this.

- we used `#pragma` once in the beginning so that when we are using this header file again and again we wont have the problem of repeating the whole code again and again and get unexpected errors while doing so unintentionally.
- we have used 2 headers in order to use the built in functions present in them which are

```
1  #include <algorithm>
2  #include <iostream>
3
```

- we have written the whole code in the namespace `Infinite_Arithmetic` so that when we are using the objects of that class we wont be having trouble with objects of classes having same names while using them.
- we have written two classes which are having the names `Integers` and `Float`.

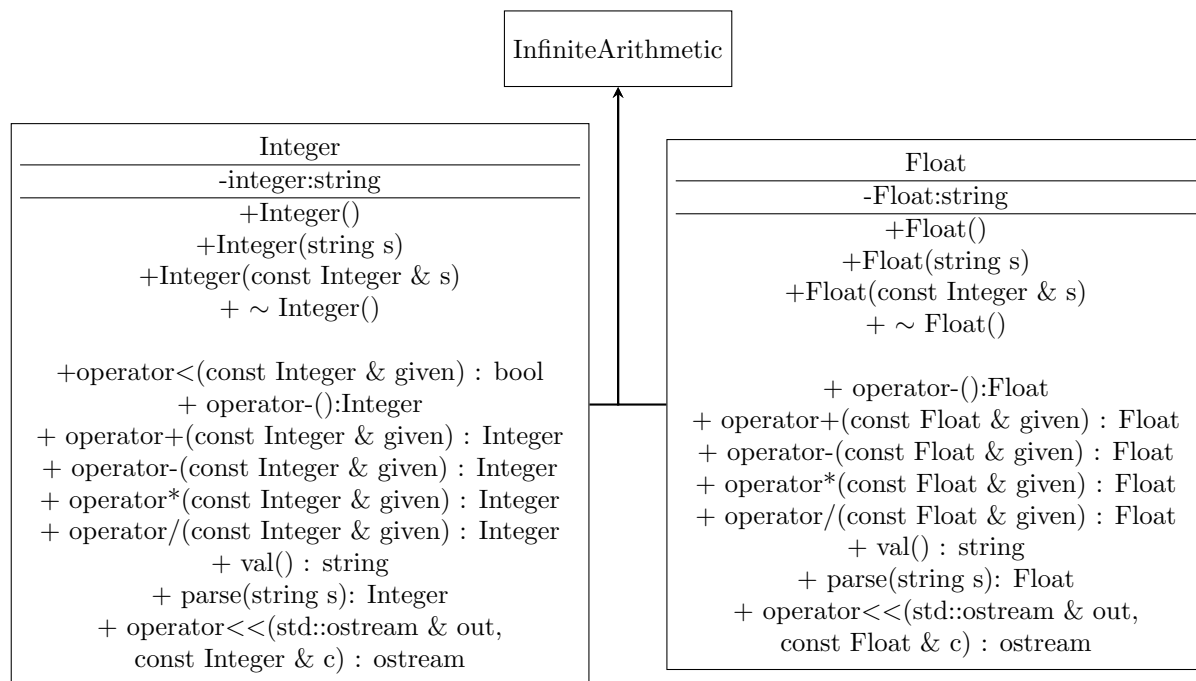
## 1.1 Integer

- we have used a private variable having the type string so that we can take inputs of very long values which cant be handles with our normal int datatypes.
- we have written a default Constructor a Constructor and a copy Constructor.we have also written a destructor.
- we have written Functions which overload various operators such as `<` `-` `+` `*` `/` -(2nd time).we have written a function which would make private variable `int0` accessible to non class functions.we have also written friend function by overloading the `cout` operator in order to directly print the Integer object `int0`.we have used static function `parse` too which could create an Integer object from string without using the instance of a class

## 1.2 Float

- we have used the Constructor similar to that of Integer class just that we have also included a private integer called `place` whose value is the index of the decimal point in a float variable.
- The functions and operator overloading that we have done is similar to that of the Integer class

### 1.3 UML diagram



## 2 Readme

**Infinite Arithmetic Library** This library provides two classes which can be used to create large instances of Integers which cant be used using the basic datatypes and hence performing arbitrary precision arithmetic operations on integers and floating point numbers created by the class.

### 2.1 Arbitrary size

The integer class as well as the float class represents integers of arbitrary size without having an overflow or underflow.

### 2.2 Private variables

If we want to use the integer or the float value outside that of a class function we can do it by using a val function that is described in the library.

## 2.3 Outstream

If we want to print an Integer which is a private variable it is ok to directly use a cout because we have overloaded the << operator making it easy for us to print the outcome after using the overloaded functions.

## 2.4 Operator overloading

We can do addition subtraction multiplication and division for various objects as we are returning the same object again after performing the operation and hence multiple operations between Integers or between Floats is possible.

## 2.5 Constructor

The Constructor has been made such that when not given any value the default Constructor is called and the values are initialized.

### 2.5.1 Integers

The Constructor is such that the leading zeroes are removed by itself when called using that Constructor.

```
1 Integer a("1234567");
2 Integer b = Integer::parse("-009876321");
3 \\leading 0s in b are removed
4 Integer c = a + b;
5 Integer d = a - b;
6 Integer e = a * b;
7 Integer f = a / b;
8 std::cout << "a = " << a << std::endl;
9 \\can directly print it
```

### 2.5.2 Floats

The Constructor when given a string only can itself remove all the trailing and leading zeroes and also find the place value of the decimal and initiates it. f we have been given a Float without a decimal point then it automatically adds a decimal point and a 0 after that so we dont have to worry about it being given an integer value

```
1 Float a("123.4567");
2 Float b = Integer::parse("-009876.3210");
3 \\leading and trailing 0s in b are removed
4 Float c = a + b;
5 Float d = a - b;
6 Float e = a * b;
7 Float f = a / b;
8 std::cout << "a = " << a << std::endl;
9 \\can directly print it
```

## 3 Limitations

### 3.1 Performance

the performance for the code of using the Floats and Integer classes are strings and using them take up a lot of memory in order to do the operations and hence the manipulations and memory allocation could slow the code down.

### 3.2 Exception handling

The integer class cant really handle the exception of being given the decimal point numbers if given they would treat the decimal point as a character or a digit by taking in its ASCII code.

### 3.3 Overloaded operators

There are only a few operators that have been overloaded and need more operators such as power == != > etc

### 3.4 Type conversions

We haven't been able to give explicit or implicit datatype conversions for the following classes.

## 4 Verification approach

- We have tested as we wrote the .h file while we wrote every function while we overloaded every operator even when we were writing the Constructor we checked weather the code was working properly as it is supposed to like when we were going through :-

### 4.1 Constructors

we checked weather the trailing 0s were being removed or not weather the leading 0s were weather the place value we were getting through the Constructor was correct or not weather the place value of the decimal was being given correct index or not.

### 4.2 Operators

with every case we checked we stumbled upon a part which we hadn't done in our algorithm which could make it better when faced with ambiguous values we would debug the code at every point using GDB and find out where the problem is and edit our code as such.

### 4.3 Short circuiting

After writing the operators algorithm we always checked whether we were being able to short circuit the code when we can use 0s anywhere possible.

## 5 Keylearnings

- we got to know how to keep testing and verifying the results by using GDB
- we came to know about various ways of how to write code, do proper indentation and also on how to comment properly
- we got to know how to write a parse function and a value function
- we learnt how difficult it is to find an algorithm no matter how small it is and came to understand the joy of finding one after thorough search for it

## 6 Git Snapshots

Commits

main	All users	All time
Commits on Apr 21, 2024		
Updated report.pdf,now taking input in command line instead of using cin suhas committed 30 minutes ago	e75cbff	
changed some code to minimize errors wrote main.cpp suhas committed 1 hour ago	90e50e6	
Delete CMakelists.txt toixdee-01 committed 4 hours ago	8f9c41a	Verified
Report Submitted suhas committed 4 hours ago	1424a21	
wrote some comments and Makefile suhas committed 6 hours ago	dd19799	
Delete my_inf_arith.cpp toixdee-01 committed 8 hours ago	1bda341	Verified

Figure 1: GIT COMMIT SCREENSHOTS

Delete my_inf_arith.h toixdee-01 committed 8 hours ago	88c9616	Verified
added some code snippets and wrote some comments suhas committed 9 hours ago	618265a	
edited constructor in Float suhas committed 10 hours ago	0b87023	
added comments to .h file suhas committed 10 hours ago	2c452b4	
Commits on Apr 19, 2024		
intoduced division by zero toixdee-01 committed 3 days ago	08896d1	
fixed some errors toixdee-01 committed 3 days ago	7ad03ea	
wrote integer division suhas committed 3 days ago	6e8c35d	
fixed some errors toixdee-01 committed 3 days ago	f4d5044	

Figure 2: GIT COMMIT SCREENSHOTS

added division for float\n\nCo-authored-by: suhas-1012 <cs23btech11038@iith.ac.in> toixdee-01 committed 3 days ago	7e77a2f	
Commits on Apr 18, 2024		
wrote the main function which would take the input as given in doc,wrote a function which would compare two Integers ,fixed some errors suhas committed 3 days ago	b8223ed	
handed some error like -1 and -1. and some minor errors toixdee-01 committed 3 days ago	bc6875a	
handed some error for - numbers toixdee-01 committed 4 days ago	44a4bc5	
overloaded '<' for ease toixdee-01 committed 4 days ago	65820e5	
Fix errors toixdee-01 committed 4 days ago	568d123	
did multiplication for floating integers suhas committed 4 days ago	69a0f66	

Figure 3: GIT COMMIT SCREENSHOTS

Commits on Apr 17, 2024		
Fix errors and modified subtraction toxidee-01 committed 4 days ago	576b84a	
wriiten to ignore leading zero and trailing zero whenever needed and also made that when an integer is given to float it changes to float like 1 to 1.0 toxidee-01 committed 4 days ago	9fa99fc	
handed the error of printing '-0' instead of '0',also there was an error in addition in one case, also added the - operation similar to addition and the float addition partially toxidee-01 committed 4 days ago	019095b	
included the multiplication for all type of integers toxidee-01 committed 4 days ago	da5a3e3	
included the subtraction for negative integers toxidee-01 committed 5 days ago	8f6771b	
did addition for negative integers suhas committed 5 days ago	1c575ee	
updated the subtraction as addition toxidee-01 committed 5 days ago	5bc280f	

Figure 4: GIT COMMIT SCREENSHOTS

changed the logic to reverse stringing to reduce compile time and introduced a function to print the integer class,also the - operator suhas committed 5 days ago	ae2848d	
removed the push_back and used direct expression for string.Also added the camakelist suhas committed 5 days ago	45f9935	
Commits on Apr 16, 2024		
used suhas committed 5 days ago	9758dde	
Merge branch 'main' of <a href="https://github.com/cse-iith/infinite-precision-arithmetic-team-20">https://github.com/cse-iith/infinite-precision-arithmetic-team-20</a> suhas committed last week	fb56eb7	
wrote the subtraction for positive integers suhas committed last week	dda2db7	
added float class in calc.h file toxidee-01 committed last week	bbe26fb	
writing a calc.h file having class definition for integers suhas committed last week	583d448	

Figure 5: GIT COMMIT SCREENSHOTS

Commits		
main		
All users All time		
Commits on Apr 16, 2024		
wriiten addition for a positive numbers toxidee-01 committed last week	63dac9c	
Commits on Apr 12, 2024		
add deadline github-classroom[bot] committed last week	Verified 6c99732	
< Previous Next		

Figure 6: GIT COMMIT SCREENSHOTS