COMPENG 2SI4 Lab 1,2 report

2021.3.8

Ruiyan Guo - L07

guor28

400256752

As a future member of the engineering profession, the student is responsible for performing the required work in an honest manner, without plagiarism and cheating. Submitting this work with my name and student number is a statement and understanding that this work is my own and adheres to the Academic Integrity Policy of McMaster University and the Code of Conduct of the Professional Engineers of Ontario. Submitted by

[Ruiyan Guo, guor28, 400256752]

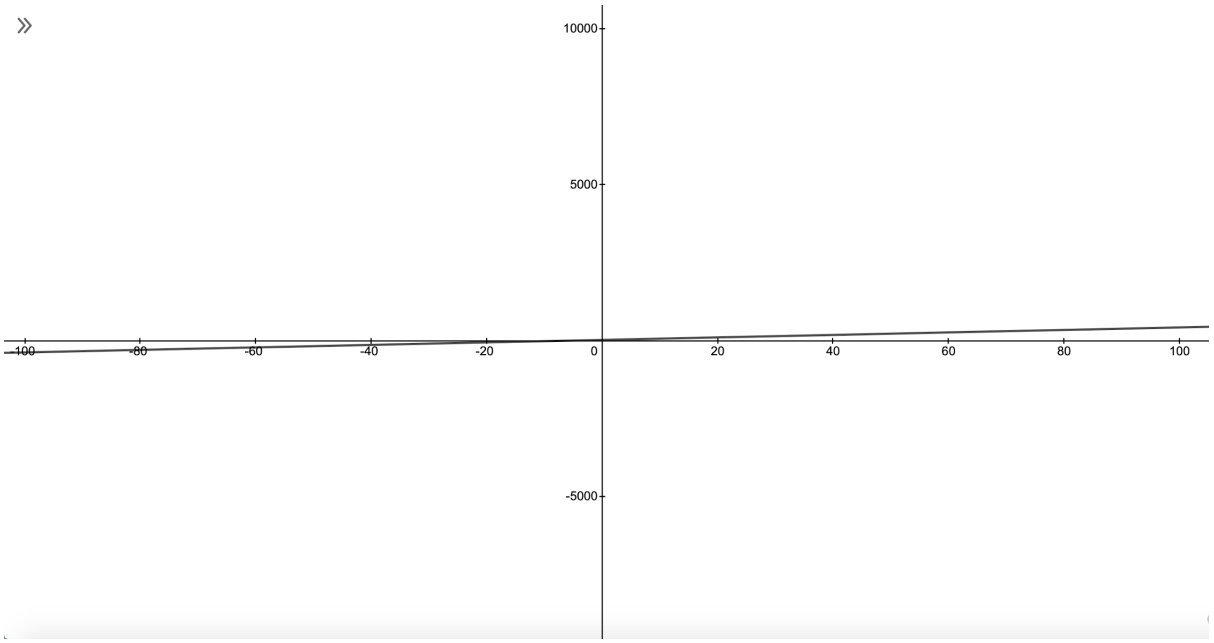
Description of Data Structures and Algorithms

Generally, array was used to implement the HugeInteger class. 1. For the addition class, the code was divided into two cases: both positive, one negative one positive. Inside each case, the code calculates with the absolute values, it compares the length and the magnitude of the integers. Then, for the first case, the code just adds the values, for the second case, the code does subtraction according to the comparison results. Finally, the code adds sign for the calculation results. 2. For the subtraction class, it is almost the same as the addition class, it modifies the sign first and then calls the add function. 3. For the multiplication class, it is similar to how we do the multiplication by hand. 4. For the comparison class, it compares the sign first, if both have the same sign, then it will compare the length of the inputs, if the lengths are still the same, it will compare the magnitude of each bit starting from the outermost bit.

Theoretical Analysis of Running Time and Memory Requirement

1). Memory complexity

The memory need to store the integer of n digits is ‘ 4n+18’ bytes



2). Running time

(a). Addition: The worst case and the average case are both big-Theta of n. The whole class is composed of some if statements and for loops (no nested loop). All the loops increment or decrement by 1. As a result the running time would be big-Theta of n.

(b). Subtraction: The worst case and average case are both big-Theta of n. The subtraction class itself only contains some if statements, it almost shares the same body of class addition since it calls the class addition inside of it.

(c). Multiplication: The worst case and average case are both big-Theta of n^2. The whole class calls the class addition inside a for loop. as a result the run time would be big-Theta of n^2.

(d). Comparison: The worst case and average case are both big-Theta of 1, which is constant complexity. The whole class only contains some if statements.

Test Procedure

For test purposes, I inserted values that start with zero, the values contain a lot of zeros in the middle and so forth. I specifically concentrated on testing the borrow/carry bits in the class addition and subtractions. All the outputs from my implementation match up with the one I get from the java implementation. As the code has passed a lot of extreme values and edge inputs, the operations are functioning properly.

Experimental Measurement, Comparison and Discussion

(a). Descriptions and Parameters

A number of experimental sets (represented by MAXNUMINTS) were created. Inside of each set, two pairs of random integers of n digits (represented by n) were generated and they were used in executions of the operation, the operations were set to run for n times (represented by MAXRUN) .

MAXNUMINTS = 600

MAXRUN = 200

(b). Table of results

i). Addition

Number of Digits	My implementation	Java implementation
10	0.003108333333333342	1.5541666666666708E-5
100	0.00989166666666663	4.945833333333315E-5
500	0.07710000000000014	3.855000000000007E-4
1000	0.2290333333333334	0.001145166666666667

ii). Subtraction

Number of Digits	My implementation	Java implementation
10	0.005191666666666659	2.5958333333333295E-5
100	0.023108333333333283	1.1554166666666641E-4
500	0.1204916666666666	6.02458333333333E-4
1000	0.3509166666666663	0.0017545833333333315

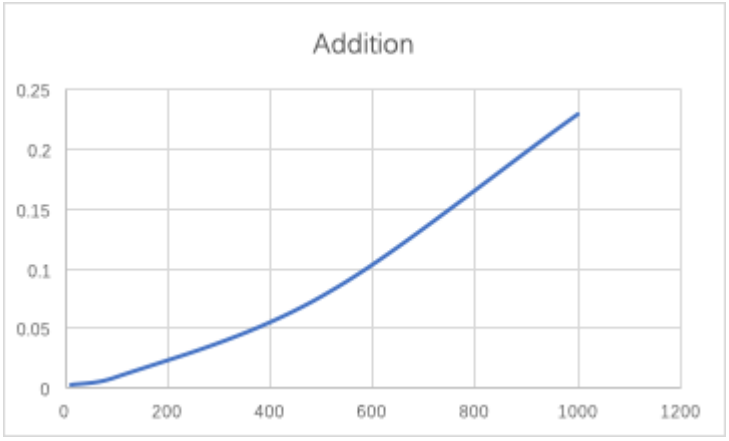
iii) Multiplication

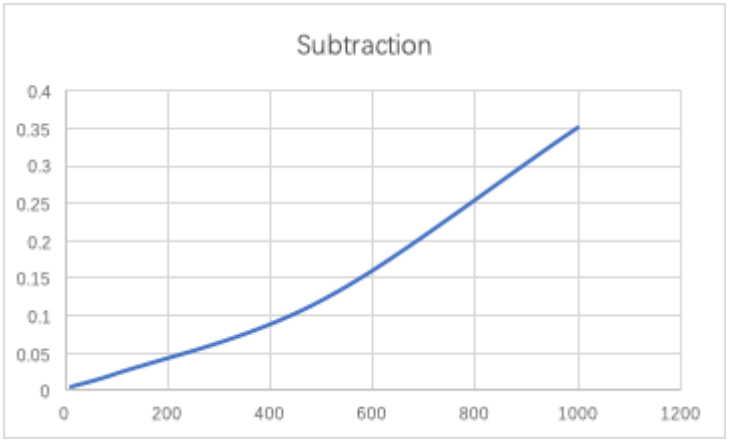
Number of Digits	My implementation	Java implementation
10	0.06138333333333334	3.069166666666667E-4
100	11.690000000000003	1.1690000000000003
500	288.885	28.8885

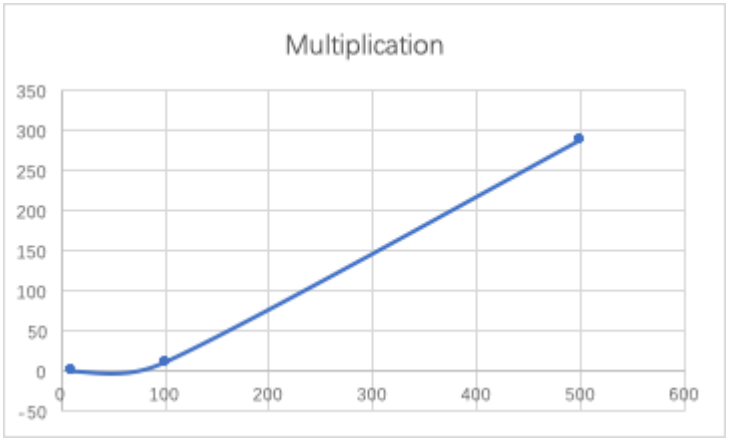
iv) Comparison

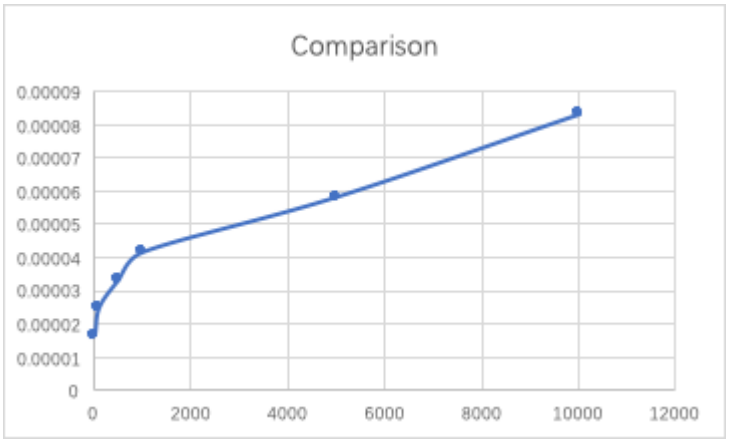
Number of Digits	My implementation	Java implementation
10	1.6666666666666667E-5	8.333333333333334E-8
100	2.5E-5	1.25E-7
500	3.3333333333333335E-5	1.6666666666666668E-7
1000	4.1666666666666665E-5	2.0833333333333333E-7
5000	5.833333333333333E-5	2.9166666666666664E-7
10000	8.333333333333333E-5	4.1666666666666667E-7

c). Plots









d). Problrms during experiment
For some operations, it took so long to execute when n got bigger and bigger. It was almost impossible to obtain the result.

Discussion of Result and Comparison

Generally, the experimental results match the theoretical expectations. the plots of the measured data have the same form as the plot of the theoretical result. For addition and subtraction, the java implementation is 100-200 times faster. For the multiplication, the java implementation is 10-100 times faster. For the comparison class, the java implementation is about 100 times faster. My multiplication class runtime may be improved if the Karatsuba algorithm is used.