Lanin gleb

DSBA 192-2

Date of submission: 28.04.2020

Supervisor name: Rudakov Kirill Alexandrovich

Different algorithms of multiplication

- Problem statement

I was given a problem of complexity of different algorithms of multiplication. To solve this problem, I wrote three different algorithms of varying complexity: grade school, divide and conquer and Karatsuba algorithm.

Theoretical information:

- Grade school multiplication algorithm – O(n^2). This is the easiest algorithm to implement, however, at the same time it is not the fastest in terms of execution time.
- Divide n conquer algorithm – O (n*log n) or O ($n \log_p n$).
  " if (a) the base cases have constant-bounded size, the work of splitting the problem and combining the partial solutions is proportional to the problem's size n, and (b) there is a bounded number p of sub-problems of size ~ n/p at each stage, then the cost of the divide-and-conquer algorithm will be O($n \log_p n$)". This algorithm is based on the recursive splitting of the task into smaller parts, which ensures less runtime.
  Source - https://en.wikipedia.org/wiki/Divide-and-conquer_algorithm
- Karatsuba algorithm – O (n^log 2 (3)).
  "Karatsuba, then a 23-year-old student, found an algorithm (later it was called "divide and conquer") that multiplies two n-digit numbers in O(n^log 2(3)) elementary steps"
  Karatsuba algorithm is quite like DnC algorithm (so the implementation of Karatsuba was like "ctrl+c, ctrl+v" after creating DnC. Nevertheless, Karatsuba has fewer components to calculate the result, which is why it is a faster algorithm than DnC. So up to a certain number of digits, both algorithms work almost the same, however, on many digits, Karatsuba will begin to overtake the DnC. The results will be presented on the graphs which I will provide later in this report.
  Source - https://en.wikipedia.org/wiki/Karatsuba_algorithm

- Research plan:

  From this research, I'm going to get data about the runtime of computing huge numbers with three algorithms and which factors affects on this.

  1. Implement Grade school algorithm and check for correct results
  2. Implement Divide and Conquer algorithm and use "grade school" for fast check
  3. Implement Karatsuba algorithm and check the correctness of results of all three algorithms
  4. Measure the running time of the algorithms and output it to the csv file
  5. Transfer these results for clarity to graphs

- Implementation details section

I created class Number with various methods that will help me in this research. It has all needed overloaded operators: [], -, -=, +, +=, ==.

My main vector "digits" is in protected field of the class.

And a few functions such as: insert (which puts supporting zeroes), pop_back, split, size and shift (which inserts zeroes in digits)

I also created class Multiplicator, where all three algorithms are presented and a function called Ran_domizer(which generates random digits and fills vector "randomed" with them.

- Column_mult
  This is my version of Grade school,
  since in fact it is a column multiplication.
  So, this algorithm uses the simplest way to multiply numbers, with two loops, nothing special. It just has shift function, and, in the end, I delete all unnecessary zeroes.

- Divide and conquer with Karatsuba

  As I already said, both these algorithms use the same idea based on recursion (Karatsuba is just an improved version of DnC), but Karatsuba must compute less elements.

  So, if G and M are two huge numbers, then the Karatsuba formula for it is:

  ```
  G * M = g1*m1*10^n + ((g1+g2) *(m1+m2) – g1*m1 – g2*m2) *10^(n/2) + g2*m2
  ```

  Where g1, g2, m1, m2 are small parts of G and M gained by recursive splitting and “n” is a size of numbers.

  Divide and conquer formula:

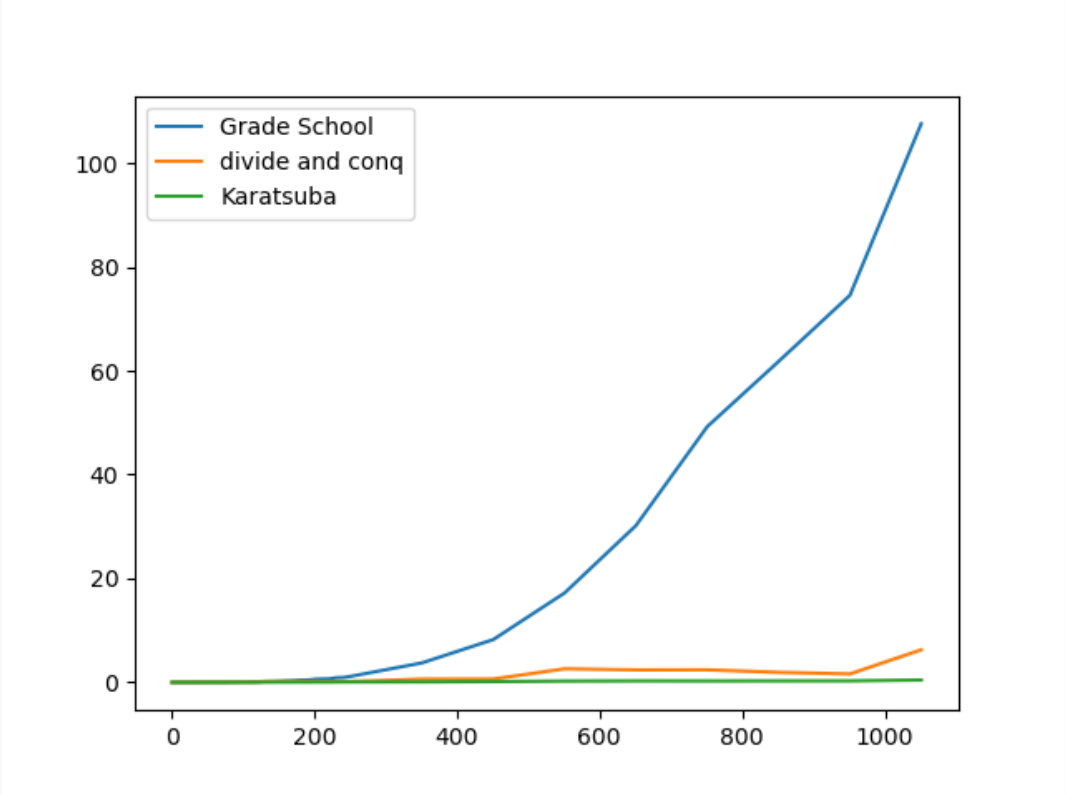
  ```
  G * M = (g1*m1) *10^n + (g1*m2 + g2*m1) *10^(n/2) + g2*m2
  ```

  After class Multiplicator I have void function “duration” which does three tests for each algorithm, measures runtimes of algorithms, find average value and outputs results into the csv file called “project results”

  Repository Address: https://github.com/Booshlya/dsba-ads2020-hw1.git

- Results:

here I enclose the results of my research expressed in graphs for clarity and convenience:



Well, I was surprised with obtained results. As I certainly expected that the algorithms of “Karatsuba” and “Divide and conquer” would be faster than the “Grade school”, but not that much. Results on 1051 digits are:

1. Grade school – 107.59 sec
2. Divide and conquer – 6.26 sec
3. Karatsuba – 0.43 sec

Nonetheless, I am pleased with my results: on small numbers the algorithms behave similarly, however after 200 characters you can notice a strong deviation of the graph of the “grade school multiplication” from the others. On approximately 900 characters the “Divide and conquer” begins to lose ground. Which makes Karatsuba the fastest method, the second place goes to the “Divide

and conquer” and in a such situation, “Grade school” algorithms is an outsider.

- Conclusion

  I managed to create three algorithms, check their results, which in spite of everything turned out to be close to my original ones results based on the final data, the conclusion is obvious : Karatsuba is the best algorithm of three (if it is necessary to multiply huge numbers).

  Ways of improvement: I think it is possible for someone to give better names for my functions, vectors, etc. But still, I think that my code is completely readable and logical