I did optional task which takes program parameter for number of threads.

First program name is program1 and run:
./program1 1000000

second program name is program2 and run as:
./program2 1000000 10

In the first part, first, I write functions for every operations. Then I called every functions that I wrote and printed them into ouput1.txt. From before the function calls to end of function calls I measure time difference and I also print this to output1.txt.

In the second part, I wrote the all functions in part 1 and put them into a function list. I take number of threads as a parameter. I wrote a runner function which take a parameter struct thread_data with l for left index and r for right index of function list. Runner function runs functions in function list from left index to right index. I create a thread list size of parameter number of threads. Then I create every thread in thread list which runs runner function with parameter thread_data. When thread operations done I print the results and runtime of functions to output2.txt.

There are differences for time of part1 and part2. Because in part1 main functions run every operations in same thread. But in part 2 tasks are distributed in threads there fore part2 runs faster than part 2. Moreover, in part 2 when thread number increase, the program which has more threads runs faster. Because number of task per threads decrease. then program runs faster.