CMPE 322 Project-2
Onur Dilsiz
2019400036

I have written 3 cpp files which are named nothread, threadTen, threadFive. The first is an implementation without threads. The other ones are implemented as 10 threads, and 5 threads.

In the first implementation, I created 10 different functions for the wanted values in the project. In order to decrease the complexities of these functions, I sorted the random number list before calling these functions. Since sorting has O(n*logn) complexity, it is the heaviest part of my program. The other functions are O(n) or O(1). Therefore, I started measuring the time after sorting the list. The endpoint for the time calculation is after the writing output. I wrote the output to "*output1.txt*".

The calculations are made for N=10000000

Average calculation: **0.25 sn** for the execution of functions **with no threads**

In 10 threads implementation, I have created 10 threads for these functions. In order to use the functions in the first implementation, I needed to change them to void * functions. Then, I calculated the time between the beginning of thread creation and the end of writing the output. I wrote the output to "*output2.tx*t" file.

Average calculation: **0.08 sn** for the execution of **10 threads**

In 5 threads implementation, I have created 5 threads for these functions. I merged 2 functions for each. While doing that I paid attention to merging an O(1) function with an O(n) function except for findMin and findMax functions. I calculated the time between the beginning of thread creation and the end of the writing output. I wrote the output to "*output3.txt*" file.

Average calculation: **0.11 sn** for the execution of **5 threads**

In conclusion, considering the calculations, we can see that using threads derives benefits for a programmer by means of speed. In this project, creating 10 threads is faster than 5 threads, which is logical since threads are executing concurrently and there are longer threads in 5 threads implementation, comparing 10 threads version.