Documentation

Implementation of Details:

I implemented my project as 3 different C++ source files.

First source file named thread.cpp for one thread, one process. This program contains a main method and many other methods in order to calculate values. I've done many of the calculations through different methods. They take input that is a vector is made of random numbers between 1000 and 10000. I used srand() to add seed to rand() function. I sorted the vector. Then I made calculations and printed to the "output1.txt".

Second source file named 10thread.cpp for 10 threads. This program splits every method to one thread. Every calculation is now a thread. The program contains of 10 calculation and 10 thread. I created a struct for every thread in order to get value from thread. Threads returns void so I need to find a way to get value from thread then I found struct solution. Structs contain numbers vector that is passed as a parameter to threads and a return value. I created 10 thread and waited of their joining. Then I printed the values that I took from threads to "output2.txt".

Third and last source file named 5thread.cpp for 5 threads. This program splits every two methods to one thread. We have 10 calculations and 5 threads now. The distribution of methods to threads is randomly. I selected random two methods and made them a thread. I created 5 structs for these 5 threads. I passed numbers vector as a parameter and took returned value thorough these structs. I waited of their joining then I printed the values that I took from threads to "output3.txt".

Differences in Duration:

I gave 1000000 as an input for my programs. In 1000000 inputs, duration of 1 thread is 0.1 seconds, duration of 5 thread is 0.068 seconds and duration of 10 thread is 0.055 seconds. Therefore, if we increase the number of threads, our program quickens. Because threads run concurrently and so, the duration of the program decreases.