Matrix Dot Product Calculator Report

1.It seems unwise to actually input data using ctrl + c so I created some randoms inside a file.

Code for random float generator:

#include <iostream>

#include <random>

#include <fstream>

using namespace std;

float get\_random()

{

static std::default\_random\_engine e;

static std::uniform\_real\_distribution<> dis(1, 10000);

return dis(e);

}

int main() {

ofstream myfile;

myfile.open("randoms.txt");

myfile<<2000000<<" ";

for (int i = 0; i <4\*1000000

; ++i) {

myfile<< get\_random()<<" ";

}

std::cout << "Hello, World!" << std::endl;

return 0;

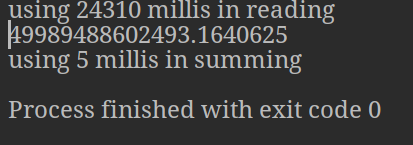
}

I decided to generate 4m random numbers instead of 400m. It seems that get random is really slow that it might take a few hours to generate such a large quantity, also It will take 3gb of my disk space which is the last thing I want to see. As for the upper boundary 10000. I think it’s better to keep the sum more manageable with a smaller numbers. If larger numbers are required please use python which is better at calculating.

2.read in files using **fstream.** I decided to store it in two vectors first but push\_back() is really slow (for eternity?) so I switched to float\* mentioned in the guide.

3. I add safety measures by read in everything as string. Checking availability and convert it later. Which adds 4000ms to the read in time.

4. Here’s the result with 2m numbers. Reading is very slow .I tried to read the file all into a single buffer then do the parsing. That saves around 60% of time. However there might not be enough memory to read them so I stick to the original version.



5.According to the result 200m numbers will take approximately 500ms to sum.