 МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ

УНИВЕРСИТЕТ

Институт №8

«Компьютерные науки и прикладная математика» Кафедра №806

«Виртуальная/дополнительная реальность

и искусственный интеллект»

Отчет по сложной задаче №1 из лабораторной работы № 2,

по учебной дисциплине

«Параллельные и распределенные вычисления»

Выполнил:

Студент 1-го курса

Гр. М80-114M-22

Гордеев Н. М.

.

(подпись, дата) .

Принял:

Семенов С. А.

.

(подпись, дата) .

Москва 2022

Оглавление

[1. Постановка задачи 2](#_Toc122003659)

[2. Метод решения 2](#_Toc122003660)

[3. Результат работы программы 2](#_Toc122003661)

[4. Вывод: 3](#_Toc122003662)

[5. Код программы 4](#_Toc122003663)

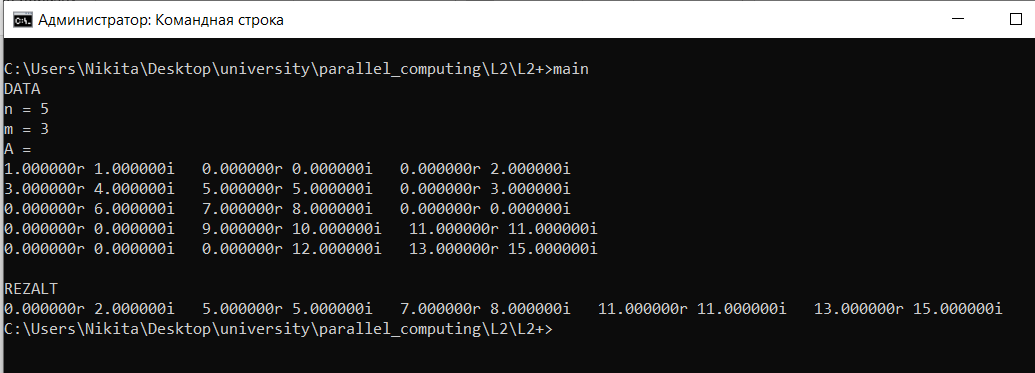
# 1. Постановка задачи

Вариант № 1. 1. Из матрицы комплексных чисел размера N х M элементов выписать массив по одному наибольшему числу из каждой строки.

# 2. Метод решения

Комплексные числа будем хранить в виде структуры из двух float, а сравнение производить по модулю ().

# 3. Результат работы программы



DATA

n = 5

m = 3

A =

1.000000r 1.000000i 0.000000r 0.000000i 0.000000r 2.000000i

3.000000r 4.000000i 5.000000r 5.000000i 0.000000r 3.000000i

0.000000r 6.000000i 7.000000r 8.000000i 0.000000r 0.000000i

0.000000r 0.000000i 9.000000r 10.000000i 11.000000r 11.000000i

0.000000r 0.000000i 0.000000r 12.000000i 13.000000r 15.000000i

REZALT

0.000000r 2.000000i 5.000000r 5.000000i 7.000000r 8.000000i 11.000000r 11.000000i 13.000000r 15.000000i

# 4. Вывод:

Познакомился с основами программирования на cuda. Задача очень простая ничего нового не освоил.

# 5. Код программы

#include "cuda\_runtime.h"      // CUDA runtime

#include "device\_launch\_parameters.h"

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

#define N 80

#define M 80

struct Complex {

    float r;

    float i;

};

// Фуккция, которая выполняется на GPU

\_\_global\_\_ void Go(Complex \*mass, Complex \*rezalt, int n, int m){

    int tid = blockIdx.x; // Получить собственный индекс

    if (tid < n){

        rezalt[tid] = mass[tid\*N];

        for(int i = 0; i < m; i++){

            if (sqrt(mass[tid\*N + i].r \* mass[tid\*N + i].r +  mass[tid\*N + i].i \* mass[tid\*N + i].i) > \

            sqrt(rezalt[tid].r \* rezalt[tid].r +  rezalt[tid].i \* rezalt[tid].i)){

                rezalt[tid] = mass[tid\*N + i];

            }

        }

    }

}

void PrintMss(Complex mass[][N],int n, int m){

    printf("n = %d\n",n);

    printf("m = %d\n",m);

    printf("A = \n");

    for(int i = 0; i < n; i++)

    {

        for(int j = 0; j<m; j++)

            printf("%fr %fi   ",mass[i][j].r,mass[i][j].i);

        printf("\n");

    }

    printf("\n");

}

int main(){

    int n;

    int m;

    FILE \*f;

    Complex mass[N][M];

    Complex rezalt[N];

    float r, i;

    size\_t pitch;

    f = fopen("data.txt","r");

    fscanf(f,"%d",&n);

    fscanf(f,"%d",&m);

    for(int j = 0; j < n; j++)

    {

        for(int k = 0; k < m; k++){

            fscanf(f,"%f",&mass[j][k].r);

            fscanf(f,"%f",&mass[j][k].i);

        }

    }

    fclose(f);

    printf("DATA\n");

    PrintMss(mass,n,m);

    Complex \*dev\_mass; // Адрес масива на GPU

    // Выделить память на GPU и сохранить её адрес в переменную

    cudaMalloc( (void\*\*)&dev\_mass, N \* M \* sizeof(Complex));

    // Копировать данные по адресу mass, размером N\*sizeof(float) в адрес dev\_mass. Копирование с устройства на GPU

    cudaMemcpy(dev\_mass, mass, N \* M \* sizeof(Complex), cudaMemcpyHostToDevice);

    Complex \*dev\_rezalt; // Адрес масива на GPU

    // Выделить память на GPU и сохранить её адрес в переменную

    cudaMalloc( (void\*\*)&dev\_rezalt, N \* sizeof(Complex));

    Go<<<n,1>>> (dev\_mass,dev\_rezalt,n,m);

    // Копировать данные по адресу dev\_mass, размером N\*sizeof(int) в адрес mass\_GPU. Копирование с GPU на устройство

    cudaMemcpy(rezalt, dev\_rezalt, N \*sizeof(Complex), cudaMemcpyDeviceToHost);

    printf("REZALT\n");

    for(int j = 0; j<n; j++)

            printf("%fr %fi   ",rezalt[j].r,rezalt[j].i);

    return 0;

}