

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук Департамент программной инженерии
Дисциплина: «Архитектура вычислительных систем»

ВЫЧИСЛЕНИЕ КОМПЛАРНОСТИ ВЕКТОРОВ

Пояснительная записка

Выполнил:
Гусейнов Ульви,
Студент группы БПИ198

Москва
2020

Содержание

1. Текст задания	2
2. Применяемые расчетные методы.....	2
2.1. Теория решения задания	2
3. Тестирование программы	3
ПРИЛОЖЕНИЕ 1	6
Список литературы.....	6
ПРИЛОЖЕНИЕ 2	7
Код программы	7-9

1. Текст задания

Найти все возможные тройки компланарных векторов. Входные данные: множество не равных между собой векторов (x, y, z) , где x, y, z – числа. Оптимальное количество потоков выбрать самостоятельно. Применяемые расчетные методы

1.1. Теория решения задания

Для нахождения и вычисления компланарности необходимо, свойство трёх векторов, которые, будучи приведёнными к общему началу, лежат в одной плоскости. Если хотя бы один из трёх векторов — нулевой, то три вектора тоже считаются компланарными. (рис 1).

Докажем, что три вектора

$\vec{a} = (1; -1; 2)$, $\vec{b} = (0; 1; -1)$ и $\vec{c} = (2; -2; 4)$ компланарны.

Как решить?

Находим смешанное произведение данных векторов:

$$\begin{aligned} \left(\vec{a}, \vec{b}, \vec{c} \right) &= \begin{vmatrix} 1 & -1 & 2 \\ 0 & 1 & -1 \\ 2 & -2 & 4 \end{vmatrix} = \\ &= 1 \times 1 \times 4 + 0 \times \begin{pmatrix} -2 \\ -1 \end{pmatrix} \times 2 + \begin{pmatrix} -1 \\ -1 \end{pmatrix} \times \begin{pmatrix} -1 \\ -1 \end{pmatrix} \times \\ &\times 2 - 2 \times 1 \times 2 - \begin{pmatrix} -2 \\ -1 \end{pmatrix} \times \begin{pmatrix} -1 \\ -1 \end{pmatrix} \times 1 - 0 \times \begin{pmatrix} -1 \\ -1 \end{pmatrix} \end{aligned}$$

Из данного примера видно, что смешанное произведение равняется нулю.

Ответ: векторы являются компланарными.

Рисунок 1. Пример вычисления компланарности.

В коде был использован параллельный поток `update_async` для ввода данных, в нем используется мьютекс `listMtx`, который уведомляет основной поток, когда данные введены и готовы к выводу с помощью доступа к переменной `bool create` между потоками.

2. Тестирование программы

При запуске программы открывается консоль, выводится информация на текстовый запрос на ввод «Векторов», для получения их количества(рис. 2).

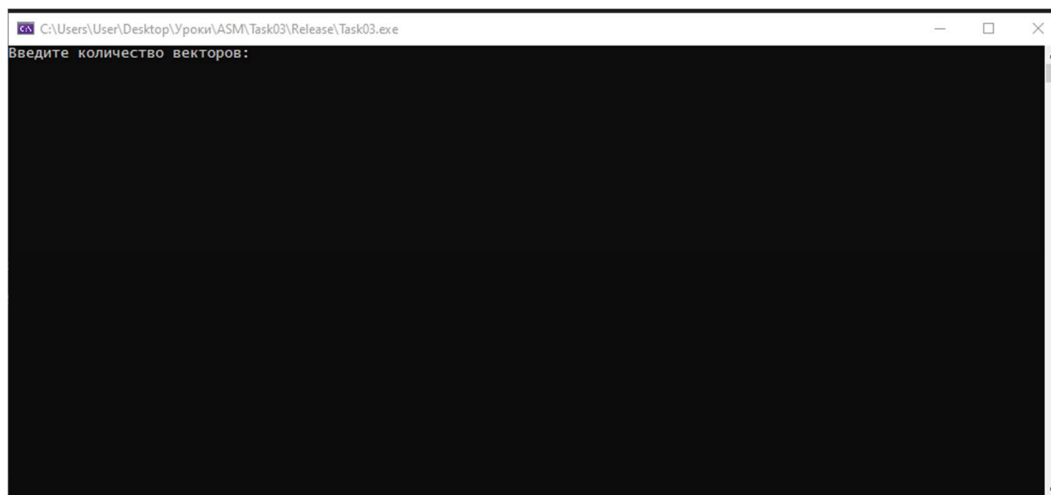


Рисунок 2. Запуск программы

Далее необходимо вписать значение «число векторов» и нажать Enter, если число меньше трех, то программа автоматически введет 3.(рис. 3).

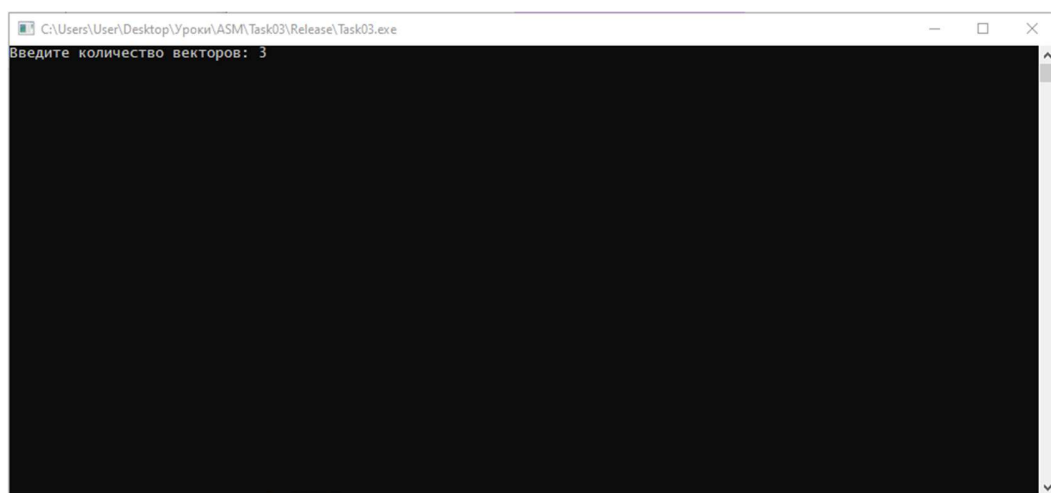


Рисунок 3. Пример работы при корректных входных данных

Далее будет запрошены (x,y,z) программы.(рис. 5).

```

C:\Users\User\Desktop\Уроки\ASMM\Task03\Release\Task03.exe
Введите количество векторов: 3
Количество введенных векторов для заполнения: 3
Введите (x, y, z):
1
2
3
Вектор номер 1 успешно записан.
Введите (x, y, z):
4
5
6
Вектор номер 2 успешно записан.
Введите (x, y, z):
7
8
9

```

Рисунок 5. Запрос x, y, z

После заполнения (x,y,z) векторов будет выведено, все возможные компланарности векторов. (рис. 5, рис. 6).

```

Консоль отладки Microsoft Visual Studio
Введите количество векторов: 3
Количество введенных векторов для заполнения: 3
Введите (x, y, z):
1 2 3
Вектор номер 1 успешно записан.
Введите (x, y, z):
4 5 6
Вектор номер 2 успешно записан.
Введите (x, y, z):
7 8 9
Вектор номер 3 успешно записан.
Комбинация Vec3(1, 2, 3) | Vec3(4, 5, 6) | Vec3(7, 8, 9) = Компланарный

```

Рисунок 5. Пример работы при входных данных

```

Консоль отладки Microsoft Visual Studio
Введите количество векторов: 6
Количество введенных векторов для заполнения: 6
Введите (x, y, z):
1
0
0
Вектор номер 1 успешно записан.
Введите (x, y, z):
0
1
0
Вектор номер 2 успешно записан.
Введите (x, y, z):
0
0
1
Вектор номер 3 успешно записан.
Введите (x, y, z):
1
1
0
Вектор номер 4 успешно записан.
Введите (x, y, z):
0
1
1
Вектор номер 5 успешно записан.
Введите (x, y, z):
1
0
1
Вектор номер 6 успешно записан.
Комбинация Vec3(1, 0, 0) Vec3(0, 1, 0) Vec3(0, 0, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 1, 0) Vec3(1, 1, 0) = Компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 1, 0) Vec3(0, 1, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 1, 0) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 0, 1) Vec3(1, 1, 0) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 0, 1) Vec3(0, 1, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 0, 1) Vec3(1, 0, 1) = Компланарный
Комбинация Vec3(1, 0, 0) Vec3(1, 1, 0) Vec3(0, 1, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(1, 1, 0) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 1, 1) Vec3(1, 1, 0) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 1, 1) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 0, 1) Vec3(1, 1, 0) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(0, 0, 1) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(1, 1, 0) Vec3(0, 1, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(1, 1, 0) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(1, 0, 1) Vec3(0, 1, 1) = Не компланарный
Комбинация Vec3(1, 0, 0) Vec3(1, 0, 1) Vec3(1, 1, 0) = Не компланарный
Комбинация Vec3(0, 1, 0) Vec3(0, 0, 1) Vec3(1, 1, 0) = Не компланарный
Комбинация Vec3(0, 1, 0) Vec3(0, 0, 1) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(0, 1, 0) Vec3(1, 1, 0) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(0, 1, 0) Vec3(1, 0, 1) Vec3(1, 1, 0) = Не компланарный
Комбинация Vec3(0, 0, 1) Vec3(1, 1, 0) Vec3(1, 0, 1) = Не компланарный
Комбинация Vec3(0, 0, 1) Vec3(1, 0, 1) Vec3(1, 1, 0) = Не компланарный
Комбинация Vec3(1, 1, 0) Vec3(1, 0, 1) Vec3(0, 1, 1) = Не компланарный

```

Рисунок 6. Пример работы при входных данных

Список литературы

1. Многопоточные классы. [Электронный ресурс] // URL: <https://habr.com/ru/post/111173/>, 2011 г.
2. Потоки, блокировки и условные переменные в C++11. [Электронный ресурс] // URL: <https://habr.com/ru/post/182610/>, 2013 г.

ПРИЛОЖЕНИЕ 2

Код программы

```
#include <iostream>
#include <vector>
#include <algorithm>
#include <sstream>
#include <future>
#include <thread>
#include <mutex>

using namespace std;
#define ILINE _inline
enum class type_zero { ZERO };

struct Vec3 {
    ILINE Vec3(type_zero) : x(0), y(0), z(0) {}
    ILINE Vec3(double x, double y, double z) : x(x), y(y), z(z) {}
    ILINE Vec3() { Vec3(0, 0, 0); }
    double x, y, z;
};

//Проверка цифра ли
bool is_numeric(string const& str) {
    auto result = double();
    auto i = istream(str);

    i >> result;

    return !i.fail() && i.eof();
}

//Проверка на компланарность
string Compl(Vec3 vecx, Vec3 vecy, Vec3 vecz)
{
    return ((vecx.x * vecy.y * vecz.z + vecy.x * vecz.y * vecx.z + vecx.y * vecy.z * vecz.x - vecx.z * vecy.y * vecz.x -
    vecz.y * vecy.z * vecx.x - vecx.y * vecy.x * vecz.z) == 0) ? "Компланарный" : "Не компланарный";
}

bool create = false;
std::mutex listMtx;

//Данные векторов
vector<Vec3> vectorsData;

int main()
{
    //Русификация консоли
    setlocale(0, "");

    std::future<void> update_async = std::async(std::launch::async, []()
    {
        //Переменные для определения X, Y, Z
        string x, y, z;
        //Переменная для определения кол-ва векторов
        string vecCount;
        //Данные векторов
```



```

while (true)
{
    //Определяем кол-во векторов
    cout << "Введите количество векторов: ";
    cin >> vecCount;
    //Проверка правильно ли ввели значение
    if (is_numeric(vecCount))
    {
        cout << "Количество введенных векторов для заполнения: " << vecCount.data() << endl;
        break;
    }
    else
        cout << "Неверный формат, введите число!" << endl;
}

if (atoi(vecCount.data()) < 3)
    vecCount = "3";

//Заполняем вектор
for (int i = 0; i < atoi(vecCount.data()); i++)
{
    cout << "Введите (x, y, z): " << endl;
    cin >> x;
    cin >> y;
    cin >> z;

    //Проверка правильности ввода
    if (is_numeric(x) && is_numeric(y) && is_numeric(z))
    {
        //заполняем массив
        vectorsData.push_back(Vec3(atoi(x.data()), atoi(y.data()), atoi(z.data())));
        cout << "Вектор номер " << i + 1 << " успешно записан." << endl;
    }
    else
    {
        //если что то не верно то идем на один шаг назад
        cout << "Неверный формат при записе в вектор, введите число!" << endl;
        i--;
    }
}

std::lock_guard<std::mutex> lock(listMtx);
create = true;

});

update_async.get();

listMtx.lock();
if (create)
{
    //Проверяем Компланарный или нет выводим
    for (int i = 0; i < vectorsData.size(); i++)
    {
        for (int j = i + 1; j < vectorsData.size(); j++)
        {
            for (int k = j + 1; k < vectorsData.size(); k++)
            {

```

```

        string combData = " Vec3(" + to_string((int)vectorsData[i].x) + ", " + to_string((int)vectorsData[i].y) + ", "
+ to_string((int)vectorsData[i].z) + ") | " +
        "Vec3(" + to_string((int)vectorsData[j].x) + ", " + to_string((int)vectorsData[j].y) + ", " +
to_string((int)vectorsData[j].z) + ") | " +
        "Vec3(" + to_string((int)vectorsData[k].x) + ", " + to_string((int)vectorsData[k].y) + ", " +
to_string((int)vectorsData[k].z) + ")";

        cout << "Комбинация " << combData << " = " << Compl(vectorsData[i], vectorsData[j], vectorsData[k])
<< endl;
    }
}
listMtx.unlock();
std::cin.get();
}
listMtx.unlock();
std::cin.get();

return 0;
}

```