**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ   
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

Домашнее задание по дисциплине «Архитектура вычислительных систем»

Ибрагимов Константин Леонидович, БПИ192

Задача

Вариант №10: 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.

Реализация

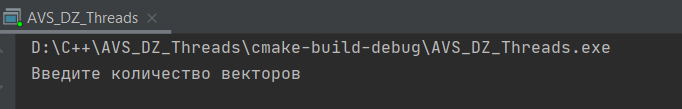


Рис.1 Начальный экран

В начале своей работы программа просит ввести количество векторов. Данное число будет проверяться на неотрицательность, неравенство нулю и так же, дабы сохранить рабоспостобность программы и ее быстродействие, на непревосхождение 10\_000.

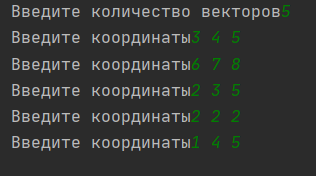
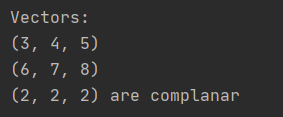


Рис.2 Ввод координат

Далее осуществляется ввод координат через пробел, где числа соответствуют координатам x, y, z соответственно.

Векторы являются компланарными тогда, когда их смешанное произведение равно 0. Для проверки этого утверждения в программе имеется метод complanar, который выводит в консоль сообщение о том, является ли тройка векторов компланарной.

По итогу пользователь получает следующее сообщение:

Рис.3 Вывод ответа

Текст программы

#include <iostream>

#include <vector>

#include <string>

#include <thread>

using namespace std;

//Метод для вывода информации о векторе

string info(vector<int> first, vector<int> second, vector<int> third) {

string res = "\nVectors:\n";

res += "(" + to\_string(first[0]) + ", " + to\_string(first[1]) + ", " + to\_string(first[2]) + ")\n";

res += "(" + to\_string(second[0]) + ", " + to\_string(second[1]) + ", " + to\_string(second[2]) + ")\n";

res += "(" + to\_string(third[0]) + ", " + to\_string(third[1]) + ", " + to\_string(third[2]) + ") are ";

return res;

}

//Метод для проверки компланарности 3 векторов

void complanar(vector<int> first, vector<int> second, vector<int> third) {

int pos = first[0] \* second[1] \* third[2] +

first[1] \* second[2] \* third[0] +

first[2] \* second[0] \* third[1];

int neg = first[2] \* second[1] \* third[0] +

first[1] \* second[0] \* third[2] +

first[0] \* second[2] \* third[1];

int res = pos - neg;

string inf = info(first, second, third);

if (res == 0)

cout << inf + "complanar\n";

}

//Метод для завпуска потоков

void threadFunc(int i, int count, vector<vector<int>> vector) {

for (int j = i + 1; j < count; ++j) {

for (int k = j + 1; k < count; ++k) {

complanar(vector[i], vector[j], vector[k]);

}

}

}

int main() {

vector<vector<int>> vect;

vector<int> temp;

string line;

int count;

cout << "Введите количество векторов";

cin >> count;

while(count <= 0 || count > 10000){

cout << "Вы ввели недопустимов количество векторов";

cin >> count;

}

for (int i = 0; i < count; ++i) {

int a, b, c;

cout << "Введите координаты";

cin >> a >> b >> c;

temp.push\_back(a);

temp.push\_back(b);

temp.push\_back(c);

vect.push\_back(temp);

temp.erase(temp.begin(), temp.end());

}

vector<thread> threads;

threads.reserve(vect.size());

for (int i = 0; i < count; ++i) {

threads.emplace\_back(threadFunc, i, count, vect);

}

for (int j = 0; j < count; ++j) {

threads[j].join();

}

return 0;

}