

**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»  
Факультет компьютерных наук  
Департамент программной инженерии**

**ПРОГРАММА, НАХОДЯЩАЯ ВСЕ ТРОЙКИ  
КОМПЛАНАРНЫХ ВЕКТОРОВ**

**Пояснительная записка**

Исполнитель  
студент группы БПИ193

\_\_\_\_\_ /С. М. Курбанова /  
« \_\_\_\_ » \_\_\_\_\_ 2020 г.

**Москва 2020**

**СОДЕРЖАНИЕ**

1.	Текст задания.....	2
2.	Применяемые расчётные методы .....	3
3.	Описание работы программы.....	4
4.	Организация входных и выходных данных.....	5
5.	Тестирование программы .....	6
6.	Список использованных источников .....	8
7.	Текст программы.....	9

**1. Текст задания**

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. Использовать OpenMP.

## 2. Применяемые расчётные методы

Для нахождения всех троек компланарных векторов [1] был использован основной критерий компланарности трёх векторов – смешанное произведение [2] компланарных векторов равно нулю.

Использованная в расчетах формула нахождения смешанного произведения:

$$(a, b, c) = a \cdot (b \times c)$$

Формула векторного произведения:

$$(b \times c) = (b_y \cdot c_z - b_z \cdot c_y) + (b_z \cdot c_x - b_x \cdot c_z) + (b_x \cdot c_y - b_y \cdot c_x)$$

Формула скалярного произведения:

$$a \cdot b = a_x \cdot b_x + a_y \cdot b_y + a_z \cdot b_z$$

### **3. Описание работы программы**

Для решения вышеописанной задачи был выбран итерационный параллелизм, так как в решении предусмотрена оптимизация вычисления смешанных произведений, что содержит в себе несколько циклов, перебирающих тройки векторов.

Задача распределена между максимальным количеством потоков, которые выполняют одну и ту же функцию: каждый поток берёт вектор и перебирает его с двумя следующими, постепенно проходя по всем парам без перестановок.

#### **4. Организация входных и выходных данных**

Ввод данных осуществляются с помощью файла, в котором координаты векторов записаны через пробел, каждый вектор находится на отдельной строке.

Ответ выводится в консоль.

## 5. Тестирование программы

Для запуска программы из командной строки необходимо ввести `coplavar_vectors_omp.exe arg1`, где `arg1` – это путь к файлу с тестом.

### 1. Тест 1. Отсутствие компланарных векторов.

```
C:\Users\Sabiyat\Documents\GitHub\coplanar-vectors-omp>coplavar_vectors_omp.exe /input/test1.txt
There are no coplanar vectors.
```

```
C:\Users\Sabiyat\Documents\GitHub\coplanar-vectors-omp>
```

test1 – Блокнот

Файл Правка Формат Вид Справка  
9 9 9

### 2. Тест 2. Поиск всех троек компланарных векторов.

```
C:\Users\Sabiyat\Documents\GitHub\coplanar-vectors-omp>coplavar_vectors_omp.exe tests/test2.txt
```

```
8 threads are used
thread 0: (1, 2, 3), (3, 4, 5), (6, 7, 8)
thread 0: (1, 2, 3), (3, 4, 5), (6, 8, 10)
thread 0: (1, 2, 3), (3, 4, 5), (3, 3.5, 4)
thread 4: (6, 8, 10), (3, 3.5, 4), (1.5, 1.75, 2)
thread 1: (3, 4, 5), (6, 7, 8), (6, 8, 10)
thread 1: (3, 4, 5), (6, 7, 8), (3, 3.5, 4)
thread 1: (3, 4, 5), (6, 7, 8), (1.5, 1.75, 2)
thread 1: (3, 4, 5), (2, 4, 9), (6, 8, 10)
thread 1: (3, 4, 5), (6, 8, 10), (3, 3.5, 4)
thread 1: (3, 4, 5), (6, 8, 10), (1.5, 1.75, 2)
thread 1: (3, 4, 5), (3, 3.5, 4), (1.5, 1.75, 2)
thread 3: (2, 4, 9), (3, 3.5, 4), (1.5, 1.75, 2)
thread 0: (1, 2, 3), (3, 4, 5), (1.5, 1.75, 2)
thread 2: (6, 7, 8), (2, 4, 9), (3, 3.5, 4)
thread 2: (6, 7, 8), (2, 4, 9), (1.5, 1.75, 2)
thread 2: (6, 7, 8), (6, 8, 10), (3, 3.5, 4)
thread 2: (6, 7, 8), (6, 8, 10), (1.5, 1.75, 2)
thread 2: (6, 7, 8), (3, 3.5, 4), (1.5, 1.75, 2)
thread 0: (1, 2, 3), (6, 7, 8), (6, 8, 10)
thread 0: (1, 2, 3), (6, 7, 8), (3, 3.5, 4)
thread 0: (1, 2, 3), (6, 7, 8), (1.5, 1.75, 2)
thread 0: (1, 2, 3), (6, 8, 10), (3, 3.5, 4)
thread 0: (1, 2, 3), (6, 8, 10), (1.5, 1.75, 2)
thread 0: (1, 2, 3), (3, 3.5, 4), (1.5, 1.75, 2)
```

test2 – Блокнот

Файл Правка Формат Вид Справка

```
1 2 3
3 4 5
6 7 8
2 4 9
6 8 10
3 3.5 4
1.5 1.75 2
12 16 20
```

### 3. Тест 3. Поиск всех троек компланарных векторов.

```
C:\Users\Sabiyat\Documents\GitHub\coplanar-vectors-omp>coplavar_vectors_omp.exe tests/test3.txt
```

```
8 threads are used
thread 0: (1, -1, 2), (0, 1, -1), (2, -2, 4)
thread 0: (1, -1, 2), (0, 1, -1), (3, -4, 7)
thread 0: (1, -1, 2), (2, -2, 4), (1, 2, 3)
thread 0: (1, -1, 2), (2, -2, 4), (1, 1, 1)
thread 0: (1, -1, 2), (2, -2, 4), (1, 2, 1)
thread 0: (1, -1, 2), (2, -2, 4), (2, -1, 2)
thread 0: (1, -1, 2), (2, -2, 4), (1, 2, -3)
thread 0: (1, -1, 2), (2, -2, 4), (3, -4, 7)
thread 0: (1, -1, 2), (2, -2, 4), (2, -2, 3)
thread 0: (1, -1, 2), (2, -2, 4), (4, 0, 6)
thread 3: (2, -1, 2), (1, 2, -3), (3, -4, 7)
thread 2: (1, 1, 1), (1, 2, 1), (2, -1, 2)
thread 1: (2, -2, 4), (1, 1, 1), (4, 0, 6)
thread 0: (1, -1, 2), (1, 1, 1), (4, 0, 6)
thread 1: (1, 2, 3), (1, 2, 1), (1, 2, -3)
thread 0: (0, 1, -1), (2, -2, 4), (3, -4, 7)
thread 0: (0, 1, -1), (2, -1, 2), (2, -2, 3)
```

test3 – Блокнот

Файл Правка Формат Вид Справка

```
1 -1 2
0 1 -1
2 -2 4
1 2 3
1 1 1
1 2 1
2 -1 2
1 2 -3
3 -4 7
2 -2 3
4 0 6
-7 -7 7
```

### 4. Тест 4. Ввод неверного количества аргументов.

```
C:\Users\Sabiyat\Documents\GitHub\coplanar-vectors-omp>coplavar_vectors_omp.exe
Wrong number of args.
```

## 5. Тест 5. Неверные значения в файле.

```
C:\Users\Sabiyat\Documents\GitHub\coplanar-vectors-openmp>coplanar_vectors_omp.exe tests/test3.txt
Wrong data in file.
```

test3 – Блокнот  
 Файл Правка Формат Вид Справка  
 1 -1 f  
 0 1 f  
 2 -2 4

## 6. Тест 6. Поиск всех троек компланарных векторов.

```
C:\Users\Sabiyat\Documents\GitHub\coplanar-vectors-openmp>coplanar_vectors_omp.exe tests/test4.txt
8 threads are used
thread 0: (-65, -179, -541), (229, 179, 664), (169, 83, 355)
thread 7: (101, -25, 0), (-18, 233, 0), (-160, -653, 0)
thread 7: (101, -25, 0), (-18, 233, 0), (156, -192, 0)
thread 7: (101, -25, 0), (-160, -653, 0), (156, -192, 0)
thread 7: (-18, 233, 0), (-160, -653, 0), (156, -192, 0)
thread 3: (-139, 80, 301), (-333, -348, 432), (58, -8, -112)
thread 3: (-332, -45, -47), (378, 38, -6), (138, -21, -159)
thread 3: (-603, -165, -219), (-378, -610, 116), (782, 0, 391)
thread 1: (82, -54, -628), (386, -358, -540), (310, -282, -562)
thread 4: (-275, 352, 25), (-467, 516, -11), (545, 120, 485)
thread 3: (448, 698, -823), (-380, -72, -82), (782, 0, 391)
thread 0: (-164, -655, 327), (0, 78, -78), (-212, -348, -76)
thread 2: (700, 596, -171), (0, 78, -78), (-168, -49, -53)
thread 2: (464, -206, 309), (115, -448, 270), (-271, -12, -116)
thread 2: (56, -78, -342), (0, 78, -78), (-40, -372, 672)
thread 3: (108, 125, 12), (-319, -393, 297), (211, 268, -309)
thread 0: (-325, -85, -275), (-240, -85, -275), (505, -153, -495)
thread 0: (-325, -85, -275), (-689, 425, 218), (782, 0, 391)
thread 2: (-30, -291, -378), (704, 287, 148), (-113, -53, -33)
thread 2: (144, 164, 154), (636, -178, 229), (782, 0, 391)
thread 1: (103, 378, 58), (-179, -504, 106), (-226, -126, 824)
thread 3: (95, 55, -490), (493, 225, -462), (-35, -3, -414)
thread 2: (149, -324, 279), (-287, -419, -150), (-102, -454, 60)
thread 0: (296, 5, -103), (-407, 363, -174), (-172, 277, -179)
thread 2: (477, -94, -822), (-97, 41, 270), (673, 118, 18)
thread 4: (67, -444, -240), (1, -219, 138), (-141, 636, 704)
thread 1: (75, -227, -148), (0, 78, -78), (-120, 703, -103)
thread 0: (-324, 724, 162), (522, 35, -261), (-30, -259, 15)
thread 0: (-324, 724, 162), (522, 35, -261), (116, -248, -58)
thread 0: (-324, 724, 162), (95, -550, -165), (-831, -372, -360)
thread 0: (-324, 724, 162), (-30, -259, 15), (116, -248, -58)
thread 4: (-631, 950, 797), (-512, 432, -152), (-394, 598, 509)
thread 3: (84, 215, 29), (240, 546, 84), (8, -173, 6)
thread 3: (-96, 40, 60), (-14, -120, -331), (782, 0, 391)
thread 3: (0, 78, -78), (-374, -161, 571), (-187, -137, 342)
thread 3: (0, 78, -78), (391, 414, -483), (-697, -625, 748)
thread 3: (0, 78, -78), (672, 27, 309), (782, 0, 391)
thread 3: (0, 78, -78), (-32, 463, 769), (12, -481, 19)
thread 3: (0, 78, -78), (81, -97, 19), (243, 563, -797)
thread 1: (117, -225, -324), (-72, 443, 262), (69, 106, -142)
thread 0: (256, -140, -4), (-12, -9, -835), (12, -15, -453)
thread 2: (-52, 48, 676), (-122, 8, 56), (782, 0, 391)
thread 0: (73, 311, 90), (0, 78, -78), (-146, -248, -554)
thread 3: (-150, 519, 371), (20, -181, 526), (48, -228, 200)
thread 0: (-135, -32, -692), (-686, -68, 126), (-169, -36, -710)
thread 2: (-688, -327, -17), (0, 78, -78), (672, 27, 309)
thread 2: (-688, -327, -17), (0, 78, -78), (782, 0, 391)
thread 2: (-688, -327, -17), (672, 27, 309), (782, 0, 391)
thread 0: (-324, -297, -411), (-3, -253, 167), (-270, -783, 23)
thread 2: (522, 35, -261), (-30, -259, 15), (116, -248, -58)
thread 3: (149, -9, -237), (-20, -521, 144), (173, 538, -393)
thread 1: (853, -10, 244), (782, 0, 391), (-265, -2, -169)
thread 3: (157, 10, 112), (5, 207, -91), (476, 237, 245)
thread 1: (-343, 316, 249), (67, -444, -240), (-266, -728, -294)
thread 0: (-224, 112, -28), (-872, 270, 57), (0, 78, -78)
thread 0: (-224, 112, -28), (-168, -49, -53), (312, -289, 7)
thread 0: (-224, 112, -28), (-258, 283, -35), (-308, 294, -41)
```

test4 – Блокнот  
 Файл Правка Формат Вид Справка  
 -65 -179 -541  
 -135 -127 338  
 -188 -626 -670  
 671 -583 472  
 -77 -65 99  
 256 -406 -381  
 930 -590 -584  
 577 -162 -61  
 33 -378 52  
 -73 443 195  
 -164 -655 327  
 -161 97 629  
 -419 454 -291  
 357 328 806  
 -157 55 179  
 -695 437 740  
 -219 333 428  
 -325 -85 -275  
 -433 -58 271  
 862 221 -432  
 178 527 320  
 261 -647 508  
 480 230 -511  
 -359 224 7  
 28 -969 -643  
 167 -401 428  
 -293 88 -336  
 -395 130 151  
 -692 -817 -850  
 153 258 100  
 296 5 -103  
 -57 -678 388  
 310 -744 493  
 411 523 592  
 -115 -159 -199  
 -94 446 -707  
 358 604 457  
 199 -621 -487  
 392 -412 575  
 123 770 401  
 -324 724 162  
 -503 -241 -496  
 -98 -830 311  
 212 -169 558  
 761 146 320

Стр 2000, сл6 12



## 6. Список использованных источников

- 1) Компланарные векторы // [Электронный ресурс] Режим доступа: <https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%BB%D0%B0%D0%BD%D0%B0%D1%80%D0%BD%D0%BE%D1%81%D1%82%D1%8C>, свободный;
- 2) Смешанное произведение // [Электронный ресурс] Режим доступа: <https://ru.wikipedia.org/wiki/%D0%A1%D0%BC%D0%B5%D1%88%D0%B0%D0%BD%D0%BD%D0%BE%D0%B5%D0%BF%D1%80%D0%BE%D0%B8%D0%B7%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5>, свободный;
- 3) Примеры программ // [Электронный ресурс] Режим доступа: <http://softcraft.ru/edu/comparch/practice/thread/03-openmp/>, свободный;
- 4) Оформление задания // [Электронный ресурс] Режим доступа: <http://softcraft.ru/edu/comparch/tasks/t04/>, свободный;
- 5) Директивы OpenMP // [Электронный ресурс] Режим доступа: <https://docs.microsoft.com/ru-ru/cpp/parallel/openmp/reference/openmp-directives?view=msvc-160#for-openmp>, свободный;
- 6) Параллельное программирование на OpenMP // [Электронный ресурс] Режим доступа: <http://ccfit.nsu.ru/arom/data/openmp.pdf>, свободный.

## 7. Текст программы

```

#include <iostream>
#include <vector>
#include <fstream>
#include "omp.h"

struct Vector
{
public:
    double X;
    double Y;
    double Z;

    Vector()
    {
        X = 0;
        Y = 0;
        Z = 0;
    }

    Vector(double x, double y, double z)
    {
        X = x;
        Y = y;
        Z = z;
    }

    /// <summary>
    /// Checks if this vector and two other are coplanar by
    /// checking if scalar triple product is equal to 0
    /// </summary>
    /// <param name="v2">second vector</param>
    /// <param name="v3">third vector</param>
    /// <returns>if vectors are coplanar</returns>
    bool isCoplanar(Vector v2, Vector v3)
    {
        double p[3] = { v2.Y * v3.Z - v2.Z * v3.Y,
                        v2.Z * v3.X - v2.X * v3.Z,
                        v2.X * v3.Y - v2.Y * v3.X };

        return X * p[0] + Y * p[1] + Z * p[2] == 0;
    }
};

/// <summary>
/// Reads vectors from file
/// </summary>
/// <param name="path">path to input file</param>
/// <param name="vectors">vector of Vectors</param>
/// <returns>if reading was successful</returns>
bool getVectorsFromFile(std::string path, std::vector<Vector>& vectors)
{
    std::fstream in(path, std::ios::in);
    double x;
    double y;
    double z;

    if (in.is_open())
    {
        in >> x >> y >> z;

        while (!in.eof())
        {
            if (in.fail() && !in.eof())

```

```

        return false;

        Vector currentVector(x, y, z);
        vectors.push_back(currentVector);

        in >> x >> y >> z;
    }

    in.close();
    return true;
}
return false;
}

/// <summary>
/// Finds all triplets of coplanar vectors (thread function)
/// </summary>
/// <param name="vectors">input vectors</param>
/// <param name="k">vector's index</param>
void findTriplets(std::vector<Vector> vectors, int k)
{
    for (int i = k + 1; i < vectors.size(); i++)
    {
        for (int j = i + 1; j < vectors.size(); j++)
        {
            if (vectors[k].isCoplanar(vectors[i], vectors[j]))
            {
                printf("thread %d: (%g, %g, %g), (%g, %g, %g), (%g, %g, %g)\n",
                    omp_get_thread_num(),
                    vectors[k].X, vectors[k].Y, vectors[k].Z,
                    vectors[i].X, vectors[i].Y, vectors[i].Z,
                    vectors[j].X, vectors[j].Y, vectors[j].Z);
            }
        }
    }
}

int main(int argc, char** argv)
{
    std::vector<Vector> vectors;

    if (argc != 2)
    {
        std::cout << "Wrong number of args." << std::endl;
        return -1;
    }

    if (!getVectorsFromFile(argv[1], vectors))
    {
        std::cout << "Wrong data in file." << std::endl;
        return -1;
    }

    if (vectors.size() < 3)
    {
        std::cout << "There are no coplanar vectors." << std::endl;
        return 0;
    }

    std::cout << omp_get_max_threads() << " threads are used" << std::endl;
    omp_set_num_threads(omp_get_max_threads());

#pragma omp parallel for

```

```
for (int i = 0; i < vectors.size(); i++)  
    findTriplets(vectors, i);  
}
```