

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

Факультет компьютерных наук  
Департамент программной инженерии

**КОНСОЛЬНОЕ ПРИЛОЖЕНИЕ, НАХОДЯЩЕЕ ТРОЙКИ  
КОМПЛАНАРНЫХ ВЕКТОРОВ СРЕДИ ЗАДАННЫХ С  
ИСПОЛЬЗОВАНИЕМ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ С  
ПРИМЕНЕНИЕМ OPENMP**

Пояснительная записка

**Исполнитель:**

Студентка группы БПИ195

\_\_\_\_\_/Зубарева Н.Д./

«17» ноября 2020 г.

## Оглавление

1. Текст задания.....	2
2. Применяемые расчетные методы .....	3
2.1. Теория решения задания .....	3
2.2. Организация многопоточности.....	3
2.3. Ввод входных данных .....	3
2.4. Вывод данных.....	3
3. Тестирование программы.....	4
3.1. Корректные значения .....	4
3.2. Некорректные значения .....	5
4. Список литературы.....	7
5. Приложение кода.....	8

## 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 расчетные методы

### 2.1. Теория решения задания

По условию требуется находить компланарные тройки векторов среди данных. Согласно [2], для этого можно использовать значение смешанного произведения векторов, а именно, оно должно быть равно нулю. Также использована формула вычисления смешанного произведения по координатам трех данных векторов [3]:

$$(\bar{a}, \bar{b}, \bar{c}) = \begin{vmatrix} a_x & a_y & a_z \\ b_x & b_y & b_z \\ c_x & c_y & c_z \end{vmatrix}.$$

### 2.2. Организация многопоточности

Программа реализует модель итеративного параллелизма [4]: для каждого вектора создается поток, который далее перебирает все вторые и третьи элементы тройки из векторов с индексами после предыдущего в тройке и проверяет для полученной тройки равенство смешанного произведения нулю. Потоки, таким образом, фиксированы по индексам векторов и не вступают в конфликт, благодаря чему можно избежать использование блокировок и семафоров, вывод каждой тройки векторов в консоль также потокобезопасен. Программа написана на языке Си и использует OpenMP для создания потоков с помощью директивы `#pragma omp parallel for`, которая выделяет оптимальное количество потоков для реализации цикла, в котором вызывается функция, выполняемая одним потоком.

### 2.3. Ввод входных данных

Ввод данных осуществлен через командную строку и чтение из файла. В командной строке задается путь к файлу, из которого нужно считать данные. Далее в файле должно быть указано число векторов. Было принято решение ограничить количество 3 векторами снизу (1 и 2 вектора всегда компланарны) и 50 векторами сверху (при большем количестве работа программы в среднем занимает больше 10 секунд). При нехватке векторов программа завершается, при избытке – считывание не осуществляется после 50 векторов. Далее в файле должны быть записаны векторы в указанном выше количестве. Считывание всех чисел осуществляется с помощью функции `fscanf` в формате `%ld` для числа векторов и `%lf` для элементов векторов.

### 2.4. Вывод данных

Тройки компланарных векторов выводятся в консоль с помощью функции `printf` с форматированием `%g`.

### 3. Тестирование программы

Программа компилируется и запускается следующим образом из командной строки (рисунок 1).

```
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ gcc -x c Vector.cpp -fopenmp -o vec.exe
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input0.txt
```

Рисунок 1 Команды компиляции и запуска программы

#### 3.1. Корректные значения

Программа осуществляет перебор троек векторов для нахождения компланарных. При некомпланарности тройки выводится сообщение без указания номера тройки, при компланарности – векторы, входящие в нее (рисунок 2, рисунок 3).

```
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input0.txt
reading...
your vectors:
{1, 2, 3}
{2, 2, 2}
{3, 6, 9}
{0, 0, 1}
{1, 0, 0}
{0, 1, 0}

not coplanar
not coplanar
not coplanar
not coplanar
coplanar are {1, 2, 3}, {2, 2, 2}, {3, 6, 9}
not coplanar
not coplanar
not coplanar
coplanar are {1, 2, 3}, {3, 6, 9}, {0, 0, 1}
coplanar are {1, 2, 3}, {3, 6, 9}, {1, 0, 0}
coplanar are {1, 2, 3}, {3, 6, 9}, {0, 1, 0}
not coplanar
not coplanar
not coplanar
not coplanar
not coplanar
not coplanar
not coplanar
not coplanar
the end
```

```
input0.txt – Блокнот
Файл  Правка  Формат  Вид  Справка
6
1 2 3
2 2 2
3 6 9
0 0 1
1 0 0
0 1 0
```

Рисунок 2 Работа программы при корректных данных

```
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input1.txt
reading...
your vectors:
{1, 0, 1}
{0, 1, 0}
{1, 1, 1}
{1, 2, 3}

not coplanar
coplanar are {1, 0, 1}, {0, 1, 0}, {1, 1, 1}
not coplanar
not coplanar
the end
```

```
input1.txt – Блокнот
Файл  Правка  Формат  Вид  Справка
4
1 0 1
0 1 0
1 1 1
1 2 3
```

Рисунок 3 Работа программы при корректных данных

### 3.2. Некорректные значения

Программа также обрабатывает случаи ввода некорректных данных, например когда число векторов меньше указанного в файле числа (в этом случае считывается максимально возможное число векторов) (рисунок 4).

```
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input2.txt
reading...
something in the file is wrong
your vectors:
{0, 0, 1}
{0, 1, 0}
{1, 0, 0}

not coplanar
the end
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$
```

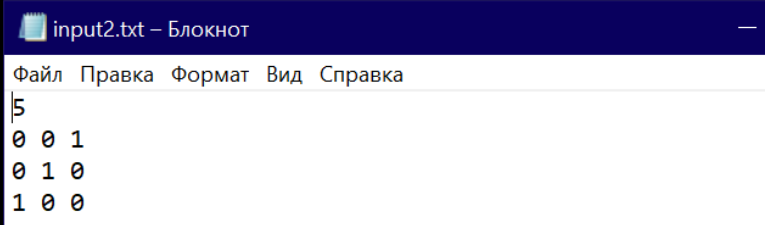


Рисунок 4 Работа программы при количестве векторов меньше указанного

Если указанное число не считывается верно, выводится сообщение об ошибке, и работа программы завершается (рисунок 5).

```
{0, 1, 0}
{1, 0, 0}

not coplanar
the end
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input3.txt
reading...
wrong number in the file
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$
```

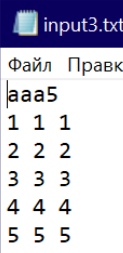


Рисунок 5 Работа программы при некорректном количестве векторов

Если какой-то из элементов векторов задан не числом, работа осуществляется со считанными до этого векторами (рисунок 6).

```
wrong number in the file
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input4.txt
reading...
something in the file is wrong
your vectors:
{1, 1, 1}
{2, 2, 2}

the end
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$
```

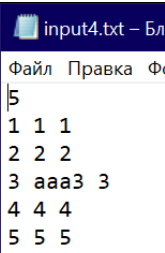


Рисунок 6 Работа программы при некорректном элементе вектора

В случае, когда векторов меньше трех, выводится сообщение о нехватке векторов и работа программы завершается (рисунок 7).

```
the end
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input5.txt
reading...
it's not interesting to check for coplanarity less than 3 vectors
nat@LAPTOP-1AGP7LH0:/mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$
```

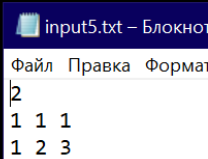


Рисунок 7 Сообщение об ошибке и завершение работы при недостаточном количестве векторов

Если векторов задано слишком много (больше 50), выводится сообщение об избытке и работа осуществляется с 50 векторами (рисунок 8, рисунок 9).

```

Выбрать nat@LAPTOP-1AGP7LH0: /mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP
nat@LAPTOP-1AGP7LH0: /mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP$ ./vec.exe input6.txt
reading...
that's just too much... believe me, 50 will do just fine
your vectors:
{1, 1, 1}
{2, 2, 2}
{3, 3, 3}
{4, 4, 4}
{5, 5, 5}
{6, 6, 6}
{7, 7, 7}
{8, 8, 8}
{9, 9, 9}
{10, 10, 10}
{11, 11, 11}
{12, 12, 12}
{13, 13, 13}
{14, 14, 14}
{15, 15, 15}
{16, 16, 16}
{17, 17, 17}
{18, 18, 18}
{19, 19, 19}
{20, 20, 20}
{21, 21, 21}
{22, 22, 22}
{23, 23, 23}
{24, 24, 24}
{25, 25, 25}
{26, 26, 26}
{27, 27, 27}
{28, 28, 28}
{29, 29, 29}
{30, 30, 30}
{31, 31, 31}
{32, 32, 32}
{33, 33, 33}
{34, 34, 34}
{35, 35, 35}
{36, 36, 36}

input6.txt - Блокнот
Файл Правка Формат Вид Справка
51
1 1 1
2 2 2
3 3 3
4 4 4
5 5 5
6 6 6
7 7 7
8 8 8
9 9 9
10 10 10
11 11 11
12 12 12
13 13 13
14 14 14
15 15 15
16 16 16
17 17 17
18 18 18
19 19 19
20 20 20
21 21 21
22 22 22
23 23 23
24 24 24
25 25 25
26 26 26
27 27 27
28 28 28
29 29 29
30 30 30
31 31 31
32 32 32
33 33 33
34 34 34
35 35 35

```

Рисунок 8 Работа программы при числе векторов больше 50 – обрезка количества

```

Выбрать nat@LAPTOP-1AGP7LH0: /mnt/c/Users/Natalya/Desktop/VectorsOMP/VectorsOMP
{38, 38, 38}
{39, 39, 39}
{40, 40, 40}
{41, 41, 41}
{42, 42, 42}
{43, 43, 43}
{44, 44, 44}
{45, 45, 45}
{46, 46, 46}
{47, 47, 47}
{48, 48, 48}
{49, 49, 49}
{50, 50, 50}

coplanar are {1, 1, 1}, {2, 2, 2}, {3, 3, 3}
coplanar are {1, 1, 1}, {2, 2, 2}, {4, 4, 4}
coplanar are {1, 1, 1}, {2, 2, 2}, {5, 5, 5}
coplanar are {1, 1, 1}, {2, 2, 2}, {6, 6, 6}
coplanar are {1, 1, 1}, {2, 2, 2}, {7, 7, 7}
coplanar are {1, 1, 1}, {2, 2, 2}, {8, 8, 8}
coplanar are {1, 1, 1}, {2, 2, 2}, {9, 9, 9}
coplanar are {1, 1, 1}, {2, 2, 2}, {10, 10, 10}
coplanar are {1, 1, 1}, {2, 2, 2}, {11, 11, 11}
coplanar are {1, 1, 1}, {2, 2, 2}, {12, 12, 12}
coplanar are {1, 1, 1}, {2, 2, 2}, {13, 13, 13}
coplanar are {1, 1, 1}, {2, 2, 2}, {14, 14, 14}
coplanar are {1, 1, 1}, {2, 2, 2}, {15, 15, 15}
coplanar are {1, 1, 1}, {2, 2, 2}, {16, 16, 16}
coplanar are {1, 1, 1}, {2, 2, 2}, {17, 17, 17}
coplanar are {1, 1, 1}, {2, 2, 2}, {18, 18, 18}
coplanar are {1, 1, 1}, {2, 2, 2}, {19, 19, 19}
coplanar are {1, 1, 1}, {2, 2, 2}, {20, 20, 20}
coplanar are {1, 1, 1}, {2, 2, 2}, {21, 21, 21}
coplanar are {1, 1, 1}, {2, 2, 2}, {22, 22, 22}
coplanar are {1, 1, 1}, {2, 2, 2}, {23, 23, 23}
coplanar are {1, 1, 1}, {2, 2, 2}, {24, 24, 24}
coplanar are {1, 1, 1}, {2, 2, 2}, {25, 25, 25}
coplanar are {1, 1, 1}, {2, 2, 2}, {26, 26, 26}
coplanar are {1, 1, 1}, {2, 2, 2}, {27, 27, 27}
coplanar are {1, 1, 1}, {2, 2, 2}, {28, 28, 28}

input6.txt - Блокнот
Файл Правка Формат Вид Справка
16 16 16
17 17 17
18 18 18
19 19 19
20 20 20
21 21 21
22 22 22
23 23 23
24 24 24
25 25 25
26 26 26
27 27 27
28 28 28
29 29 29
30 30 30
31 31 31
32 32 32
33 33 33
34 34 34
35 35 35
36 36 36
37 37 37
38 38 38
39 39 39
40 40 40
41 41 41
42 42 42
43 43 43
44 44 44
45 45 45
46 46 46
47 47 47
48 48 48
49 49 49
50 50 50
51 51 51

```

Рисунок 9 Работа программы при числе векторов больше 50

## 4. Список литературы

- [1] Инструкция по составлению пояснительной записки [Электронный ресурс]. //URL: <http://softcraft.ru/edu/comparch/tasks/mp01/> (Дата обращения: 30.10.2020, режим доступа: свободный)
- [2] Статья «Coplanarity» Wikipedia.org //URL: <https://en.wikipedia.org/wiki/Coplanarity> (Дата обращения: 15.11.2020, режим доступа: свободный)
- [3] Статья «Triple product» Wikipedia.org //URL: [https://en.wikipedia.org/wiki/Triple\\_product#Scalar\\_triple\\_product](https://en.wikipedia.org/wiki/Triple_product#Scalar_triple_product) (Дата обращения: 15.11.2020, режим доступа: свободный)
- [4] Практические приемы построения многопоточных приложений [Электронный ресурс]. //URL: <http://softcraft.ru/edu/comparch/tasks/t03/> (Дата обращения: 15.11.2020, режим доступа: свободный)



## 5. Приложение кода

```
6. #include <omp.h>
7. #include <stdlib.h>
8. #include <stdbool.h>
9. #include <stdio.h>
10.
11. /// <summary>
12. /// Структура для вектора, состоящая из трех координат.
13. /// </summary>
14. typedef struct Vector {
15.     double x, y, z;
16. } Vector;
17.
18. /// <summary>
19. /// Глобальные переменные для массива считанных векторов и их количества.
20. /// </summary>
21. Vector* vectors;
22. size_t numberOfVectors;
23.
24. /// <summary>
25. /// Метод для чтения из файла, в котором должно быть записано
26. /// число векторов, а далее векторы по координатам.
27. /// В случае неверного формата происходит обработка ошибок.
28. /// </summary>
29. void read(char* filename) {
30.     FILE* file = fopen(filename, "r");
31.     if (!file) {
32.         perror(filename);
33.         exit(1);
34.     }
35.
36.     if (fscanf(file, "%ld", &numberOfVectors) != 1) {
37.         printf("wrong number in the file\n");
38.         fclose(file);
39.         exit(1);
40.     }
41.
42.     /// Проверка на то, что количество векторов не меньше 3.
43.     /// Если меньше, то искать компланарные тройки бесполезно.
44.     /// Завершаем программу.
45.     if (numberOfVectors < 3)
46.     {
47.         printf("it's not interesting to check for coplanarity less than 3
vectors\n");
48.         fclose(file);
49.         exit(0);
50.     }
51.
52.     /// Проверка на то, что количество векторов не превышает 50.
53.     /// Если превышает, мы все же не будем считать больше 50 векторов.
54.     if (numberOfVectors > 50)
55.     {
56.         printf("that's just too much... believe me, 50 will do just fine\n");
57.         numberOfVectors = 50;
58.     }
59.
60.     /// Выделяем память для векторов в глобальной переменной и считываем.
61.     vectors = (Vector*)calloc(numberOfVectors, sizeof(Vector));
62.     for (size_t i = 0; i < numberOfVectors; i++) {
63.
64.         if (fscanf(file, "%lf", &vectors[i].x) != 1) {
65.             numberOfVectors = i;
66.             printf("something in the file is wrong\n");
67.             break;
```

```

68.         }
69.         if (fscanf(file, "%lf", &vectors[i].y) != 1) {
70.             numberOfVectors = i;
71.             printf("something in the file is wrong\n");
72.             break;
73.         }
74.         if (fscanf(file, "%lf", &vectors[i].z) != 1) {
75.             numberOfVectors = i;
76.             printf("something in the file is wrong\n");
77.             break;
78.         }
79.     }
80.
81.     fclose(file);
82. }
83.
84. /// <summary>
85. /// Метод для печати считанных векторов в консоль.
86. /// </summary>
87. void printVectors(Vector* vectors) {
88.     printf("your vectors:\n");
89.
90.     for (size_t i = 0; i < numberOfVectors; i++)
91.     {
92.         printf("{%g, %g, %g}\n", vectors[i].x, vectors[i].y, vectors[i].z);
93.     }
94.
95.     printf("\n");
96. }
97.
98. /// <summary>
99. /// Метод для проверки, является ли тройка векторов компланарной через
100. /// равенство смешанного произведения нулю.
101. /// </summary>
102. bool coplanar(Vector a, Vector b, Vector c) {
103.     int value = (a.x * b.y * c.z) + (a.y * b.z * c.x) + (a.z * b.x * c.y) -
104.         (a.z * b.y * c.x) - (a.x * b.z * c.y) - (b.x * a.y * c.z);
105.     return value == 0;
106. }
107.
108. /// <summary>
109. /// Функция, выполняемая потоком. Каждый поток прикреплен к
110. /// первому вектору в тройке и внутри него происходит подбор
111. /// второго и третьего векторов, проверка их на компланарность
112. /// и вывод результата.
113. /// </summary>
114. void threadFunction(size_t i) {
115.
116.     for (size_t j = i + 1; j < numberOfVectors; ++j) {
117.         for (size_t k = j + 1; k < numberOfVectors; ++k) {
118.             if (coplanar(vectors[i], vectors[j], vectors[k])) {
119.                 printf("coplanar are {%g, %g, %g}, {%g, %g, %g}, {%g, %g,
120. %g}\n",
121.                     vectors[i].x, vectors[i].y, vectors[i].z,
122.                     vectors[j].x, vectors[j].y, vectors[j].z,
123.                     vectors[k].x, vectors[k].y, vectors[k].z);
124.             }
125.             else { printf("not coplanar\n"); }
126.         }
127.     }
128.
129.     /// <summary>
130.     /// Метод для организации потоковой работы. Потоки создаются
131.     /// с помощью директивы параллельного цикла, в котором вызывается
132.     /// функция. Программа сама выбирает оптимальное количество потоков

```

```

133.     /// для распараллеливания цикла.
134.     /// </summary>
135.     void threadWork() {
136.
137.         #pragma omp parallel for
138.             for (size_t i = 0; i < numberOfVectors; i++) {
139.                 threadFunction(i);
140.             }
141.     }
142.
143.     /// <summary>
144.     /// Точка входа, если аргументы входной строки верны,
145.     /// отсюда вызываются методы чтения, вывода
146.     /// считанных векторов, проверки компланарности.
147.     /// </summary>
148.     int main(int argc, char** argv) {
149.
150.         if (argc != 2) {
151.             printf("wrong number of args %d\n", argc);
152.             return 1;
153.         }
154.
155.         char* input = argv[1];
156.
157.         printf("reading...\n");
158.         read(input);
159.
160.         printVectors(vectors);
161.
162.         threadWork();
163.
164.         printf("the end\n");
165.
166.         /// Освобождение памяти, выделенной для векторов.
167.         free(vectors);
168.         return 0;
169.     }

```