Министерство Образования и Науки Российской Федерации  
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
КАФЕДРА ПРИКЛАДНОЙ МАТЕМАТИКИ

**Курсовая работа**

**по практикуму на ЭВМ: структуры данных и алгоритмы**

Факультет: прикладной математики и информатики

Группа: ПМ-53

Студент: Тябин Егор Алексеевич

Преподаватель: Тракимус Юрий Викторович

### Новосибирск

### 2016

# Условие задачи

Подсчитать количество попарно неизоморфных графов с n вершинами и четырьмя ребрами.

# 2. Анализ задачи

2.1. **Исходные данные задачи:** *n* – количество вершин, *n∈N.*

2.2. **Результат:** *answer ∈ N –* количество попарно не изоморфных графов.

2.3. Решение.   
Для начала определим, какие графы будут использоваться в нашей задаче. Воспользуемся определением графа.

***Граф*** - Граф G состоит из конечного непустого множества V, содержащего р вершин, и заданного множества Х, содержащего q неупорядоченных пар различных вершин из V.  
….Отметим, что из определения вытекает, что в графе не может быть петель, т.е. ребер, соединяющих вершины сами с собой…

***Мультиграф*** – в мультиграфе не допускаются петли, но пары вершин могут быть соединены более чем 1 ребром (кратные ребра).

***Псевдограф*** – мультиграф, в котором допускаются петли.

Воспользуемся свойством графа из книги Р. Седжвика «Фундаментальные алгоритмы на с++». *Граф,* состоящий из V вершин, содержит не более V(V-1)/2 ребер.  
Исходя из этого свойства определим, сколько минимум вершин может содержать наш граф. 2\*(2-1)/2=1 ребро, 3\*(3-1)/2=3 ребра, 4\*(4-1)/2=6 ребер. Таким образом наш граф может иметь минимум 4 вершины. Для решения задачи нам понадобится определение изоморфного графа.

Два графа *G*1 и *G*2 *изоморфны*, если существует взаимно однозначное отображение множества вершин графа *G*1 на множество вершин графа *G*2, сохраняющее смежность.

Часть определения: «содержащего q неупорядоченных пар различных вершин» говори­т о том, что граф можно представить в виде неупорядоченных пар различных вершин, где количество этих пар q, где q – это количество ребер и по условию q=4.

Пусть есть множество вершин V1={A, B, C, D, E, F, H, T}, где A<=n, B<=n, …, T<=n, и A≠B≠C≠…≠T. A∈N, B∈N, … , T∈N.

Так как количество ребер ограничено по условию, то в нашей задаче существует всего несколько типов изоморфных графов. Исходя из определения изоморфных графов два графа будут неизоморфны, если нельзя сделать такие парные перестановки вершин одного графа из множества V1, что при изменении индекса вершин (например, A → B, B→A) получится другой граф. Такие типы графы находятся методом перебора:

Сразу можно отметить, что у каждой вершины есть степень –  количество рёбер графа, [инцидентных](https://ru.wikipedia.org/wiki/%D0%93%D0%BB%D0%BE%D1%81%D1%81%D0%B0%D1%80%D0%B8%D0%B9_%D1%82%D0%B5%D0%BE%D1%80%D0%B8%D0%B8_%D0%B3%D1%80%D0%B0%D1%84%D0%BE%D0%B2#.D0.B8.D0.BD.D1.86.D0.B8.D0.B4.D0.B5.D0.BD.D1.82.D0.BD.D0.BE.D1.81.D1.82.D1.8C) вершине. Это понадобится в решении задачи.

1 тип: A B Графическое изображение графа: Степени вершин:

A C A B C D

D

C

A

B

A D 3 2 2 1

B C

2 тип: A B Графическое изображение графа: Степени вершин:

A C A B C D

D

C

A

B

B D 2 2 2 2

C D

3 тип: A B Графическое изображение графа: Степени вершин:

B C A B C D E

A

B

C D 1 2 2 2 1

C

D E

E

D

4 тип: A B Графическое изображение графа: Степени вершин:

B C A B C D E

A

B

C D 1 2 3 1 1

C

C E

E

D

5 тип: A B Графическое изображение графа: Степени вершин:

A C A B C D E

E

A

A D 4 1 1 1 1

B

A E

C

D

6 тип: A B Графическое изображение графа: Степени вершин:

B

C D A B C D E

A

C E 1 1 2 2 2

C

D E

E

D

7 тип: A B Графическое изображение графа: Степени вершин:

B C A B C D E F

C

A

B

D E 1 2 1 1 2 1

E F

F

D

E

8 тип: A B Графическое изображение графа: Степени вершин:

C D A B C D E F

A

B

D E 1 1 1 2 2 1

E F

F

E

D

C

9 тип: A B Графическое изображение графа: Степени вершин:

С D A B C D E F H

E

C

A

E F 1 1 1 1 1 2 1

F H

H

D

B

F

10 тип: A B Графическое изображение графа: Степени вершин:

C D A B C D E F H T

H

E

A

B

E F 1 1 1 1 1 1 1 1

H T

T

D

C

F

В задаче можно выделить признаки изоморфности по степеням вершин. Кроме графов 3-его и 6-ого типов, или 7-ого и 8-ого типов. Которые имеют одинаковые признаки по изоморфности по степеням вершин. Для того, чтобы различить их тип, надо искать пару вершин (несвязное с графом ребро), степень которых равна 1. Если такая пара нашлась, то это либо 6-й тип изоморфизма графа, либо 8-й, здесь для выявления типа идёт сравнение количества вершин со степенью 2. Если такой пары не находится, то это либо 3-й тип изоморфизма графа, либо 7-й, здесь выявление типа идёт так же, сравнением количества вершин со степенью 2.

Всевозможное количество графов с 4 вершинами и n вершинами считается по формуле: C*n*2\*(*Cn*2-1) \*(Cn2-2) \*(Cn2-3)/4!

Где Cn2– количество сочетаний 2 вершин между n вершинами. Cn2-1 – количество сочетаний 2 вершин (1-ого ребра) между n вершинами исключая вариант первого ребра. Cn2-2 – количество сочетаний 2 вершин (1-ого ребра) между n вершинами исключая первое и второе ребро. Cn2-3- количество сочетаний 2 вершин (1-ого ребра) между n вершинами исключая первое, второе ребро и третье ребро.

Всё делится на факториал 4, этим действием исключаются повторения одних и тех же графов графов.

По условию задачи надо посчитать количество неизоморфных между собой графов.   
Для этого воспользуемся следующим решением:

1) Находим всевозможные варианты графов с 4 ребрами и указанным количеством вершин. Рёбра A B и B A равны т.е. это одно и то же ребро, поэтому исключаем такие случаи из перебора путём упорядочивания пар вершин. Так же исключаются случаи перестановки рёбер т.е. перестановка рёбер A B → B C исключается в процессе перебора.

B C → A B

И в конце получаются всевозможные варианты графов с n вершинами и 4-мя рёбрами.

2) Параллельно проверяем тип графа и увеличиваем количество графов определённого типа на 1.

3) Считаем количество попарно-неизоморфных графов перемножив количество графов одного типа на количество графов других типов и суммировав все получившиеся множители.

Пример решения задачи в математической модели:

Пусть n=4.

Построим всевозможные варианты графов с 4 вершинами и 4 рёбрами в графическом виде:

1) 2) 3) 4)

4

3

1

2

4

3

1

2

4

3

1

2

2

1

3

4

# 5) 6) 7) 8)

4

3

1

2

4

3

1

2

4

3

1

2

4

3

1

2

9) 10) 11) 12)

4

3

1

2

4

3

1

2

4

3

1

2

4

3

1

2

13) 14) 15)

4

3

1

2

4

3

1

2

4

3

1

2

# 3. Структура данных

Входные данные

# 3. Программа

#include <iostream>

#include <conio.h>

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <iomanip>

using namespace std;

int isomorphic (int \*\*c, int \*\*v, int M){

int i, j, q, l, s, d;

bool isom;

int \*\*g = new int\*[M];

for (i = 0; i < M; i++){

g[i] = new int[M];

}

int\*\* o = new int\*[M];

for (i = 0; i < M; i++){

o[i] = new int[M];

}

for (i = 0; i < M - 1; i++){

q = i+1;

for (j = q; j < M; j++){

isom = true;

for (l = 0; l < M; l++){

for (d= 0; d < M; d++){

o[l][d] = c[l][d];

}

}

for (l = 0; l < M; l++){

o[l][j] = c[l][i];

o[l][i] = c[l][j];

}

for (s = 0; s < M; s++){

for (d = 0; d< M; d++){

g[s][d] = o[s][d];

}

}

for (l = 0; l < M; l++){

g[i][l] =o[j][l];

g[j][l] = o[i][l];

}

for (s = 0; s < M; s++){

for (d = 0; d < M; d++){

if (g[s][d] != v[s][d]) isom = false;

}

}

if (isom == true) return 1;

}

}

return 0;

}

void main(){

setlocale(LC\_CTYPE, "Russian");

int q, j, i, k, N, w, e, r, t, y, u, z, x, notiz, answer=0;

int g[4][2];

printf("Введите количество вершин: ");

scanf\_s("%d", &N);

int n = ((N\*(N - 1) / 2)\*((N\*(N - 1) / 2) - 1)\*(N\*(N - 1) / 2 - 2)\*(N\*(N - 1) / 2 - 3)) / (2 \* 3 \* 4);

int \*\*\*b = new int\*\* [n];

for (i = 0; i < n; i++){

b[i] = new int\* [N];

}

for (i = 0; i < n; i++){

for (j = 0; j < N; j++){

b[i][j] = new int[N];

}

}

int\*\*\* a = new int\*\*[n];

for (i = 0; i < n; i++){

a[i] = new int\*[4];

}

for (i = 0; i < n; i++){

for (j = 0; j < 4; j++){

a[i][j] = new int[2];

}

}

k = 0;

for (q = 1; q <= N; q++){

g[0][0] = q;

for (w = 1; w <= N; w++){

g[0][1] = w;

for (e = 1; e <= N; e++){

g[1][0] = e;

for (r = 1; r <= N; r++){

g[1][1] = r;

for (t = 1; t <= N; t++){

g[2][0] = t;

for (y = 1; y <= N; y++){

g[2][1] = y;

for (u = 1; u <= N; u++){

g[3][0] = u;

for (i = 1; i <= N; i++){

g[3][1] = i;

notiz = 0;

for (j = 0; j < 4; j++){

if (g[j][0] >= g[j][1]) notiz =1;

}

if (!notiz){

for (j = 0; j < 3; j++){

z = g[j][0] \* 10 + g[j][1];

x = g[j + 1][0] \* 10 + g[j + 1][1];

if (z >= x) notiz = 1;

}

}

if (!notiz){

a[k][3][1] = g[3][1];

a[k][3][0] = g[3][0];

a[k][2][1] = g[2][1];

a[k][2][0] = g[2][0];

a[k][1][1] = g[1][1];

a[k][1][0] = g[1][0];

a[k][0][1] = g[0][1];

a[k][0][0] = g[0][0];

k++;

}

}

}

}

}

}

}

}

}

for (i = 0; i < n; i++){

for (q = 0; q < N; q++){

for (w = 0; w < N; w++){

b[i][q][w] = 0;

}

}

}

for (i = 0; i < k; i++){

for (q = 0; q < N; q++){

b[i][a[i][q][0]-1][a[i][q][1]-1] = 1;

b[i][a[i][q][1]-1][a[i][q][0]-1] = 1;

}

}

q = 0;

while (q < k-1){

for (i = q+1; i < k; i++){

if (isomorphic(b[q], b[i], N)) answer++;

}

q++;

}

for (i = 0; i < n; i++){

for (j = 0; j < N; j++){

for (q = 0; q < N; q++){

printf("%d ", b[i][j][q]);

}

printf("\n");

}

printf("\n");

}

\_getch();

}