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

По условию сказано «количество попарно неизоморфных графов с n вершинами», это означает что в множестве перебираемых графов каждый будет содержать n вершин, но типы сохранятся, просто будет некоторое количество висячих вершин. То есть при n=5, например, в 1 и 2 типе будет 1 висячая вершина. Причем такие графы будут изоморфны, но не будут являться одним и тем же графом.

Пример:

D

C

A

B

D

C

A

B

E

E

и

или

E

C

A

B

E

D

C

A

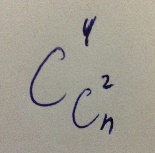
B

и

D

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 рёбрами и n вершинами считается по формуле:

 = C*n2*\*(*Cn*2-1) \*(C*n2*-2) \*(C*n2*-3)/4!

То есть количество сочетаний из Cn2 объектов(ребёр) по 4 ребра.

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.

Но эта формула понадобится только для проверки правильности решения задачи. На самом деле, не зачем хранить все графы в памяти, достаточно просто перебрать каждый возможный граф, установить его тип и увеличить количество графов полученного типа на 1.

*Выбор формальной модели*

Исходя из условия задачи графы представляются стандартно. Подсчитывается количество попарно неизоморфных графов. Таким образом имеют место следующие характеристики графа:

-Граф простой, так как нет кратных ребер и петель.

-Граф невзвешенный, так как, согласно условию задачи, каждому ребру и каждой вершине не ставится в соответствии какая-либо информация, следовательно, все вершины и рёбра в графе равнозначны.

-Граф неориентированный, т.к. в условиях задачи не сказано учитывать направление дуг.

-Граф неразмеченный, так как граф представим, как математический объект и не имеет размеченных вершин.

**Формальная постановка задачи**

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

Математическое представление графов:

1) 1 2 2) 1 2 3) 1 2 4) 1 2 5) 1 2 6) 1 2 7) 1 2 8) 1 2 9) 1 2 10) 1 2  
 1 3 1 3 1 3 1 3 1 3 1 3 1 4 1 4 1 4 2 3  
 1 4 1 4 1 4 2 3 2 3 2 4 2 3 2 3 2 4 2 4  
 2 3 2 4 3 4 2 4 3 4 3 4 2 4 3 4 3 4 3 4

11) 1 3 12) 1 3 13) 1 3 14) 1 3 15) 1 4  
 1 4 1 4 1 4 2 3 2 3  
 2 3 2 3 2 4 2 4 2 4  
 2 4 3 4 3 4 3 4 3 4

Как видно методом упорядоченного перебора получается в общем 15 графов. Среди них 6) 8) и 11) - графы 2-ого типа, остальные 1-ого типа. Следовательно, имеем 12 графов 1-ого типа и 3 графа 2-ого типа. Всего 12 неизоморфных пар с графом 6), 12 неизоморфных пар с графом 8) и 12 неизоморфных графов с графом 11). Таким образом 12\*3=36 попарно неизоморфных графов.

# 3. Структура данных, используемых для представления исходных данных и результатов задачи

*Входные данные:*

1) n – количество вершин

*Внешнее представление:* натуральное число вводимое пользователем.  
*Внутреннее представление:* переменная целого типа.

int n;

*Выходные данные:*2) answer – количество попарно неизоморфных графов

*Внешнее представление:* натуральное число.  
*Внутреннее представление:* без знаковаяпеременная целого типа long long.

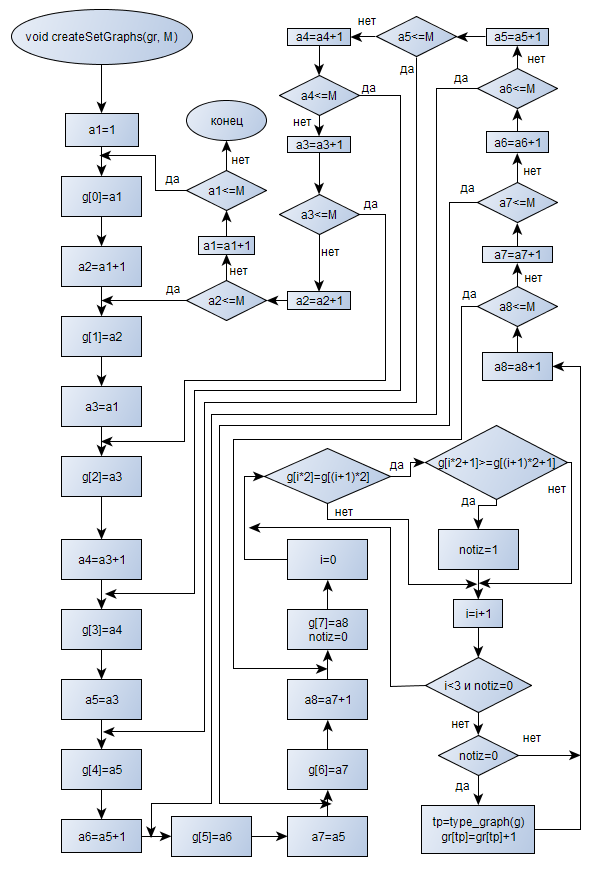
unsigned long long answer;

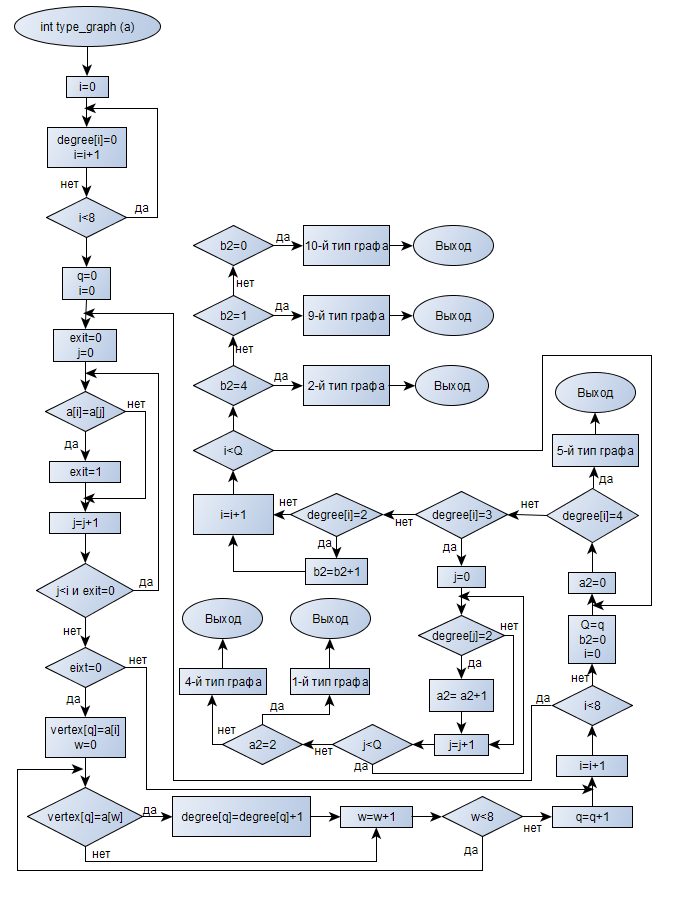
***Примечание:*** так как в C++ существует предельное значение всех типов данных, то при выборе типа данных с максимальным предельным значение, после n=36 идёт потеря данных, потому что при n=36 достигается значение близкое к максимальному, а при n=37 идёт уменьшение количество попарно неизоморфных графов, что является противоречивым. Поэтому при вводе n>36 идёт потеря данных и, следовательно, некорректный ответ.

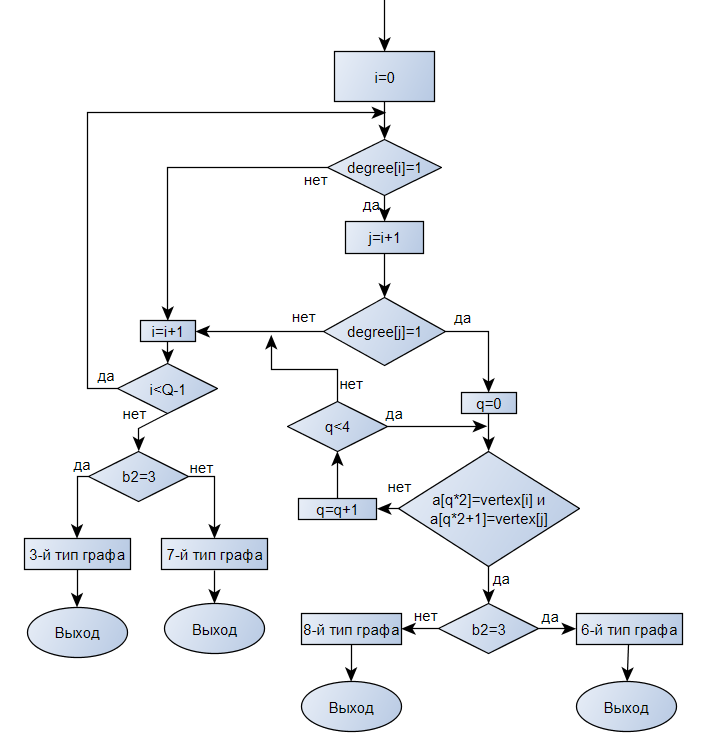
3) Для удобства проверки выводится количество типов изоморфизма.  
group[10] - массив количества каждого типа изоморфизма.

*Внешнее представление:*  последовательность натуральных чисел.  
*Внутреннее представление:* массив указателей на без знаковые переменные целого типа long long.  
unsigned long long \*group=new unsigned long long[10];

# 4. Укрупнённый алгоритм решения







# 5. Структура программы

Текст программы разбит на 3 модуля:

*Модуль main* – главный модуль программы. Осуществляет управление функциями решения основных подзадач и реализует пользовательский интерфейс.

Модуль *create –* содержит функцию перебора и функцию определения типа графа.

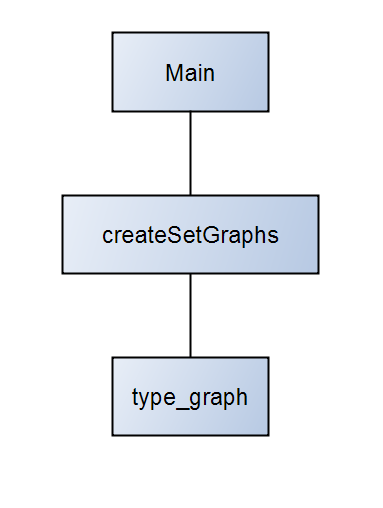
5.1 Состав модуля main

Главная функция main.   
Прототип: void main()   
Назначение: управление другими функциями, считывание входных данных и вывод результата.

5.2 Состав модуля create

Функция createSetGraphs:  
- назначение:   
перебор всевозможных вариантов графов и счет количества графов каждого типа изоморфности.  
- прототип функции:  
void createSetGraphs(unsigned long long \*&gr, int M)  
- параметры:  
gr – (выходной параметр) массив указателей на количество графов каждого типа.  
M – (входной параметр) количество вершин (n), нужные для перебора.

5.3 Функция type\_graph:  
-назначение:  
установление типа графа  
-прототип функции:  
int type\_graph(int\* a){  
-параметры:  
a –(входной параметр) указатель на массив[4\*2] (граф), тип которого определяется.

5.4 Структура программы по управлению

# 6. Текст программы

***create.h:***

#ifndef create\_H

#define create\_H

//Создание всевозможных вариантов графов с 4 ребрами и N вершинами методом перебора

void createSetGraphs(unsigned long long \*&gr, int M);

//Установление типа изоморфизма графа

int type\_graph(int\* a);

#endif

***create.cpp:***

#include <stdio.h>

#include <iostream>

#include "create.h"

void createSetGraphs(unsigned long long \*&gr, int M){

int a1, a2, a3, a4, a5, a6, a7, a8, z, x, i, tp, notiz;

int \*g = new int [8];

for (a1 = 1; a1 <= M; a1++){

g[0] = a1;

for (a2 = a1+1; a2 <= M; a2++){

g[1] = a2;

for (a3 = a1; a3 <= M; a3++){

g[2] = a3;

for (a4 = a3+1; a4 <= M; a4++){

g[3] = a4;

for (a5 = a3; a5 <= M; a5++){

g[4] = a5;

for (a6 = a5+1; a6 <= M; a6++){

g[5] = a6;

for (a7 = a5; a7 <= M; a7++){

g[6] = a7;

for (a8 = a7 + 1; a8 <= M; a8++){

g[7] = a8;

notiz = 0;

for (i = 0; i < 3 && !notiz; i++){

//Действия исключающие повторения одного и того же графа

if (g[i \* 2] == g[(i + 1) \* 2])

if (g[i \* 2 + 1] >= g[(i + 1) \* 2 + 1])

notiz = 1;

}

//Подсчет количества графов разных типов изоморфности

if (!notiz){

tp = type\_graph(g);

gr[tp]++;

}

}

}

}

}

}

}

}

}

}

int type\_graph(int\* a){

int i, j, q, w, b2, a2, Q, exit;

int vertex[8];

int degree[8];

for (i = 0; i < 8; i++){

degree[i] = 0;

}

q = 0;

//Действия считающие степень каждой вершины

for (i = 0; i < 8; i++){

exit = 0;

for (j = 0; j < i && !exit; j++){

if (a[i] == a[j]) exit = 1;

}

if (!exit){

vertex[q] = a[i];

for (w = 0; w < 8; w++){

if (vertex[q] == a[w]){

degree[q]++;

}

}

q++;

}

}

Q = q;

//Установление типа графа по степеням вершин

b2 = 0;

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

a2 = 0;

if (degree[i] == 4) return 4;

if (degree[i] == 3) {

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

if (degree[j] == 2) a2++;

}

if (a2 == 2) return 0;

else return 3;

}

if (degree[i] == 2) b2++;

}

if (b2 == 4) return 1;

if (b2 == 1) return 8;

if (b2 == 0) return 9;

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

if (degree[i] == 1){

j = i + 1;

if (degree[j] == 1){

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

if (a[q \* 2] == vertex[i] && a[q \* 2 + 1] == vertex[j]){

if (b2 == 3) return 5;

return 7;

}

}

}

}

}

if (b2 == 3) return 2;

return 6;

}

***main.cpp:***

#include "create.h"

#include <conio.h>

#include <stdio.h>

#include <iostream>

#include <stdlib.h>

#include <locale.h>

void main(){

setlocale(LC\_CTYPE, "Russian");

int n;

int i, j;

unsigned long long answer;

unsigned long long \*group=new unsigned long long[10]; //Массив в котором хранится количество графов каждого типа изоморфности (всего 10 типов)

printf("Введите количество вершин: ");

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

//Создание всевозможных вариантов графов с 4 ребрами и N вершинами методом перебора

for (i = 0; i < 10; i++){

group[i] = 0;

}

createSetGraphs(group, n);

answer = 0;

i = 0;

printf("Количество графов каждого типа: \n");

while (group[i] != 0 && i<10){

j = i+1;

printf("%llu ", group[i]);

while (group[j] != 0 && j<10){

answer += group[i] \* group[j];

j++;

}

i++;

}

printf("\nКоличество попарно неизоморфных графов с %d вершинами и 4-мя ребрами равно: \n%llu", n, answer);

\_getch();

}

# 7. Набор тестов

**Тест 1**

*Входные данные:* 2

*Назначение:* количество вершин <4

Результат: 0

Тест 2

*Входные данные:* 4

*Назначение:* проверка правильности работы с 4-мя вершинами

Результат: 36

Тест 3

*Входные данные:* 5

*Назначение:* проверка правильности работы с 5-ю вершинами

Результат: 16475

Тест 4

*Входные данные:* 6

*Назначение:* проверка правильности работы с 6-ю вершинами

Результат: 738900

Тест 5

*Входные данные:* 7

*Назначение:* проверка правильности работы с 7-ю вершинами

Результат: 14541975

Тест 6

*Входные данные:* 8

*Назначение:* проверка правильности работы с 8-ю вершинами

Результат: 171639650

Тест 7

*Входные данные:* 15

*Назначение:* проверка работоспособности, когда вершин >8

Результат: 8502983798400

Тест 8. Супертест.

*Входные данные:* 36

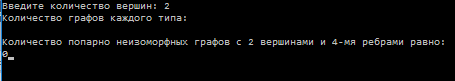
*Назначение:* супертест, проверка работоспособности при больших объемах вычислений. Максимальное количество вершин, при которых в ходе решения не происходит потери данных.

Результат: 12542389072376763600

Как видно программа вывела значение меньше предельного. При n>36 идёт потеря данных, так как значение ответа выше предельного.  
Для сравнения: 12 542 389 072 376 763 600 - результат.  
 18 426 123 330 337 413 967 - предельное значение.

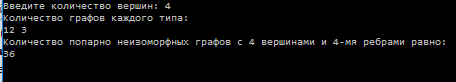
# 8. Результаты работы программы

**Тест 1  
Вывод программы:**

****

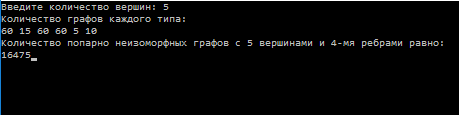
Действительно графов с 4 ребрами 2 вершинами не существует, значит количество попарно-неизоморфных графов равно 0.

**Тест 2  
Вывод программы:**

**

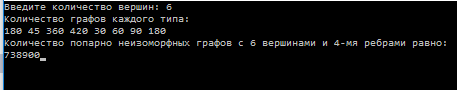
Проверим работоспособность программы с помощью формулы выведенной в анализе задачи.  
4\*3/2\*(4\*3/2-1)\*(4\*3/2-2)\*(4\*3/2-3)/\*(2\*3\*4)=15  
Действительно 12+3=15  
Задача решена верно.

**Тест 3  
Вывод программы:**

****

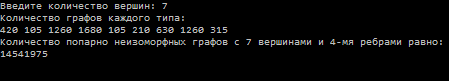
Проверим с помощью формулы.  
5\*4/2\*(5\*4/2-1)\*(5\*4/2-2)\*(5\*4/2-3)/(24)=210  
Действительно 60+15+60+60+5+10=210  
Задача решена верно.

**Тест 4  
Вывод программы:**



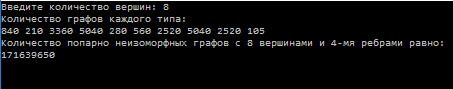
Проверим с помощью формулы.  
6\*5/2\*(6\*5/2-1)\*(6\*5/2-2)\*(6\*5/2-3)/(24)=1365  
Действительно 180+45+360+420+30+60+90+189=1365  
Задача решена верно.

**Тест 5  
Вывод программы:**



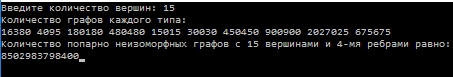
Проверим с помощью формулы.  
7\*6/2\*(7\*6/2-1)\*(7\*6/2-2)\*(7\*6/5-3)/24=5985  
Действительно 420+105+1260+1680+105+210+630+1260+315=5985  
Задача решена верно.

**Тест 6  
Вывод программы:**



Проверим с помощью формулы.  
8\*7/2\*(8\*7/2-1)\*(8\*7/2-2)\*(8\*7/2-3)/24=20475  
Действительно 840+210+3360+5040+280+560+2520+5040+2520+105=20475.  
Задача решена верно.

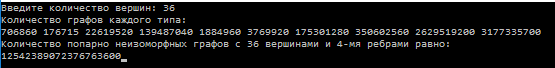
**Тест 7  
Вывод программы:**



Проверим с помощью формулы.  
15\*14/2\*(15\*14/2-1)\*(15\*14/2-2)\*(15\*14/2-3)/24=4780230  
Действительно 16380+4095+180180+480480+15015+30030+450450+900900+   
+2027025+675675=4 780 230.  
Задача решена верно.

**Тест 8**

**Вывод программы:**

****

Проверим с помощью формулы.

36\*35/2\*(36\*35/2-1) \*(36\*35/2-2) \*(36\*35/2-3)/24=6 501 403 755  
Действительно 706860+176715+22619520+139487040+1884960+3769920+175301280+350602560+2629519200  
+3177335700=6 501 403 755  
Задача решена верно

Литература:   
1) Седжвик. Фундаментальные алгоритмы на C++. Часть 5. Алгоритмы на графах. — «ДиаСофтЮП», 2002. — 496 с.

2) Новиков. Дискретная математика для программистов. — «Питер», 2009. — 384 с.   
3) Иванов. Дискретная математика. Алгоритмы и программы. — «Лаборатория Базовых Знаний», 2003. — 288 с.   
4) Кнут. Искусство программирования, том 1. Основные алгоритмы. — 2-е изд. — «Вильямс», 2006. — 720 с.   
5) Акимов. Дискретная математика: логика, группы, графы. — «Лаборатория Базовых Знаний», 2003. — 376 с.  
6) Структуры данных и алгоритмы : методические указания к курсовой работе для 1 курса ФПМИ  дневного отделения; [сост. В. П. Хиценко, Т. А. Шапошникова] НГТУ, 2008.  
7) Практикум на ЭВМ. Алгоритмы : учебное пособие для 1 курса ФПМИ дневной формы обучения; [В. П. Хиценко, Т. А. Шапошникова] НГТУ, 2004.