**Отчёт по расчётной работе по дисциплине ПиОИвИС**

**Тема:** Графы

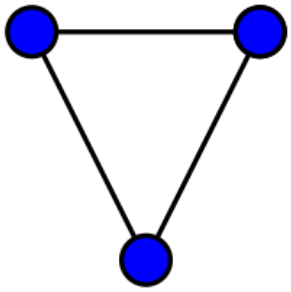
**Цель:** Определить, является ли вводимый граф – графом Паппа

**Задача:** Создать алгоритм, который будет брать из файла данные и проверять по матрице смежности является ли этот граф графом Паппа

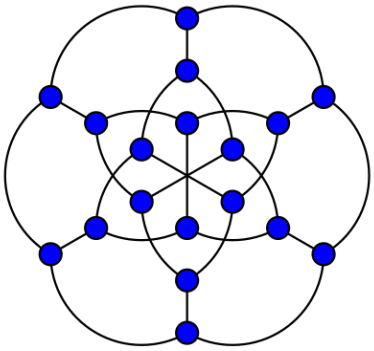
**Вариант:** 1.20(мс)

**Список ключевых понятий(определения):**

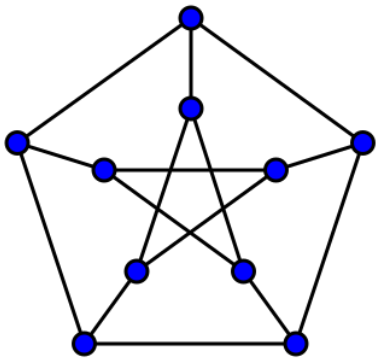
- Граф - это совокупность непустого множества вершин и множества пар вершин



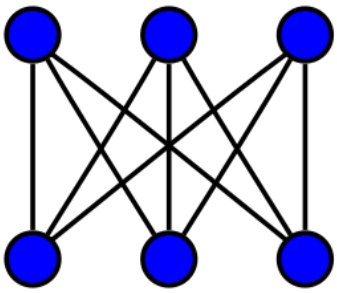
- Граф Паппа - двудольный 3-регулярный неориентированный граф с 18 вершинами и 27 рёбрами. Является единственным кубическим симметричным графом с 18 вершинами



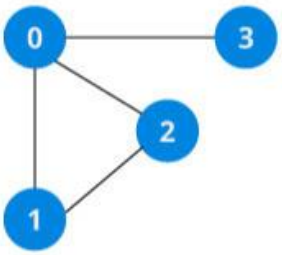
- Кубический граф - граф, в котором все вершины имеют степень три.



- Симметричный граф - граф G, для любых двух пар смежных вершин которого u1—v1 и u2—v2 имеется автоморфизм:
  f : V(G) → V(G) такой, что:
  f(u1) = u2 and f(v1) = v2.



- Матрица смежности - один из способов представления графа в виде матрицы. Матрица NxN, где N – кол-во вершин графа. Если вершины связаны, то пересечению в матрице призначается 1, иначе 0




| | 0 | 1 | 2 | 3 |
|---|---|---|---|---|
| 0 | 0 | 1 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 |
| 2 | 1 | 1 | 0 | 0 |
| 3 | 1 | 0 | 0 | 0 |

**Файлы с содержанием матрицы смежности:**

**graph.txt**

```
0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1
0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0
0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0
0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1 0
0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 1
0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0 0
0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 0
0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0
```

## graph2.txt

```
0 0 0 0 0 0 0 1 0 0 0 1 1 0 0 0 0 0
0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0 0
0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0 0
0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 1 0
0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1
0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0
0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1
0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0
```

## graph3.txt

```
0 1 1 1 0 0
1 0 1 0 1 0
1 1 0 0 0 1
1 0 0 0 1 1
0 1 0 1 0 1
0 0 1 1 1 0
```

## graph4.txt

```
0 1 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0
1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0
0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0
0 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1
0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0
0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0
0 0 0 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0
1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0
0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0
0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1 0
0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 1
0 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 1 0
```

**graph5.txt**

```
0 0 0 0 0 0 0 1 0 0 0 1 1 0
0 0 0 0 0 0 1 0 1 0 0 0 0 1
0 0 0 0 0 0 0 1 0 1 0 0 0 0
0 0 0 0 0 0 0 0 1 0 1 0 0 0
0 0 0 0 0 0 0 0 0 1 0 1 0 0
0 0 0 0 0 0 1 0 0 0 1 0 0 0
0 1 0 0 0 1 0 0 0 1 0 0 0 0
1 0 1 0 0 0 0 0 0 0 1 0 0 0
0 1 0 1 0 0 0 0 0 0 0 1 0 0
0 0 1 0 1 0 1 0 0 0 0 0 0 0
0 0 0 1 0 1 0 1 0 0 0 0 0 0
1 0 0 0 1 0 0 0 1 0 0 0 0 0
1 0 0 0 0 0 0 0 0 0 0 0 0 1
```

**Алгоритм:**

1. Выбрать файл с матрицей смежности и открыть его
2. Сосчитать количество элементов и строк в файле
3. Сравнить количество элементов с квадратом строк матрицы
   - **3.1.** Если числа не совпадают, то вывести, что это не является матрицей смежности и завершить программу
   - **3.2.** Если числа совпадают, перейти к пункту 4
4. Создать матрицу NxN, где N – кол-во строк и считать данные с файла в эту матрицу
5. Проверить правдиво ли то, что N=18
6. Проверить на правильность матрицы: чтобы элемент $a_{ij} = a_{ji}$
7. Проверить на то, чтобы каждая вершина имела ровно 3 связи
8. Вывести матрицу смежности
9. Вывести на экран является или нет этот граф графом Паппа

**Код:**

```
#include <iostream>
#include <fstream>

using namespace std;

int main()
{
	setlocale(LC_ALL, "rus");
	int numberOfGraph = 0;
	string nameOfFile = "";
	cout << "Введите номер файла, где хранится граф: ";
	cin >> numberOfGraph;
	switch (numberOfGraph)
	{
	case 1:
		nameOfFile = "graph.txt";
		break;
	case 2:
```

```
        nameOfFile = "graph2.txt";
        break;
case 3:
        nameOfFile = "graph3.txt";
        break;
case 4:
        nameOfFile = "graph4.txt";
        break;
case 5:
        nameOfFile = "graph5.txt";
        break;
default:
        break;
}
ifstream fin(nameOfFile);
char temp = 'a';
int count = 0;
int N = 1;
//Подсчёт строк и вершин
for (; !fin.eof();)
{
        temp = fin.get();
        if (temp != ' ' && temp != '\n' && !fin.eof())count++;
        if (temp == '\n') N++;
        if (temp == '\0') break;
}
fin.close();
cout << "Кол-во элементов матрицы: " << count << endl;
int M = N;
cout <<"Кол-во строк: "<< N << endl;
cout << "Кол-во столбцов: " << M << endl;
//Проверка на квадратность матрицы
if (count != N * N)
{
        cout << "В этом файле не хранится матрица смежности.";
        return 0;
}

//Создание матрицы и чтение из файла
fin.open(nameOfFile);
int **matrix = new int*[N];
for (int i = 0; i < N; i++)
{
        matrix[i] = new int[M];
}
for (int i = 0; i < N; i++)
{
        for (int j = 0; j < M; j++)
        {
                fin >> matrix[i][j];
        }
}
cout << '\n';

bool isGraph=true;
//Проверка на кол-во вершин
if (N != 18 && M != 18) isGraph = false;
//Проверка на кубичность и правильность матрицы смежности
for (int i = 0; i < N; i++)
{
        int count = 0;
        for (int j = 0; j < M; j++)
        {
                if (matrix[i][j] == 1)
                {
```

```
                        count++;
                        if (matrix[j][i] != 1) isGraph = false;
                  }
            }
            if (count != 3) isGraph = false;
      }
      //Проверка на то, имеет ли граф 27 связей
      int countOfConnections = 0;
      for (int i = 0; i < N; i++)
      {
            for (int j = 0; j < M; j++)
            {
                  if (matrix[i][j] == 1)
                  {
                        countOfConnections++;
                  }
            }
      }
      if (countOfConnections/2 != 27) isGraph = false;
      for (int i = 0; i < N; i++)
      {
            for (int j = 0; j < M; j++)
            {
                  cout << matrix[i][j];
                  cout << ' ';
            }
            cout << '\n';
      }
      cout << '\n';
      if (isGraph) cout << "Этот граф - граф Паппа";
      else cout << "Этот граф не является графом Паппа";
      fin.close();
}
```

## Вывод:

В результате выполнения данной работы были получены следующие практические навыки:

- ❖ изучены основы теории графов
- ❖ изучены способы представления графов
- ❖ изучены базовые алгоритмы для работы с графами