

**МИНОБРНАУКИ РОССИИ**  
**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**  
**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**  
**Кафедра Информационных систем**

**ОТЧЕТ**  
**по практической работе №6**  
**по дисциплине «Объектно-ориентированное программирование»**

Студент гр. 8374	_____	Пихтовников К.С.
Студент гр. 8374	_____	Подсекин Г.С.
Преподаватель	_____	Егоров С.С.

Санкт-Петербург

2021

## Задание на практическую работу

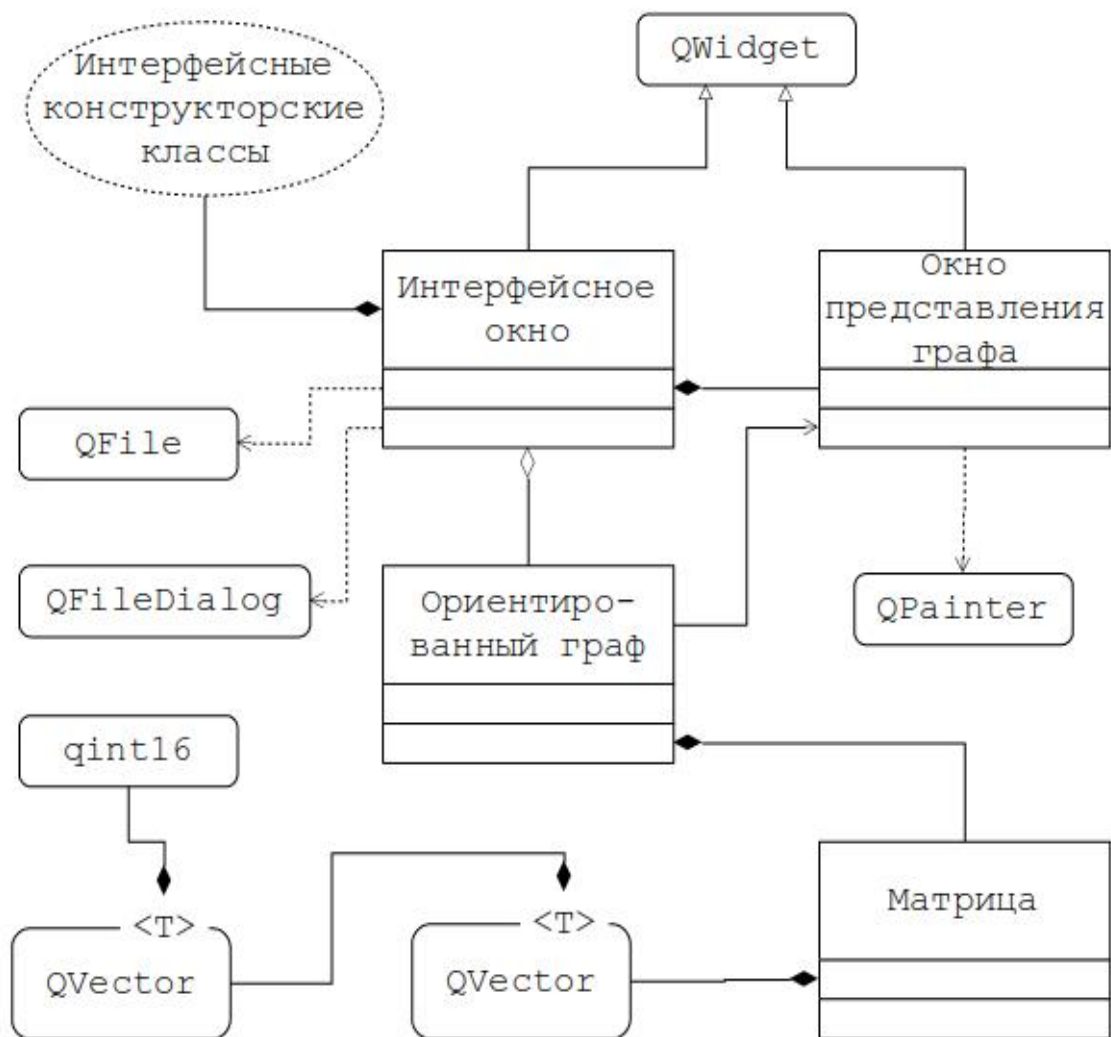
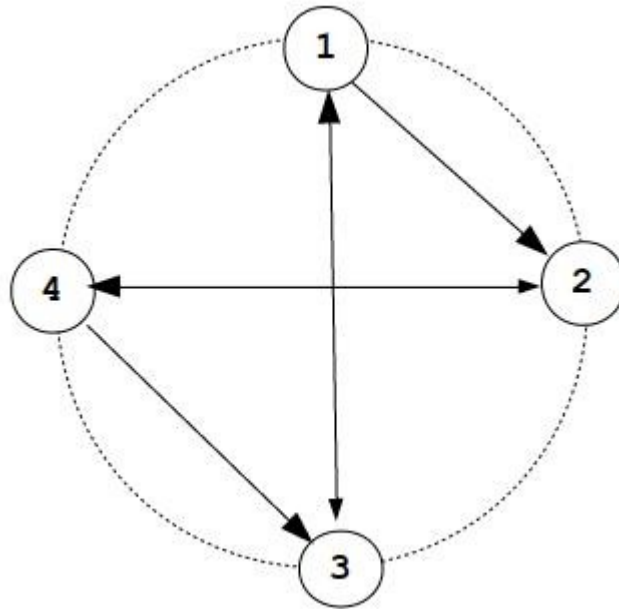


Рис.1. Диаграмма классов работы №6

Разработать GUI приложение, выполняющее функцию визуализации ориентированного графа, задаваемого матрицей смежности, представленной в виде файла, структуру которого требуется разработать. На рис.1 представлен макет диаграммы классов приложения, который требуется реализовать в приложении.

Основной функцией объекта класса "Интерфейсное окно" является выбор файла, который содержит данные об ориентированном графе. При чтении файла необходимо проверить корректность данных и в случае обнаружения ошибки необходимо сформировать соответствующее сообщение пользователю.

При корректности данных создается объект класса "Ориентированный граф", устанавливаются (если необходимо) связи между новым объектом и существующими, после чего граф отображается в соответствующем окне (объект класса "Окно представления графа"). Пример вида графа с 4 вершинами представлен ниже:



*Рис. 2. Пример графа для работы № 6*

При выборе в интерфейсе другого графа (другого файла) старый должен заменяться на новый и перерисовываться.

Реализовать и отладить программу, удовлетворяющую сформулированным требованиям и заявленным целям. Разработать контрольные примеры и протестировать на них программу. Оформить отчет, сделать выводы по работе.

## Спецификация классов

### Класс TInterface

Метод/атрибут	Описание
Атрибут TGraph * g;	область видимости - private. Переменная, которая хранит матрицу смежности
Атрибут TCanvas * canvas;	объект класса TCanvas, область видимости private
Атрибуты: QLabel * lb_size; QSpinBox * size_; QLabel * lb_matrix; QPushButton * btn_matrix; QPushButton * btn_show;	область видимости - private. Виджеты для взаимодействия пользователя с программой
Метод void ChangeGraph(TGraph *);	область видимости - private. Метод позволяет перерисовывать граф
Метод TInterface(QWidget *parent = nullptr)	Область видимости public. Конструктор класса
Метод ~TInterface()	Область видимости public. Деструктор класса
Метод void OpenCanvas();	Формальных параметров нет, область видимости private. Метод преобразует сформированный запрос в сигнал.
Метод void OpenFile();	Формальных параметров нет, область видимости private. Метод позволяет загрузить файл
Метод void CloseCanvas();	Формальных параметров нет, область видимости private. Метод отключает сформированный запрос

*Таблица 1. Класс Tinterface*

### Класс TApplication

Метод/атрибут	Описание
Атрибут TInterface *interface	объект класса TInterface, область видимости private
Метод TApplication(int, char**);	Тип формальных параметров-int, char. Область видимости public. Конструктор класса
Метод ~TApplication();	Область видимости public. Деструктор класса

*Таблица 2. Класс Tapplication*

### Класс TGraph

Метод/атрибут	Описание
Атрибут int count;	область видимости — private. Хранит размер матрицы
Атрибут TMatrix matrix;	объект класса TMatrix, область видимости private
Методы TGraph(int, Tmatrix); ~TGraph();	область видимости — public. Конструктор и деструктор класса
Метод int getCount();	область видимости — public. Метод возвращает текущий размер матрицы
Метод void setCount(int);	Тип формальных параметров-int. Область видимости — public. Метод устанавливает размер предыдущей введенной матрицы на размер исходной
Метод void getMatrix(TMatrix);	Тип формальных параметров-Tmatrix. область видимости — public. Метод возвращает текущую матрицу

Таблица 3. Класс TGraph

#### Класс TMatrix

Метод/атрибут	Описание
Атрибут int x, y;	область видимости — private. Количество строк и столбцов
Методы Tmatrix(); ~TMatrix();	Формальных параметров нет, область видимости public. Конструктор и деструктор класса
Метод TMatrix(int, int, QVector<QVector<qint16>>);	Тип формальных параметров — int, int, QVector<QVector<qint16>>, область видимости public. Метод устанавливает кол-во строк и столбцов и размер матрицы
Метод bool Is_Adjacency_Matrix();	Формальных параметров нет, область видимости public. Метод проверяет является ли введенная матрица, матрицей смежности

Таблица 4. Класс TMatrix

#### Класс TCanvas

Метод/атрибут	Описание
Атрибут TGraph *g;	объект класса TGraph, область видимости private
Методы: TCanvas(TGraph *, QWidget*parent = 0); ~TCanvas();	область видимости — public. Конструктор и деструктор класса
Методы: void paintEvent(QPaintEvent *);	область видимости — public. 1 метод рисует граф, 2 закрывает окно при

void closeEvent(QCloseEvent*);	нажатии на определенную кнопку
--------------------------------	--------------------------------

*Таблица 5. Класс TCanvas*

## Диаграмма классов

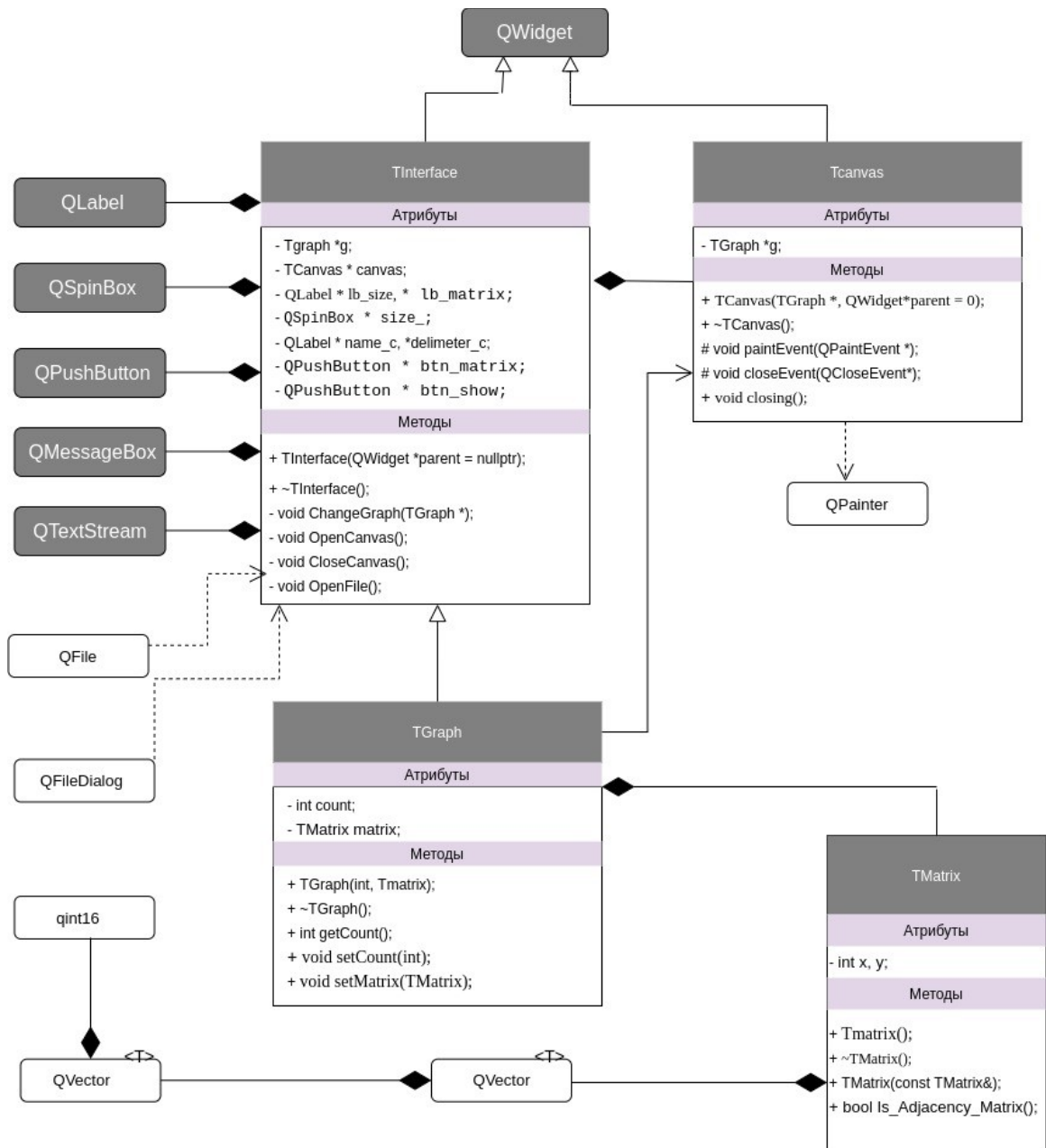


Рис.2. Реализация диаграммы классов клиентской части

Символ	Значение
+	public - открытый доступ
-	private - только из операций того же класса
#	protected - только из операций этого же класса и классов, создаваемых на его основе

Таблица 6. Обозначение атрибутов и методов класса

## Описание контрольного примера с исходными и ожидаемыми (расчетными) данными

### Пример 1:

Исходные данные:

Матрица смежности:

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

Ожидаемые данные:

Должен быть построен граф с 3 вершинами, у которого стрелки направлены: из 1 вершины во 2, из 2 вершины в 3.

### Пример 2:

Исходные данные:

Матрица смежности:

$$\begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Ожидаемые данные:

Должен быть построен граф с 4 вершинами, у которого стрелки направлены: из 1 вершины во 2 и 4; из 2 вершины в 3; из 3 вершины в 4.

### Пример 3:

Исходные данные:

Матрица смежности:

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$

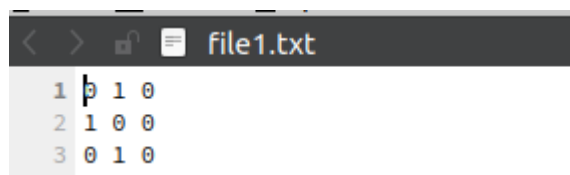
Ожидаемые данные:

Должен быть построен граф с 6 вершинами, у которого стрелки направлены: из 1 вершины в 2,3,5; из 2 вершины в 1,3,5; из 3 вершины в 1,2,4; из 4 вершины в 3,5,6; из 5 вершины в 1,2,4,6; из 6 вершины в 4,5.

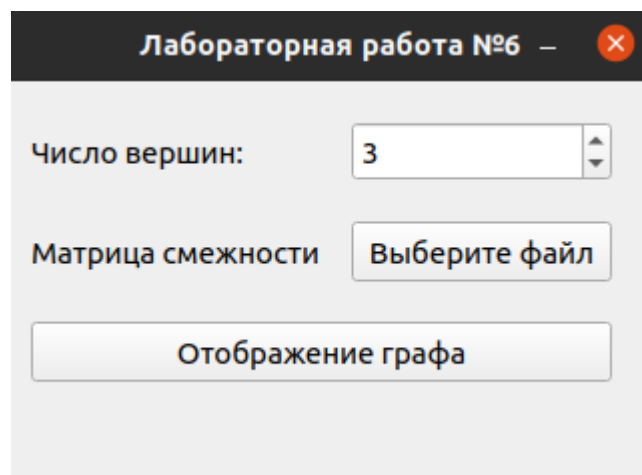


## Скриншоты программы на контрольных примерах

### Пример 1:



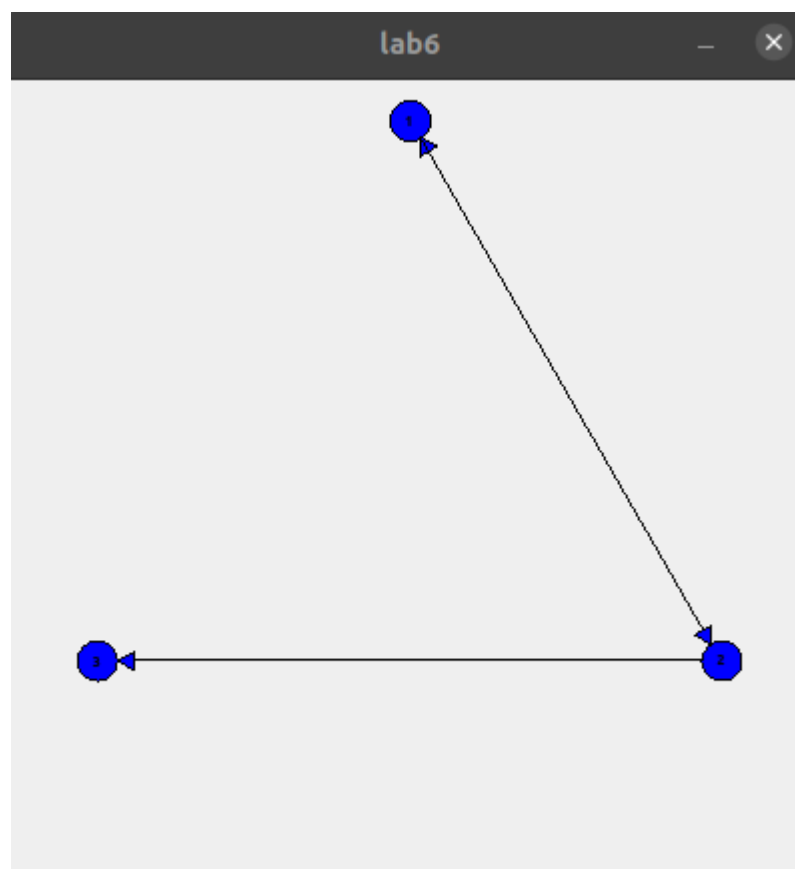
```
1 0 1 0
2 1 0 0
3 0 1 0
```



Лабораторная работа №6

Число вершин:

Матрица смежности



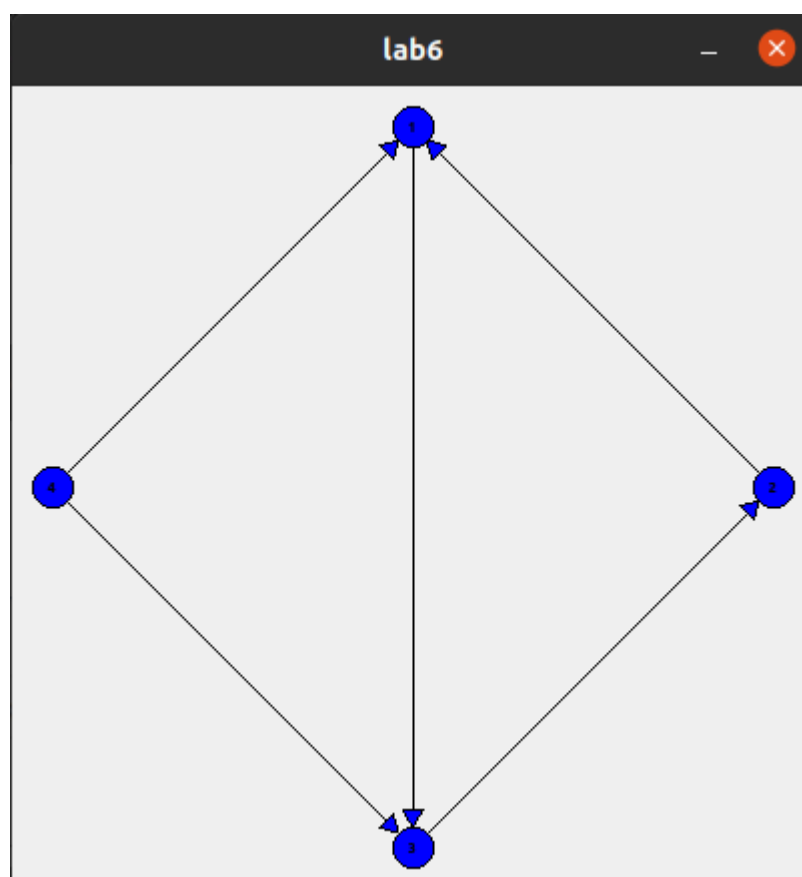
## Пример 2:

```
< > 🔒 📄 file2.txt
1 1 1 0 1
2 0 0 1 0
3 1 0 0 1
4 0 0 0 0
5
```

Лабораторная работа №6 — ✕

Число вершин:

Матрица смежности



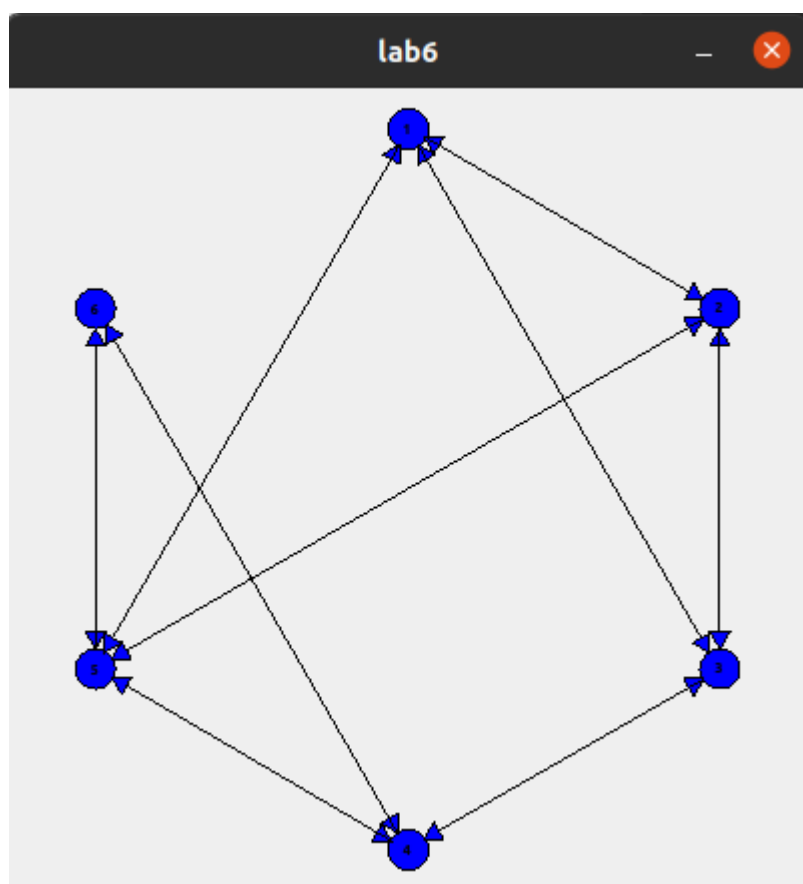
### Пример 3:

```
file3.txt
1 0 1 1 0 1 0
2 1 0 1 0 1 0
3 1 1 0 1 0 0
4 0 0 1 0 1 1
5 1 1 0 1 0 1
6 0 0 0 1 1 0
7
```

Лабораторная работа №6

Число вершин:

Матрица смежности



## **Вывод**

В ходе данной лабораторной работы было создано GUI приложение, выполняющее функцию визуализации ориентированного графа, задаваемого матрицей смежности, представленной в виде файла. На рис.1 представлен макет диаграммы классов приложения, который требуется реализовать в приложении.

Помимо этого, была создана диаграмма классов(рис.2) а также произведена отладка работы программы. Разработаны контрольные примеры с исходными и ожидаемыми данными, которые затем были протестированы в созданном GUI приложении. Все результаты совпали.