Министерство науки и высшего образования Российской Федерации
Муромский институт (филиал)
Федерального государственного бюджетного образовательного учреждения
высшего образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»

Факультет ИТР

Кафедра ПИн

ЛАБОРАТОРНАЯ РАБОТА №2

По Дискретной математике

Руководитель

Кульков Я.Ю.
(фамилия, инициалы)

(подпись) (дата)

Студент ПИн - 121
(группа)

Ермилов М.В.
(фамилия, инициалы)

(подпись) (дата)

Муром 2022

Лабораторная работа №2

Тема: Изучение способов задания графа.

Цель работы: Изучить способы описания и представления в ЭВМ графов.

Порядок выполнения работы

1. Составить программу, осуществляющую ввод матрицы смежности Введенную пользователем матрицу смежности вывести на экран.
2. Составить программу, осуществляющую ввод матрицы инциденций из поля ввода. При вводе матрицы, проверить на корректность ее элементов. Введенную пользователем матрицу инциденций вывести на экран.
3. Программа должна выводить на экран номера смежных с заданной пользователем вершиной.
4. Осуществить преобразование матрицу инциденций в матрицу смежности и вывести ее на экран.

Рисунок 1 – пример задания

Код по заданию:

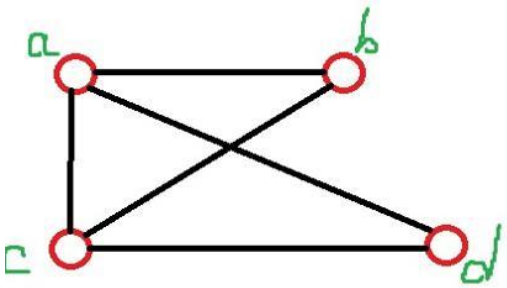
```
using lab2dis;
using System;

namespace Lab2
{
    class Program
    {
        static void Main(string[] args)
        {
            WorkArr arr = new WorkArr();
            Console.WriteLine("Creating an adjacency matrix:");
            Console.Write("Enter the number of vertices: N => ");
            int size = int.Parse(Console.ReadLine());
            int[,] arrayAdjacency = arr.CreateArrayAdjacency(size);
            arr.FindVertex(arrayAdjacency, size);
            Console.WriteLine("\n\nCreating a graph incidence matrix: ");
            int[,] arrayIncidence = arr.CreateArrayIncidence();
            Console.ReadLine();
        }
    }
}
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace lab2dis
{
      class WorkArr
      {
            public int[,] CreateArrayAdjacency(int size)
            {
                  Console.WriteLine("Write <1> when the passage from one vertex to
another is free(there is an edge).\nWrite < 0 > if false");
                  int vertex = 123;
                  int[,] arraymetod = new int[size, size];
                  for (int i = 0; i < size; i++)
                  {
                        Console.WriteLine("{0} Row:", i + 1);
                        for (int j = 0; j < size; j++)
                        {
                              bool vertexChecking = false;
                              while (vertexChecking == false)
                              {
                                    Console.Write("Enter {0} number: ", j + 1);
                                    vertex = int.Parse(Console.ReadLine());
                                    if (vertex == 0 || vertex == 1)
                                    {
                                          vertexChecking = true;
                                    }
                                    else
                                    {
                                          Console.WriteLine("Invalid input (Enter 1
or 0)");
                                    }
                              }
                              arraymetod[i, j] = vertex;
                        }
                  }
                  EnterArrayAdjacency(arraymetod, size);
                  return arraymetod;
```

```
        }
        public int[,] CreateArrayIncidence()
        {
            int vertex = 0; int edges = 0; int edgesTest = 123;
            Console.Write("Enter the number of vertices: ");
            vertex = int.Parse(Console.ReadLine());
            Console.Write("Enter the number of edges: ");
            edges = int.Parse(Console.ReadLine());
            int[,] arraymetod = new int[vertex, edges];
            Console.WriteLine("Write <1> if an edge enters a vertex.\nWrite <-1>
if an edge exits a vertex.\nWrite < 0 > if the vertex and edge are not incident.");
            for (int i = 0; i < vertex; i++)
            {
                Console.WriteLine("{0} Vertex:", i + 1);
                for (int j = 0; j < edges; j++)
                {
                    bool edgesChecking = false;
                    while (edgesChecking == false)
                    {
                        Console.Write("Enter {0} edge: ", j + 1);
                        edgesTest = int.Parse(Console.ReadLine());
                        if (edgesTest == 0 || edgesTest == 1 || edgesTest
== -1)
                        {
                            edgesChecking = true;
                        }
                        else
                        {
                            Console.WriteLine("Invalid input (Enter 1,
0 or -1)");
                        }
                    }
                    arraymetod[i, j] = edgesTest;
                }
            }
            EnterArrayIncidence(arraymetod, vertex, edges);
            return arraymetod;
        }
        //найти смежные вершины с задаваемой вершиной
        public void FindVertex(int[,] arrayIncidents, int size)
        {
            Console.WriteLine("Search for adjacent vertices");
            Console.WriteLine("Choose the vertex:");
            Console.Write("Enter index of vertex (abc-z): ");
            string vertex = Console.ReadLine();
            int indexVertex = 0;
            for (int i = 0; i < size; i++)
            {
                if (((char)(97 + i)).ToString() == vertex)
                {
                    indexVertex = i;
                }
            }
            Console.WriteLine("Entered vertes: {0}", vertex);
            for (int i = 0; i < size; i++)
            {
                if (arrayIncidents[i, indexVertex] == 1)
                {
                    Console.WriteLine("Vertex <{0}> adjacent to the vertex
<{1}>",
                       vertex, (char)(97 + i));
                }
            }
        }
        public void EnterArrayAdjacency(int[,] arraymetod, int size)
        {
```

```
                Console.WriteLine("\nArray Adjacency:");
                Console.Write(" ");
                for (int i = 0; i < size; i++)
                {
                        Console.Write("{0} ", (char)(97+i));
                }
                Console.WriteLine();
                for (int i = 0; i < size; i++)
                {
                        Console.Write("{0}|", (char)(97 + i));
                        for (int j = 0; j < size; j++)
                        {
                                Console.Write("{1} ", i, arraymetod[i, j]);
                        }
                        Console.WriteLine();
                }
        }
        public void EnterArrayIncidence(int[,] arraymetod, int vertex, int edges)
        {
                Console.WriteLine("\nArray Incidence:");
                Console.Write(" ");
                for (int i = 0; i < edges; i++)
                {
                        Console.Write("{0,2} ", (char)(97 + i));
                }
                Console.WriteLine();
                for (int i = 0; i < vertex; i++)
                {
                        Console.Write("{0,2}|", i);
                        for (int j = 0; j < edges; j++)
                        {
                                Console.Write("{1,2} ", i, arraymetod[i, j]);
                        }
                        Console.WriteLine();
                }
        }
    }
}
```



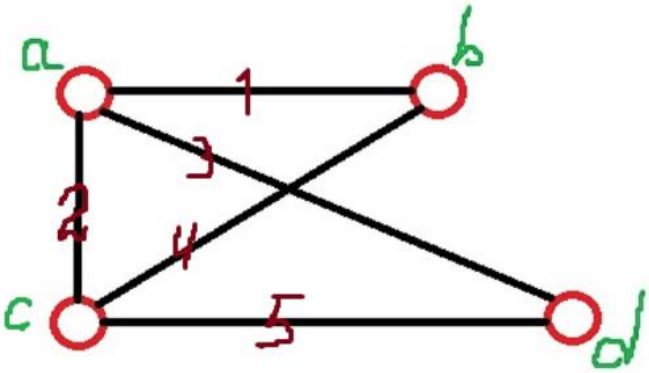
	a	b	c	d
a	0	1	1	1
b	1	0	1	0
c	1	1	0	1
d	1	0	1	0

Рисунок 2 – исходная матрица смежности

```
Creating an adjacency matrix:
Enter the number of vertices: N => 4
Write <1> when the passage from one vertex to another is free (there is an edge).
Write <0> if false
1 Row:
Enter 1 number: 0
Enter 2 number: 1
Enter 3 number: 1
Enter 4 number: 1
2 Row:
Enter 1 number: 1
Enter 2 number: 0
Enter 3 number: 1
Enter 4 number: 0
3 Row:
Enter 1 number: 1
Enter 2 number: 1
Enter 3 number: 0
Enter 4 number: 1
4 Row:
Enter 1 number: 1
Enter 2 number: 0
Enter 3 number: 1
Enter 4 number: 0

Array Adjacency:
  a b c d
a|0 1 1 1
b|1 0 1 0
c|1 1 0 1
d|1 0 1 0
Search for adjacent vertices
Choose the vertex:
Enter index of vertex (abc-z): b
Entered vertes: b
Vertex <b> adjacent to the vertex <a>
Vertex <b> adjacent to the vertex <c>
```

Рисунок 3 – заполнение и работа с матрицей смежности




```
Array Incidence:
   a  b  c  d  e
1| 1  1  1  0  0
2| 1  0  0  1  0
3| 0  1  0  1  1
4| 0  0  1  0  1
```

Рисунок 4 – исходная матрица инциденций

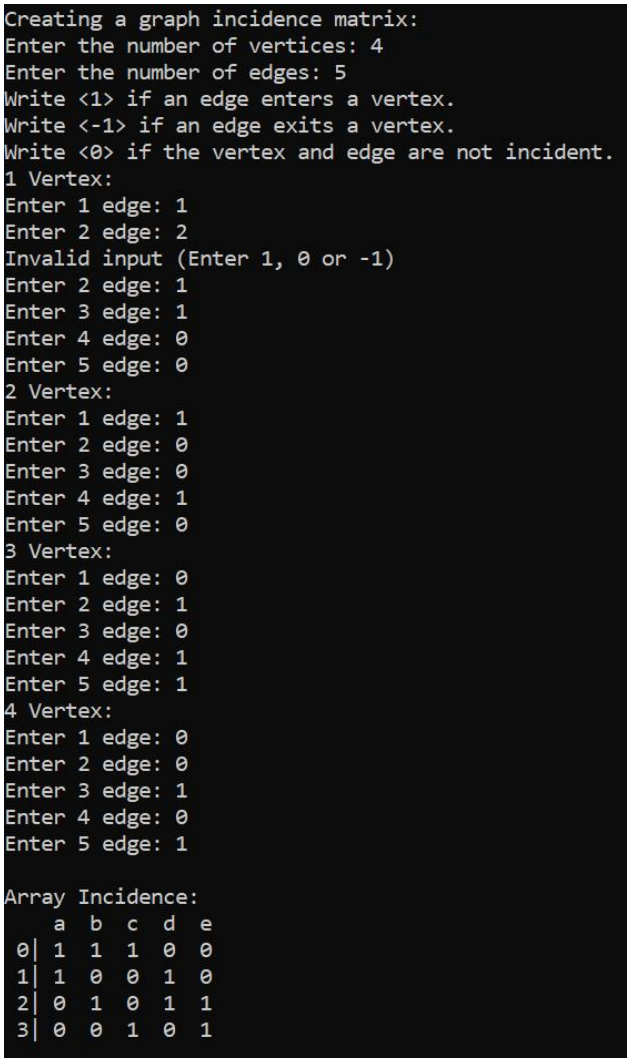


Рисунок 5 – заполнение матрицы инциденций, её вывод