Министерство образования и науки Российской Федерации
Пензенский государственный университет
Кафедра «Вычислительная техника»

ПОЯСНИТЕЛЬПАЯ ЗАПИСКА

К курсовому проектированию
по курсу «Логика и основы алгоритмизации в инженерных задачах»
на тему «Алгоритм Флойда»

Выполнил студент группы 21вв1.1:
Кузнецов А.С.
Принял
Юрова О.В.

Хорошо
28.12.22г.

Пенза 2022

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Факультет Вычислительной техники

Кафедра "Вычислительная техника"

"УТВЕРЖДАЮ"

Зав. кафедрой ВТ _____

«13» октября 2022

ЗАДАНИЕ

на курсовое проектирование по курсу

Логика и основы алгоритмизации в инженерных задачах

Студенту Кудрецову Павлу Сергеевичу Группа 21ВВ1

Тема_проекта реализация алгоритма Флойда

Исходные данные (технические требования) на проектирование

Разработка алгоритмов и программ, выполняющих в соответствии с целью задания курсового проекта.

Пояснительная записка должна содержать:

1) Постановку задачи
2) Математическую часть задания
3) Описание алгоритма написанной задачи
4) Критерии-руководство работы задания с инструкцией
5) Описание самой программы
6) Тесты
7) Список литературы
8) Листинг программы
9) Результат работы программы

Объем работы по курсу

1. Расчетная часть

Ручной расчёт работы алгоритма

2. Графическая часть

Схема алгоритма в формате блок-схема

3. Экспериментальная часть

Тестирование программы.
Подсчитать работу программы на тестовых данных

Срок выполнения проекта по разделам

1 Исследование теоретической части курсового
2 Разработка алгоритмов программы
3 Разработка программы
4 Тестирование и завершение разработки программы
5 Оформление пояснительной записки
6
7
8

Дата выдачи задания "13" сентября

Дата защиты проекта "___" _____

Руководитель Юрова О.В.

Задание получил "13" сентября 2022 г.

Студент Кудряшов Алексей Сергеевич

Содержание

Реферат 5

Введение..... 6

Постановка задачи..... 7

Теоретическая часть задания 8

Описание алгоритма программы..... 9

Описание программы..... 11

Тестирование 15

Ручной расчёт 20

Заключение 21

Список литературы 22

Приложение А. Листинг программы 23

Реферат

Отчет 33 стр, 12 рисунков.

ГРАФ, ТЕОРИЯ ГРАФОВ,АЛГОРИТМ ФЛОЙДА.

Цель исследования – разработка программы, способная находить кратчайший путь между вершинами, использую алгоритм Флойда.

В работе рассмотрен алгоритм Флойда, на основе которого

Находится кратчайшие пути между вершинами в графе. Установлено, что с

помощью данного алгоритма можно находить кратчайшие пути в графе , только если в цикле нет отрицательной величины

Введение

В информатике алгоритм Флойда(также известный как алгоритм Флойда, алгоритм Роя–Уоршелла, алгоритм Роя–Флойда или алгоритм WFI) - это алгоритм поиска кратчайших путей во взвешенном графе с положительным или отрицательным весом ребер (но без отрицательных циклов). За одно выполнение алгоритма будут найдены длины (суммарные веса) кратчайших путей между всеми парами вершин. Хотя он не возвращает детали самих путей, можно реконструировать пути с помощью простых модификаций алгоритма.

В качестве среды разработки мною была выбрана среда MicrosoftVisualStudio2022, язык программирования – C#. Целью данной курсовой работы является разработка программы на языке C#, который является широко используемым. Именно с его помощью в данном курсовом проекте реализуется алгоритм Флойда, осуществляющий поиск кратчайших путей во взвешенном графе с положительным или отрицательным весом ребер.

Постановка задачи

Требуется разработать программу, которая будет находить кратчайшие пути между вершинами в графе , используя алгоритм Флойда.

Задать исходный граф с помощью случайной генерации. Программа должна работать так, чтобы пользователь вводил количество вершин для генерации графа. После обработки этих данных на экран должна выводиться матрица на основе сгенерированного графа , после вывода матрицы в ней должны находиться все кратчайшие пути. Необходимо предусмотреть различные исходы поиска, чтобы программа не выдавала ошибок и работала правильно.

Устройство ввода – клавиатура и мышь.

Задания выполняются в соответствии с вариантом No15.

Теоретическая часть задания

Алгоритм Флойда-Уоршелла предназначен для решения задача поиска всех кратчайших путей на графе. Для заданного ориентированного взвешенного графа алгоритм находит кратчайшие расстояния между всеми парами вершин за время O(n3). Алгоритм применим к графам с произвольными, в том числе с отрицательными, весами. Таким образом, он является более общим в сравнении с алгоритмом Дейкстры, который не работает с отрицательными весами ребер. Также алгоритм распознаёт наличие отрицательных циклов.

Ключевая часть алгоритма состоит из трех циклов:
Для k от 1 до |V| выполнять
Для i от 1 до |V| выполнять
Для j от 1 до |V| выполнять
Если Dik+Dkj<Dij, то Dij:=Dik+Dkj.
При этом вычисляются минимумы из двух чисел min(Dik+Dkj,Dij) и их значения становятся новыми значениями элементов смежной матрицы.

Описание алгоритма программы

Для программной реализации алгоритма понадобиться один массив и один массив массивов:

gf(int) – массив в котором изначально хранится сгенерированная матрица по графу,

matr(int) – массив массивов с которым мы проводим реализацию самого алгоритма Флойда.

Для генерации самого графа мы создаем переменную ras , в которую пользователь с клавиатуры вводит размер графа. Далее ищем диагональ и присваиваем значение 0 , а все остальные пути заполняются рандомно и сохраняются в массив gf.

Ниже представлен псевдокод функции gf_print.

Это функция выполняет вывод матрицы графа и реализована она по средствам двух циклов for и нескольких условий:

Условие 1

```
if (i == 0 && j != 0)
{
   Console.Write(" " + Convert.ToChar(64 + j));
   continue;
}
```

Условие 2

```
                    if (j == 0 && i != 0)
{
   Console.Write(Convert.ToChar(64 + i));
   continue;
}
```

Условие 3

```
                if (i == 0 && j == 0)
{
   Console.Write("#");
   continue;
}
```

После этих условий выводится сам массив

```
Console.Write(" " + gf[i - 1, j - 1]);
```

Функция floid:

Используя переменную ras получаем размер графа , который ввел пользователь .Далее нам нужно передать заполненную матрицу графа в массив массивов

```
matr[0][i, j] = gf[i, j];
```

когда мы передали значения применяем стандартный алгоритм Флойда

Для k от 1 до |V| выполнять

Для i от 1 до |V| выполнять

Для j от 1 до |V| выполнять

Если Dik+Dkj<Dij, то Dij:=Dik+Dkj

Описание программы

Для написания данной программы использован язык программирования C#.Язык программирования C# - объектно-ориентированный язык программирования общего назначения.

Проект был создан в виде консольного приложения Win32 (Visual C#).

Данная программа является многомодульной, поскольку состоит из нескольких функций: consize, sc , floid, graf_gen, graf_print.

Работа программы начинается с запроса генерации графа. Если пользователь выбрал сгенерировать граф, то на экран выводится запрос на размер графа. Затем выводятся уведомление о том, что граф успешно создан .

```
Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
        Console.WriteLine(" 1) Сгенерировать граф");
        Console.WriteLine(" 2) Вывести граф\n");
        Console.Write("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
        mod = Convert.ToInt32(Console.ReadKey().KeyChar);
        if ((mod < 49 || mod > 51) && mod != 32)
        {
           Console.Clear();
           consize(w, h);
           goto MenuMain;
        }
```

Далее выбрав “Вывести граф” пользователь получен первоначальный граф , где еще не найдены все пути .

```
case 50://ВЫВОД ГРАФА
                {
                   if (grf.matrSM == null)
                   {
                      Console.Clear();
                      consize(w, h + 4);
                      Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                      sc(2);
                      Console.WriteLine(" графы не сгенерированы\n");
                      sc(0);
                      goto MenuMain;
```

```
                }
                Console.Clear();
                Console.WriteLine();
                consize(Convert.ToInt32(Math.Sqrt(grf.matrSM.Length)) * 2 + 2,
Convert.ToInt32(Math.Sqrt(grf.matrSM.Length)) + 5);
                for (int i = 0; i <= Math.Sqrt(grf.matrSM.Length); i++)
                {
                    Console.Clear();
                    Console.WriteLine(i + "\n");
                    graf_print(matr[i]);
                    Console.ReadKey();
                }
                Console.Clear();
                consize(w, h);
                goto MenuMain;
            }
```

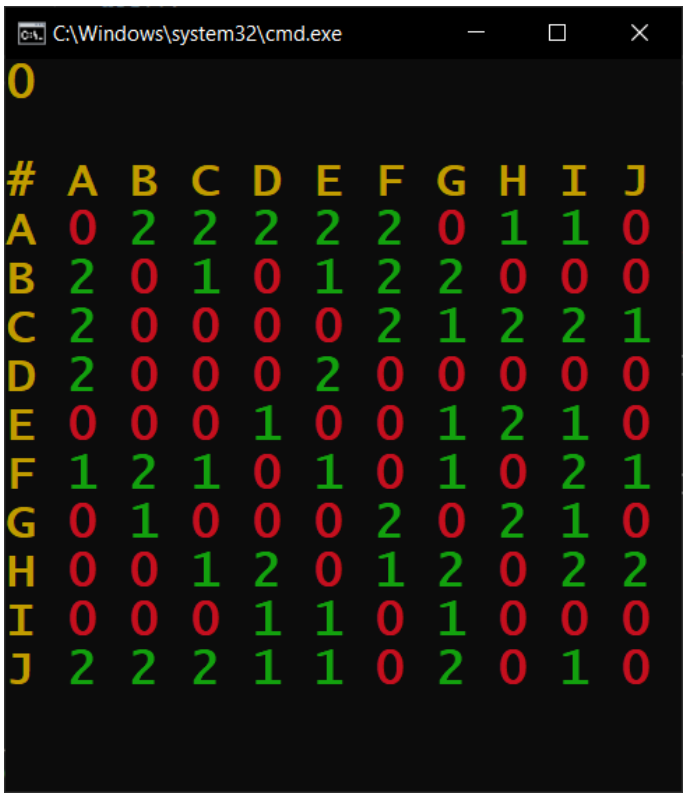
Затем нажимая на любую клавишу граф начнется заполняться недостающими значениями , и далее будет выбираться кратчайший путь и с заменять собой старый более длинный.

```
int ras = Convert.ToInt32(Math.Sqrt(grf.Length));
        for (int i = 0; i < ras; i++)
        {
            for (int j = 0; j < ras; j++)
            {
                matr[0][i, j] = grf[i, j];
            }
        }
        for (int v = 0; v < ras; v++)
        {
            for (int i = 0; i < ras; i++)
            {
                for (int j = 0; j < ras; j++)
                {
                    matr[v + 1][i, j] = matr[v][i, j];
                }
            }
            ras++;
            ras--;
            for (int i = 0; i < ras; i++)
            {
                for (int j = 0; j < ras; j++)
                {
```

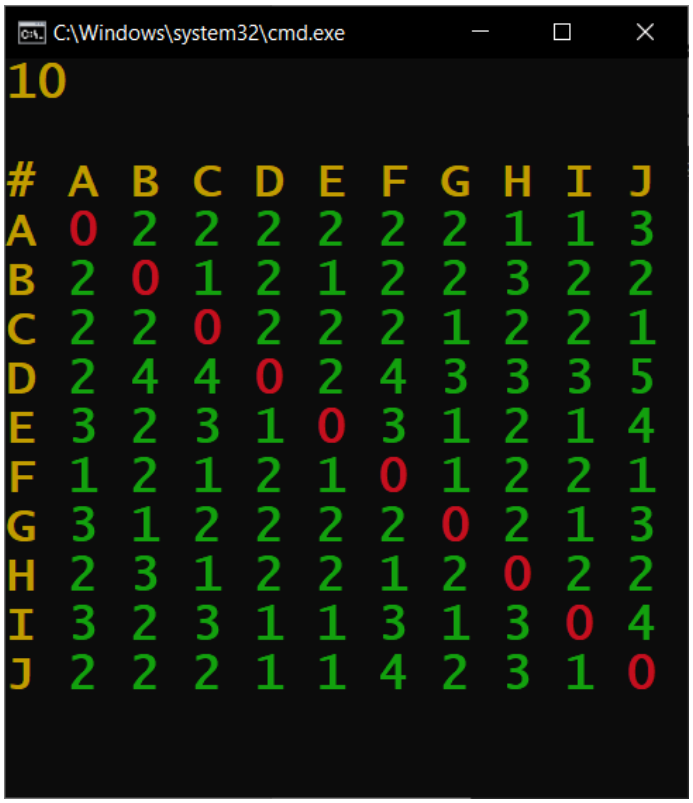
```
            if (i == v || j == v) { continue; }
            if (i == j) { continue; }
            if (matr[v][v, j] != 0 && matr[v][i, v] != 0)
            {
               if (matr[v][i, j] == 0)
               {
                  matr[v + 1][i, j] = matr[v][v, j] + matr[v][i, v];
               }
               else
               {
                  if (matr[v][i, j] > matr[v][v, j] + matr[v][i, v]) { matr[v + 1][i, j] = matr[v][v, j] + matr[v][i, v]; }
               }
            }

         }
      }
      ras++;
      ras--;
   }
}
```

Ниже можно увидеть оформление начального запроса и дальнейшие действия с ним.



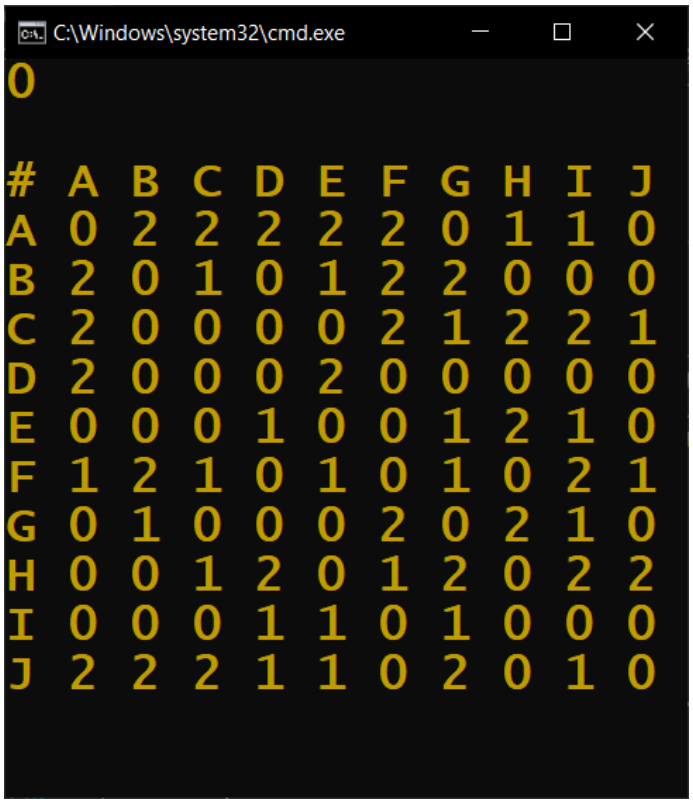
Ниже уже представлена полностью заполненая матрица кратшайших путей



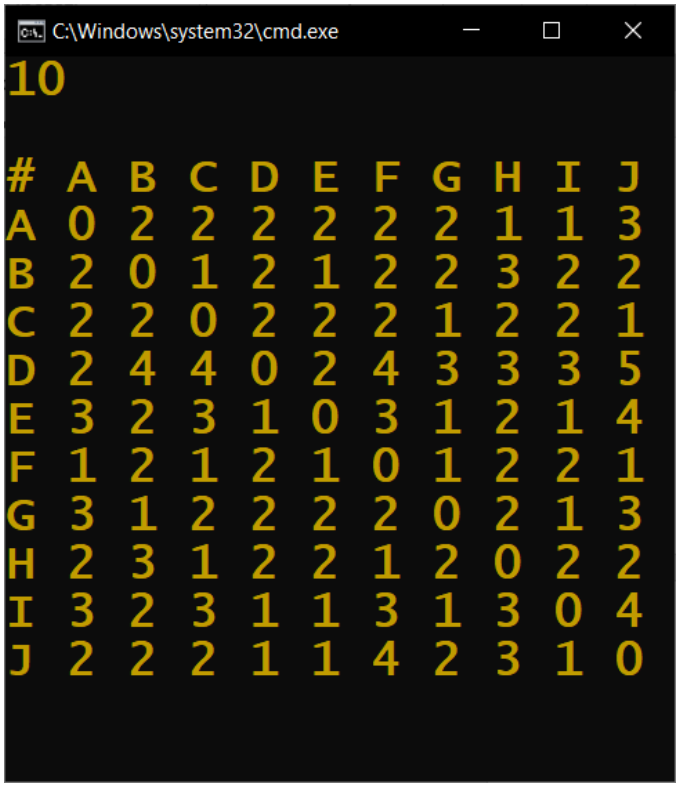
Тестирование

Среда разработки MicrosoftVisualStudio 2022 предоставляет все средства, необходимые при разработке и отладке многомодульной программы. Тестирование проводилось в рабочем порядке, в процессе разработки, после завершения написания программы. В ходе тестирования было выявлено и исправлено множество проблем, связанных с вводом данных, изменением дизайна выводимых данных, алгоритмом программы, взаимодействием функций. Ниже продемонстрирован результат тестирования программы при вводе пользователем различных размеров графа и вывод заполненных матриц

Ниже продемонстрирован результат тестирования программы при генерации матрицы на основе графа для матрицы 10x10

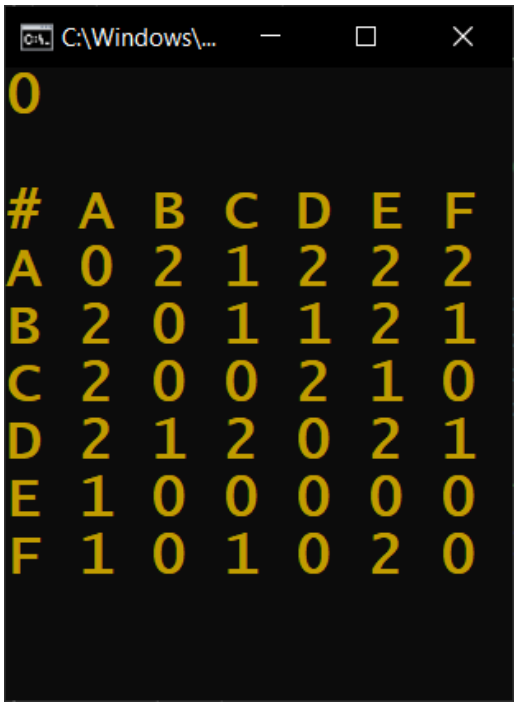


Конечный результат:



Для матрицы 6x6:

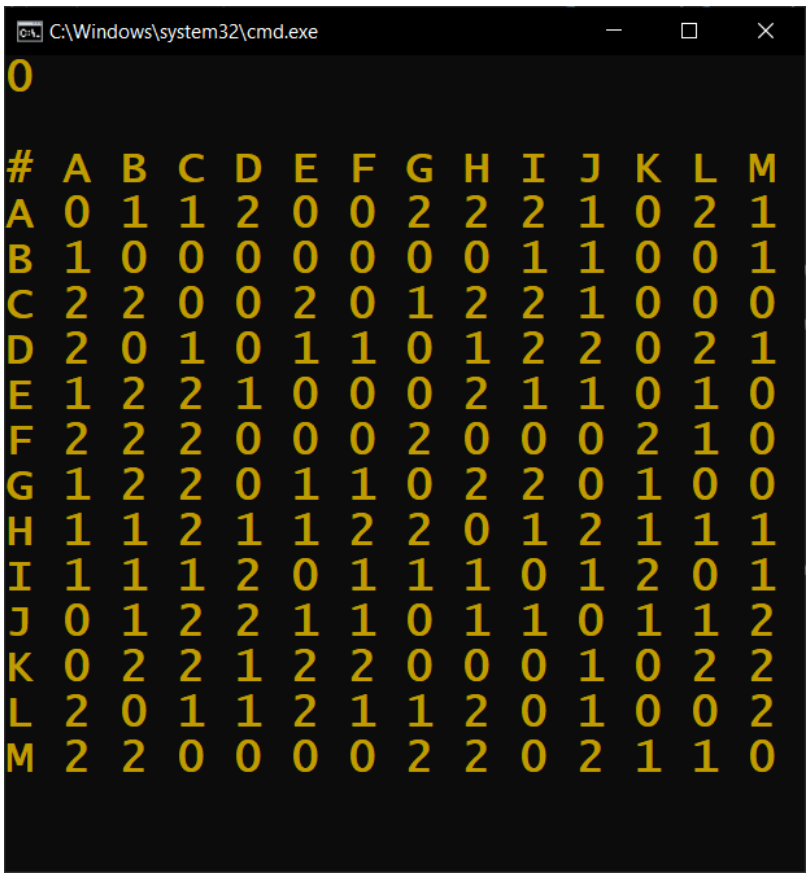
Исходные данные:



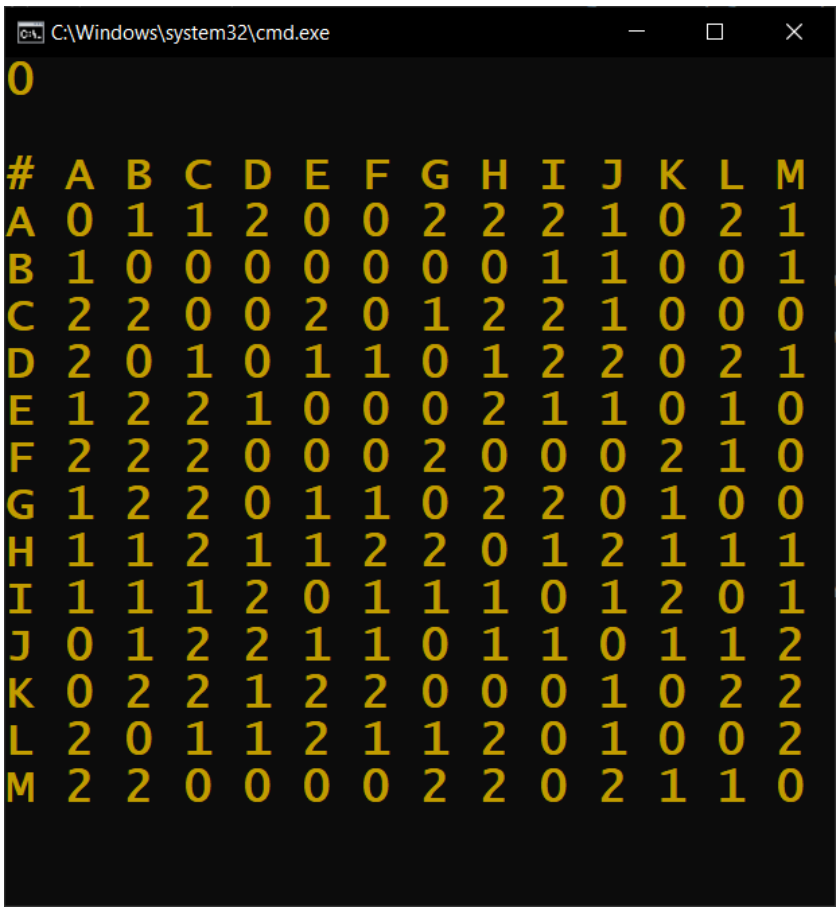
Конечный результат:

Для матрицы 13x13

Исходные данные:

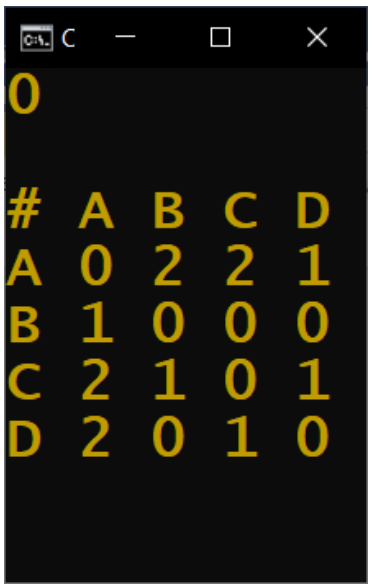


Конечный результат :



Ручной расчёт

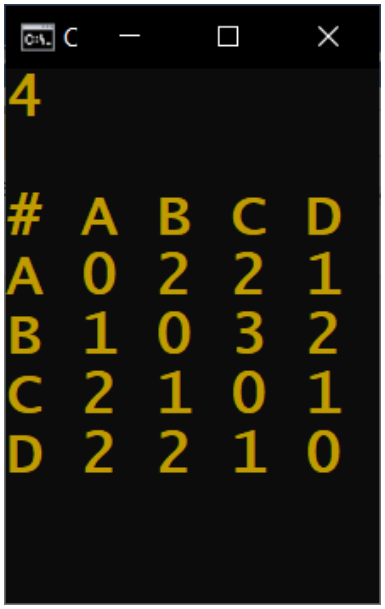
В ручную рассчитаем кратчайшие пути в графе и сравним с тем , что получится в программе на матрице 4x4



Дорога из B в C = B->A->C = 3

Дорога из B в D = B->A->D = 2

Дорога из D в B = D->C->B = 2



Ручной расчёт совпадает с работой в программе , следовательно программа работает правильно

Заключение

Таким образом, в процессе создания данного проекта разработана программа, реализующая алгоритм Флойда для поиска кратчайшего пути в MicrosoftVisualStudio 2022. При выполнении данной курсовой работы были получены навыки разработки программ и освоены приемы создания графа. Приобретены навыки по осуществлению алгоритма Флойда. Углублены знания языка программирования C#. Недостатком разработанной программы является примитивный пользовательский интерфейс. Потому что программа работает в консольном режиме, не добавляющем к сложности языка сложность программного оконного интерфейса. Программа имеет небольшой, но достаточный для использования функционал возможностей.

Список литературы

1)https://algowiki-project.org/ru/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A4%D0%BB%D0%BE%D0%B9%D0%B4%D0%B0%D0%A3%D0%BE%D1%80%D1%88%D0%B5%D0%BB%D0%BB%D0%B0

2)https://ru.wikipedia.org/wiki/%D0%90%D0%BB%D0%B3%D0%BE%D1%80%D0%B8%D1%82%D0%BC_%D0%A4%D0%BB%D0%BE%D0%B9%D0%B4%D0%B0_%E2%80%94_%D0%A3%D0%BE%D1%80%D1%88%D0%B5%D0%BB%D0%BB%D0%B0

Приложение А.

Листинг программы.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace kurs2
{
    internal class Program
    {
        struct graf
        {
            public int[,] matrSM;
        }

        const int w = 30;
        const int h = 6;

        //АЛГОРИТМ ФЛОЙДА \/ \/ \/
        static void floid(int[,] grf, int[][,] matr)
        {
            int ras = Convert.ToInt32(Math.Sqrt(grf.Length));
            for (int i = 0; i < ras; i++)
            {
                for (int j = 0; j < ras; j++)
                {
                    matr[0][i, j] = grf[i, j];
                }
            }
            for (int v = 0; v < ras; v++)
            {
                for (int i = 0; i < ras; i++)
                {
                    for (int j = 0; j < ras; j++)
                    {
                        matr[v + 1][i, j] = matr[v][i, j];
                    }
                }
                ras++;
                ras--;
```

```
        for (int i = 0; i < ras; i++)
        {
            for (int j = 0; j < ras; j++)
            {
                if (i == v || j == v) { continue; }
                if (i == j) { continue; }
                if (matr[v][v, j] != 0 && matr[v][i, v] != 0)
                {
                    if (matr[v][i, j] == 0)
                    {
                        matr[v + 1][i, j] = matr[v][v, j] + matr[v][i, v];
                    }
                    else
                    {
                        if (matr[v][i, j] > matr[v][v, j] + matr[v][i, v]) { matr[v + 1][i, j] = matr[v][v, j] + matr[v][i, v]; }
                    }
                }

            }
        }
        ras++;
        ras--;
    }
}
//ГЕНЕРАЦИЯ ГРАФА \/ \/ \/
static void graf_gen(int[,] grf)
{
    int ras = Convert.ToInt32(Math.Sqrt(grf.Length));
    Random rand = new Random(ras);
    for (int i = 0; i < ras; i++)
    {
        for (int j = 0; j < ras; j++)
        {
            if (i == j) { grf[i, j] = 0; }
            else { grf[i, j] = rand.Next(3); }
        }
    }
}
//ВЫВОД МАТРИЦЫ ГРАФА \/ \/ \/
static void graf_print(int[,] grf)
{
    int ras = Convert.ToInt32(Math.Sqrt(grf.Length));
    for (int i = 0; i <= ras; i++)
    {
        for (int j = 0; j <= ras; j++)
        {
            if (i == 0 && j != 0)
            {
                Console.Write(" " + Convert.ToChar(64 + j));
                continue;
            }
            if (j == 0 && i != 0)
            {
                Console.Write(Convert.ToChar(64 + i));
                continue;
            }
            if (i == 0 && j == 0)
            {
                Console.Write("#");
                continue;
            }
            if (true) { if (grf[i - 1, j - 1] == 0) { sc(2); } else { sc(1); } }
            Console.Write(" " + grf[i - 1, j - 1]);
            sc(0);
        }
        Console.WriteLine("");
```

```
    }
}
//ИЗМЕНЕНИЕ РАЗМЕРА КОНСОЛИ \/ \/ \/
static void consize(int weigh, int hight)
{
    Console.SetWindowSize(weigh - 1, hight - 1);
    Console.SetBufferSize(Console.WindowWidth, Console.WindowHeight);
    Console.SetWindowSize(weigh, hight);
    Console.SetBufferSize(Console.WindowWidth, Console.WindowHeight);
}
//ИЗМЕНЕНИЕ ЦВЕТА КОНСОЛИ \/ \/ \/ (0 - жёлтый; 1 - зелёный; 2 - красный)
static void sc(byte mod)
{
    switch (mod)
    {
        case 0:
            Console.ForegroundColor = ConsoleColor.DarkYellow;
            break;
        case 1:
            Console.ForegroundColor = ConsoleColor.DarkGreen;
            break;
        case 2:
            Console.ForegroundColor = ConsoleColor.DarkRed;
            break;
    }
}



static void Main(string[] args)
{
    Console.CursorVisible = false;
    int mod = 0;
    graf grf = new graf();
    int[][,] matr = new int[1][,];
    sc(0);
    consize(w, h);
MenuMain:
    Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
    Console.WriteLine(" 1) Сгенерировать граф");
    Console.WriteLine(" 2) Вывести граф\n");
    Console.Write("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
    mod = Convert.ToInt32(Console.ReadKey().KeyChar);
    if ((mod < 49 || mod > 51) && mod != 32)
    {
        Console.Clear();
        consize(w, h);
        goto MenuMain;
    }
    switch (mod)
    {
        case 49://СОЗДАНИЕ ГРАФА
            {
                int ras;
                Console.Clear();
                consize(w, 8);
            ERROR1:
                Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                Console.WriteLine(" Введите размер графа");
                Console.WriteLine(" (от 2 до 14)\n\n\n");
                Console.Write("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                Console.SetCursorPosition(15, Console.WindowHeight - 3);
                if (int.TryParse(Console.ReadLine(), out ras) == false)
                {
                    Console.Clear();
                    consize(w, 12);
```

```
                    Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                    sc(2);
                    Console.WriteLine(" введено некоректное число\n");
                    sc(0);
                    goto ERROR1;
                }
                if (ras < 2)
                {
                    Console.Clear();
                    consize(w, 13);
                    Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                    sc(2);
                    Console.WriteLine(" Размер матрицы должен");
                    Console.WriteLine(" быть больше!\n");
                    sc(0);
                    goto ERROR1;
                }
                if (ras > 14)
                {
                    Console.Clear();
                    consize(w,
                    12);
                    Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                    sc(2);
                    Console.WriteLine(" превышен максимальный размер\n");
                    sc(0);
                    goto ERROR1;
                }
                grf.matrSM = new int[ras, ras];
                matr = new int[ras + 1][,];
                for (int i = 0; i < ras + 1; i++) { matr[i] = new int[ras, ras]; }
                graf_gen(grf.matrSM);
                floid(grf.matrSM, matr);
                Console.Clear();
                consize(w, h + 4);
                Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                sc(1);
                Console.WriteLine(" !граф успешно создан!\n");
                sc(0);
                goto MenuMain;
            }
        case 50://ВЫВОД ГРАФА
            {
                if (grf.matrSM == null)
                {
                    Console.Clear();
                    consize(w, h + 4);
                    Console.WriteLine("■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■");
                    sc(2);
                    Console.WriteLine(" графы не сгенерированы\n");
                    sc(0);
                    goto MenuMain;
                }
                Console.Clear();
                Console.WriteLine();
                consize(Convert.ToInt32(Math.Sqrt(grf.matrSM.Length)) * 2 + 2,
Convert.ToInt32(Math.Sqrt(grf.matrSM.Length)) + 5);
                for (int i = 0; i <= Math.Sqrt(grf.matrSM.Length); i++)
                {
                    Console.Clear();
                    Console.WriteLine(i + "\n");
                    graf_print(matr[i]);
                    Console.ReadKey();
                }
                Console.Clear();
                consize(w, h);
```

```
                goto MenuMain;
            }
        }
    }
}
```