Министерство образования и науки РФ

Федеральное государственное автономное образовательное учреждение высшего образования

«Омский государственный технический университет»

|  |  |
| --- | --- |
| Факультет (институт) | *Информационных технологий и компьютерных систем* |
|  |  |
| Кафедра | *Прикладная математика и фундаментальная информатика* |
|  |  |

**Расчетно–графическая работа**

|  |  |
| --- | --- |
| по дисциплине | ***Дискретная математика*** |
|  |  |
| на тему | Программная реализация задач |

Пояснительная записка

|  |  |
| --- | --- |
| **Шифр проекта** | 020–РГР–02.03.02–№ 02 – ПЗ |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | **Студента** | | Бакланова Даниил Леонидовича | | | | | |
|  |  |  |  | | фамилия, имя, отчество полностью | | | | | |
|  |  |  | Курс | 1 |  | Группа | | МО–241 | | |
|  |  |  |  |  |  | |  |  | |  |
|  | | | **Направление (специальность)** | | | | | ***02.03.03*** | | |
|  | | | Математическое обеспечение и администрирование информационных систем | | | | | | | |
|  |  |  | код, наименование | | | | | | | |
|  |  |  | Руководитель | | ***ст. преподаватель*** | | | | | |
|  |  |  | ученая степень, звание | | | | | |
|  |  |  | ***Федотова И.В.*** | | | | | | | |
|  |  |  | фамилия, инициалы | | | | | | | |
|  |  |  | Выполнил | |  | | | | | |
|  |  |  | дата, подпись студента | | | | | |
|  |  |  |  | | | | | | | |
|  |  |  | **Работа защищена с количеством баллов** | | | | | |  | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | дата, подпись руководителя |  |  |  |

Омск 2025

**Содержание**

[Введение 4](#_Toc199281490)

[Постановка задачи «Города» 5](#_Toc199281491)

[Ход решения задачи «Города» 6](#_Toc199281492)

[Постановка задачи «Минное поле» 10](#_Toc199281493)

[Ход решения задачи «Минное поле» 11](#_Toc199281494)

[Постановка задачи «Расстояние между городами» 15](#_Toc199281495)

[Решение задачи «Расстояние между городами» 16](#_Toc199281496)

[Постановка задачи «из дискретной математики» 19](#_Toc199281497)

[Ход решения задачи «из дискретной математики» 20](#_Toc199281498)

[Заключение 22](#_Toc199281499)

[Список литературы 23](#_Toc199281500)

# Введение

Наша жизнь всё сильнее связывается с вычислительной техникой, которая автоматизируют различные процессы и упрощают работу, а потому важно быстро и правильно доносить свои мысли до неё. Компьютеры понимают только последовательность из нулей и единиц, а не привычный нам язык, и любая команда, поданная человеком, должна принимать именно такую форму. Для этого придумали языки программирования, которые способны наладить диалог между человеком и машиной. Человек по определенным правилам пишет команды, которые автоматически переводятся в машинный код. Но спектр задач у компьютера огромный, а потому невозможно придумать идеальный язык программирования, который подойдет под все ситуации и будет максимально эффективным. Поэтому появилось множество языков, которые эффективны в своей области. Они постепенно совершенствовались и упрощались, повышая уровень абстракции, чтобы было легче писать программы. Так мы прошли путь от Fortran к C#, JavaScript, Python и другим.

С# является высокоуровневым объектно-ориентированным языком программирования, что делает его относительно простым в изучении и использовании. Он унаследовал лучшие черты от своих предшественников: с и с++ – что делает его эффективным в работе с аппаратной частью устройства, но при этом был модифицирован: в него добавили автоматическую сборку мусора – контроль памяти и её освобождение от неиспользуемых объектов, что значительно упрощает жизнь программисту.

Целью данной расчётно-графической работы является закрепление основ C#, а также применение его для решения практический задачи из дискретной математики. В ходе работы будут изучены функции, методы работы с ними, чтение и запись файлов. А также будут задействованы основные алгоритмические структуры, и будет уделено внимание тестированию и отладке кода.

# Постановка задачи

Предположим, что вы хотите проехать из Новосибирска в Волгоград, используя магистральные шоссейные дороги, соединяющие различные города.

В построенном графе определить кратчайший путь между городами.

**Формат входных данных**

Во входном файле записано сначала число N (1<=N<=100), определявшее количество рассматриваемых городов. Затем идет число M - количество дорог в стране, далее идет описание самих дорог. Каждая дорога задается тремя числами - номерами городов, которые она соединяет и расстоянием. Все дороги двухсторонние (то есть по ним можно ездить как в одну, так и в другую сторону); между двумя городами всегда существует не более одной дороги; не существует дорог, ведущих из города в себя.

**Формат выходных данных**

На экран выведите одно число - суммарную длина маршрута или -1, если добраться невозможно.

# Программное решение

Решение данной задачи является задача поиска кратчайшего расстояния между городами. Для этой задачи я выберу алгоритм Дейкстры.

Первым этапом будет считывание данных из текстового файла. Задача представляет собой сеть из дорог, а потому её удобно решать в терминологии теории графов. Входные данные – это список дорог, по которым необходимо составить матрицу смежности. Негде не сказано, что они направленные, то мы будем их считать ребрами, поэтому матрица симметричная (рисунок 1).

Следующим этапом является реализация алгоритма Крускала (рисунок 2), а потом необходимо определить, нашелся ли путь (сравнить с изначальным значением int.MaxValue). Если да, до вывести значение, иначе вывести -1.

Выбор вершин от которой ведётся поиск пути до которой осуществляется путём указания значений переменных start и end соответственно. Итогом работы получилось число 18

static int[,] GetData(string filePath)

        {

            string[] lines = File.ReadAllLines(filePath);

            int vertexCount = int.Parse(lines[0]);

            int edge\_count = int.Parse(lines[1]);

            int[,] data = new int[vertexCount, vertexCount];

            for (int line = 2; line < 2 + edge\_count; line++)

            {

                string[] parts = lines[line].Split(' ');

                int vertex1 = int.Parse(parts[0]) - 1;

                int vertex2 = int.Parse(parts[1]) - 1;

                int weight = int.Parse(parts[2]);

                data[vertex1, vertex2] = weight;

                data[vertex2, vertex1] = weight;

            }

            return data;

        }

Рисунок 1 – код функции считывания данных

static void Main(string[] args)

        {

            int[,] data = GetData("data.txt");

            int start = 1;

            int end = 7;

            int vertexCount = data.GetLength(0);

            int[] distances = new int[vertexCount];

            int[] vertexBefore = new int[vertexCount];

            bool[] visited\_vertex = new bool[vertexCount];

            for (int vertex = 0; vertex<vertexCount; vertex++) vertexBefore[vertex] = -1;

            for (int vertex = 0; vertex<vertexCount; vertex++) distances[vertex] = (vertex == start-1) ? 0 : int.MaxValue;

            for (int step = 0; step<vertexCount-1; step++)

            {

                int closedVertex = -1;

                int min\_distance = int.MaxValue;

                for (int vertex = 0; vertex<vertexCount; vertex++)

                {

                    if (!visited\_vertex[vertex] && distances[vertex]<min\_distance)

                    {

                        min\_distance = distances[vertex];

                        closedVertex = vertex;

                    }

                }

                if (closedVertex == -1) break;

                visited\_vertex[closedVertex] = true;

                for (int neighborVertex =  0; neighborVertex<vertexCount; neighborVertex++)

                {

                    int edge\_weight = data[closedVertex, neighborVertex];

                    if (edge\_weight == 0 || visited\_vertex[neighborVertex]) continue;

                    if (distances[closedVertex] != int.MaxValue &&

                        distances[closedVertex]+edge\_weight<distances[neighborVertex])

                        {

                            distances[neighborVertex] = distances[closedVertex]+edge\_weight;

                            vertexBefore[neighborVertex] = closedVertex;

                        }

                }

            }

            if (distances[end-1] == int.MaxValue)

            {

                Console.WriteLine(-1);

            }

            else

            {

                Console.WriteLine(distances[end-1]);

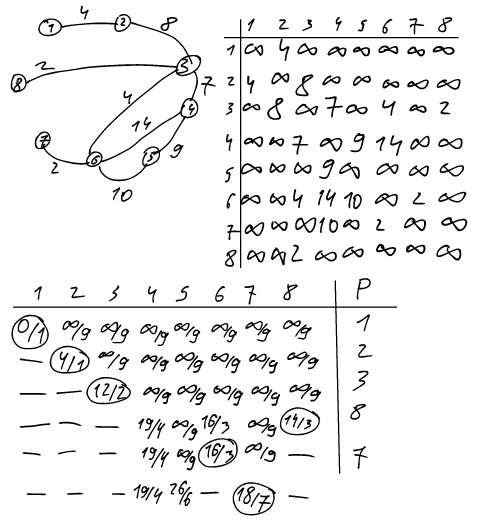
            }

        }

Рисунок 2 – код основной программы

**Ручное решение**

В качестве примера я найду кратчайшее расстояние между вершинами 1 и 6 (рисунок 3)



На этом этапе видно, что найдены оптимальные пути до вершин 2, 3, 8, 7. Искомый путь из вершины 1 в вершину 7 среди них, а потому можно завершить работу алгоритма и выдать ответ – 18.

# Заключение

В ходе выполнения работы были отработаны основы языка C# и применены знания для решения практических задач. Были изучены основные алгоритмические конструкции, методы работы с функциями и написание собственных, а также библиотека System в области работы с файлами.

В процессе выполнения работы были создана программа, использующая алгоритм Дейкстры и показывающая особенности и возможности применения языка С#. Так же был получен навык тестирования и отладки кода.

В результате выполнения работы я получил ценные знания и навыки программирования на языке C#. Но помимо этого я так же получил знания об алгоритмах, работе с данными, функциями и файлами, что так же полезно при использовании в других языках, поскольку включает общие фундаментальные принципы. Язык С# стал хорошей точкой входа в программирование. И теперь, с полученным опытом, я дальше смогу развиваться в этом направлении.

# Список литературы

1. Мюллер Д.П. C# для чайников, 3-e издание – Диалектика, Москва, 2019 – 591 с.
2. Полное руководство по языку программирования C# 13 и платформе

.NET 9: https://metanit.com/sharp/tutorial/ (дата обращения 15.12.2024).

1. Руководство по C# – управляемый язык .NET | Microsoft Learn: https://learn.microsoft.com/ru-ru/dotnet/csharp/ (дата обращения 10.12.2024)
2. C# уроки для чайников - курсы обучения C# для начинающих с нуля на itProger: https://itproger.com/course/csharp (дата обращения 16.12.2024)
3. Первая программа на C#: https://gb.ru/blog/pervaya-programma-na-c-sharp/ (дата обращения 20.12.2024).