Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Саратовский государственный технический университет

имени Гагарина Ю.А.»

[Институт прикладных информационных технологий](http://rasp.sstu.ru/) и коммуникаций

Кафедра Информационная безопасность автоматизированных систем

Направление подготовки 10.03.01 Информационная безопасность

**Расчётно-графическая работа**

по дисциплине «Языки программирования»

**Определение кратчайшего пути**

|  |  |
| --- | --- |
|  | Выполнил: студент 1 курса  учебной группы с-ИБС11  очной формы обучения  Подорван В.В.  Проверил: ассистент каф.ИБС  Романчук С.П. |

**АННОТАЦИЯ**РГР на тему:Определение кратчайшего пути  
Сроки выполнения РГР 08.05.2020 – 29.05.2020  
Выполнил: студент 1 курса группы с-ИБС11 Подорван В.В  
  
В работе выполнен анализ наиболее удобных алгоритмов решений для поиска кратчайшего пути, а так же выполнение данного метода в виде программы. В ходе анализа был выведен алгоритм “Дейкстры” и алгоритм  
 “Белмана-Форда”, а так же алгоритм“Флойда-Уоршелла”.  
  
Ключевые слова: программа, кратчайший путь.  
Работа состоит из: описание работы, скриншотов кода.

**СОДЕРЖАНИЕ** 1.Введение…………………………………4  
  
 2. Теоретическая часть……………………….5-7  
  
 3. Практическая часть……………………………8-10  
  
 4. Заключение………………………………………..11  
  
 5. Приложение…………………………………………..12-13  
  
 6. Литература………………………………………………14  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
  
   
 **ВВЕДЕНИЕ  
  
 Тема “**Поиск кратчайшего пути”  
  
**Цель задачи**: Написать программу, которая для заданных параметров создаёт некую карту городов, и определяет кратчайший путь между городами.  
  
  
  
  
  
  
  
  
  
  
 **ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**Задача о “поиске кратчайшего пути” остается актуальной во все времена и находит применение во многих областях нашей жизни. Но если ранее эта задача сводилась только к отысканию лучшего способа передвижения между пунктами А и Б, то сейчас алгоритмы нахождения кратчайшего пути используются в картографических сервисах, автопилотах при передаче пакетов в сети, и в множестве других областей.   
  
Существует множество различных постановок задачи, которые учитывают такие параметры как наличие отрицательных весов, необходимость подсчитывать не только кратчайший путь, но и входящий в его состав ребра, и т.п. Ниже приведены наиболее популярные алгоритмы решения:

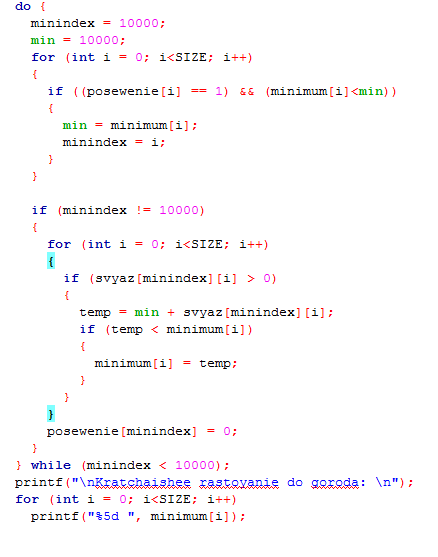
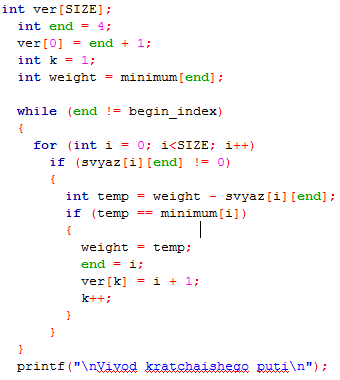
**Алгоритм Дейкстры** (находит кратчайший путь от одной из вершин графа до всех остальных; алгоритм работает только для графов без рёбер отрицательного веса)

**Алгоритм Белмана-Форда**(находит кратчайшие пути от одной вершины графа до всех остальных во взвешенном графе; вес ребер может быть отрицательным)

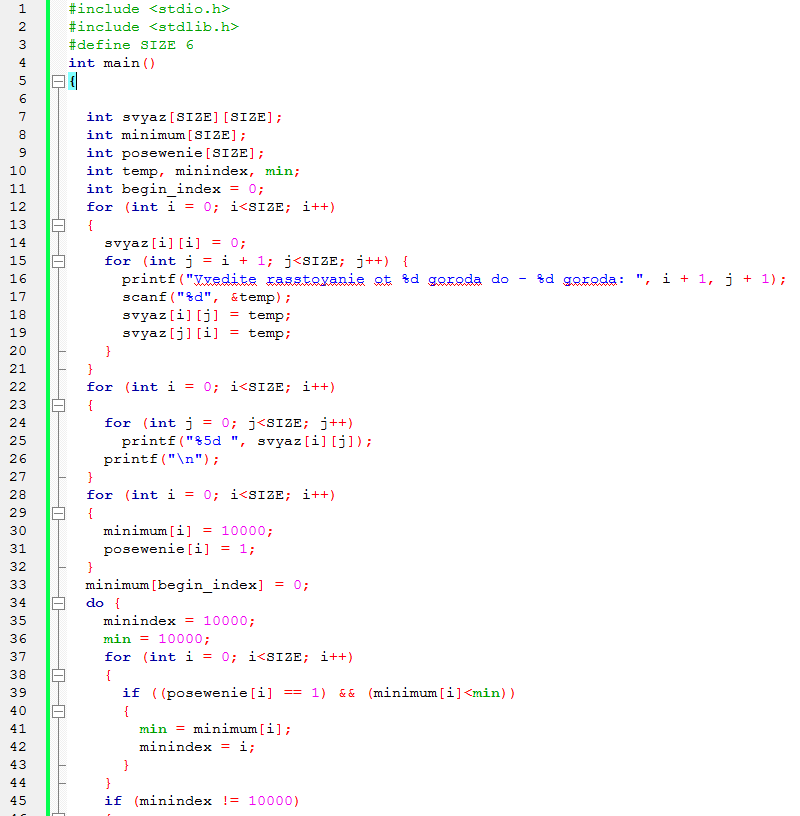
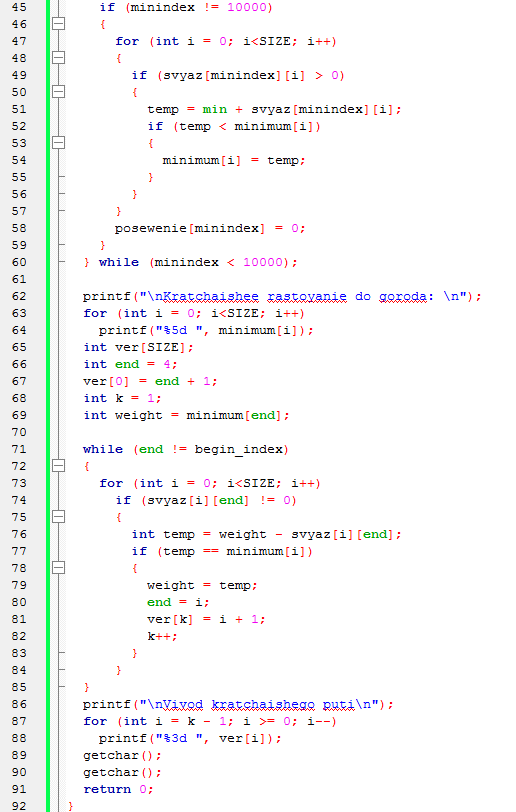
**Алгоритм Флойда-Уоршелла**(находит кратчайшие пути между всеми вершинами взвешенного ориентированного графа)  
  
В выполнении данной РГР будет выполняться метод **“Дейкстры”**Алгоритм **Дейкстры** по праву считается одним из наиболее эффективных алгоритмов решения задачи отыскания кратчайшего пути в графе для случая неотрицательных весов дуг. Он был разработан нидерландским математиком **Эдсгером Дейстрой** в 1959 году. Данный алгоритм широко применяется в программировании.  
  
Суть алгоритма заключается в следующем: сопоставим каждой вершине метку - минимальное известное расстояние от данной вершины до ***a***. Алгоритм работает пошагово, на каждом последующем шаге посещая одну из вершин, пытаясь уменьшить величину метки. Работа алгоритма завершится, когда все вершины буду посещены. Первая метка, с которой начинается алгоритм, полагается равной 0, а метки остальных вершин - бесконечности (метка бесконечности означает, что расстояние то данной вершины пока неизвестно). До того момента, когда все вершины окажутся посещенными, будет выбираться некоторая из ещё не посещённых вершин, имеющая минимальную метку. Если все вершины посещены, алгоритм   
завершается. В противном случае, из ещё не посещённых вершин выбирается вершина ***u***, имеющая минимальную метку. Вершины, в которые ведут рёбра из ***u***, назовём соседями этой вершины. Для каждого соседа вершины ***u***, кроме отмеченных как посещённые, рассмотрим новую длину пути, равную сумме значений текущей метки ***u*** и длины ребра, соединяющего ***u*** с этим соседом. Если полученное значение длины меньше значения метки соседа, заменим значение метки полученным значением длины. Рассмотрев всех соседей, пометим вершину ***u*** как посещённую и повторим шаг алгоритма.

Недостатком алгоритма Дейкстры является невозможность обработки графов, в которых имеются ребра с отрицательным весом, т. е. если, например, некоторая система предусматривает убыточные маршруты, то для работы с ней необходимо воспользоваться другим методом.  
  
  
  
  
  
  
 **Актуальность и применение данного алгоритма  
  
Алгоритм Дейкстры, а также его модификации используются в области сетей, транспортных потоков, и**, конечно же, в области обработки графов. Например, протокол динамической маршрутизации OSPF разбивает процесс построения таблицы маршрутизации на два этапа, и второй этап состоит в нахождении оптимальных маршрутов с помощью созданного на первом этапе графа. На этом этапе применяется итеративный алгоритм Дейкстры: каждый маршрутизатор считает себя центром сети и ищет оптимальный маршрут до каждой известной ему сети. В каждом найденном таким образом маршруте запоминается только один шаг - до следующего маршрутизатора. Данные об этом шаге попадают в таблицу маршрутизации. Если несколько маршрутов имеют одинаковую метрику до сети назначения, то в таблице маршрутизации запоминаются первые шаги всех этих маршрутов. Также алгоритм применяется при эвакуации населения из очагов бедствия. Оптимальные маршруты до пунктов сбора транспорта для каждой группы людей в штабе МЧС рассчитывает программа на основе алгоритма Дейкстры.

В области разработки игр широкое применение находит алгоритм А\*, классическим примером его применения является игра «Lines», «ColorLines», в которой игроку необходимо составить в ряд несколько шаров, путь шара как раз рассчитывается с помощью этого алгоритма. Кроме этого А\* применяется во многих играх жанра RTS (Real Time Strategy), его модификации применяются в таких крупных проектах как «StarCraft2» и «Warcraft3»  
**Данный метод будет осуществлён на языке программировании C++.**

**ПРАКТИЧЕСКАЯ ЧАСТЬ  
  
Алгоритм:  
1.** Заполняем нашу “карту города” , то есть вводим расстояние от 1 до N города (вершины). **2.** Происходит инициализация городов (вершин) и расстояний.  
**3.**  Далее выполняется шаг алгоритма. **4.**  Вывод кратчайшего расстояния.  
**5.** Вывод кратчайшего пути.  
  
**Используемые библиотеки:  
  
1.**  #include <stdio.h> - ввод-вывод, **2. #**include <stdlib.h> - стандартная библиотека.   
  
А так же, **макрос - #define SIZE 6 .**  
**Классы:** отсутствуют.  
  
  
  
  
  
  
  
  
  
  
  
  
 **ОПИСАНИЕ АЛГОРИТМА**  
   
**1. Поиск кратчайшего расстояния.** minindex = 10000 // индекс вершины с минимальным весом  
 min = 10000 // мин вес  
**  
  
  
  
  
  
2. Поиск кратчайшего пути.**  
  
int ver[SIZE] // массив посещенных вершин  
int end = 4 // индекс конечной вершины = 5 – 1  
ver[0] = end + 1 // начальный элемент - конечная вершина  
int k = 1 // индекс предыдущей вершины  
int weight = minimum[end] // вес конечной вершины

**ЗАКЛЮЧЕНИЕ**  
Был написан код реализующий алгоритм “**Дейкстры”.** «Задача поиска кратчайшего пути», в последнее время получила широкое распространение, благодаря своему применению для решения множества других задач. В настоящее время она применяется в алгоритмах поиска оптимального пути между двумя объектами (GPS-навигация), в системах автоматического пилотирования, для нахождения кратчайшего пути прохождения Internet-пакета по сети, и множества других.

**ПРИЛОЖЕНИЯ.  
  
  
  
**

**ЛИТЕРАТУРА  
 https://habr.com/ru/post/119158   
https://ru.wikipedia.org/wiki/Задача\_о\_кратчайшем\_пути**