ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

Государственное образовательное учреждение среднего профессионального образования

«Кемеровский государственный профессионально-педагогический колледж»

(КемГППК)

НАХОЖДЕНИЕ ПОТОКА МИНИМАЛЬНОЙ СТОИМОСТИ

Пояснительная записка

КП 230105.ММ.00.00 ПЗ

Студент гр. Пр-061

\_\_\_\_\_\_\_\_\_\_Леванов А.С.

Руководитель:

\_\_\_\_\_\_\_\_\_\_Печерских И.А.

СОДЕРЖАНИЕ

2

45

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

###### ОТ

Разраб.

Руков.

Лит.

Лист

Листов

КемГППК

гр.ПР - 061

Н. контр.

Леванов А.С.

ППК ПР - 051.В.

Печерских И.А.

Печерских И.А.

НАХОЖДЕНИЕ ПОТОКА

МИНИМАЛЬНОЙ

СТОИМОСТИ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

**У**

Инв. № подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

|  |  |
| --- | --- |
| Введение |  |
| 1 Основная часть | 5 |
| 1.1 Аналитический обзор предметной области | 5 |
| 1.2 Описание методов решения поставленной задачи | 7 |
| 1.3 Описание программы | 9 |
| 2 Общая часть | 14 |
| 2.1 Технология разработки программы | 14 |
| 2.2 Описание процесса отладки и испытания программы | 14 |
| 2.3 Надежность программного изделия | 23 |
| Заключение  Литература  Приложение А Входная информация  Приложение Б Выходная информация  Приложение В Блок-схема программы  Приложение Г Тестовые примеры  Приложение Д Текст программы |  |
|  |
|  |
|  |
|  |
|  |
|  |

ВВЕДЕНИЕ

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

3

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Современная наука широко использует математические методы как для решения прикладных, практических задач, так и для теоретического моделирования социально-экономических явлений и процессов.

Назначение математических методов в решении практических задач:

- совершенствование системы сбора информации о сложном объекте;

- интенсификация и повышение точности техническо-экономических расчетов;

- углубление количественного анализа проблем в технических, экономических и др. приложениях;

- решение принципиально новых и практических задач в любой сфере приложений.

Многие проблемы, возникающие в таких различных областях, как психология, химия, электротехника, планирование перевозок, управление, торговля и образование, могут быть сформулированы как задачи теории графов. Ввиду этого теория графов интересна не только сама по себе, но также и тем, что представляет общую основу, на которой результаты, полученные в различных областях знания, могут быть собраны, классифицированы и распространены.

В отличие от других научных дисциплин теория графов имеет вполне определенную дату рождения. Первая работа по теории графов, написанная швейцарским математиком Леонардом Эйлером (1707-1783гг.), была опубликована в 1736 г. в Трудах Академии наук в Санкт-Петербурге. Исследование Эйлера было проведено в связи с так называемой задачей о Кенигсбергских мостах. Город Кенигсберг (нынешний Калининград), располагавшийся тогда в Восточной Пруссии, был построен в месте слияния двух рек на их берегах и на двух островах. В городе было семь мостов, которые соединяли острова между собой и с береговыми частями города. Мог ли любой житель Кенигсберга, выйдя из дома, пройти по всем семи мостам города в точности по одному разу и вернуться домой? Ответ на этот вопрос должен быть отрицательным.

Данная задача получила название «Задача почтальона».

Развитие теории графов в конце XIX в начале XX вв. было связано с распространением представлений о молекулярном строении вещества и становлением теории электрических цепей. К 50-м годам нашего века в теории графов сложились два существенно различных направления: алгебраическое и оптимизационное. Последнее получило широкое развитие благодаря появлению электронных вычислительных машин (ЭВМ) и в связи с разработкой методов линейного программирования.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

4

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

При изучении транспортных или коммуникационных сетей естественно возникает математическая задача определения потока минимальной стоимости от одной точки до другой в сети, подчиненной ограничениям пропускной способности на дугах.

1 Основная часть

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

5

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

1.1 Аналитический обзор предметной области

Задача отыскания потока минимальной стоимости относится к классу задач о потоках в сетях, которые можно сформулировать как задачи линейного программирования. Но благодаря специальной структуре потоковых задач для них было получено большое число эффективных алгоритмов и изящных теорем, чем и объясняется особое внимание к этим задачам. В большинстве из них оптимальные решения всегда целочисленны, чего нельзя сказать о решении общей задачи линейного программирования.

Введем сначала понятие сети. Сеть G=(X, A) состоит из множества узлов X (называемых также вершинами или точками соединения) и множества дуг A (называемых также звеньями, или ребрами), которые связывают эти узлы. Будем считать, что число узлов и дуг конечно. Если дуга имеет определенную ориентацию (направление), то она называется ориентированной, или направленной, дугой, в противном случае она называется неориентированной дугой.

Как можно видеть, определение сети совпадает с определением графа. Термин «сеть» используется, так как с каждой дугой будет сопоставлено несколько чисел, тогда как в теории графов дуга лишь указывает на то, что узлы соединены. Пусть с каждой дугой (i, j) сопоставлено положительное число qij, называемое пропускной способностью дуги. (Используется также термин «пропускная способность ребра».) В сети выделяют два специальных узла. Один из них называется источником и обозначается s, другой называется стоком и обозначается t.

Чтобы сформулировать эту задачу более точно, введем понятие потока в сети. Потоком из источника s в сток t в сети называется множество неотрицательных чисел fij (каждое из которых поставлено в соответствие некоторой дуге сети), если эти числа удовлетворяют следующим линейным ограничениям:

где v – величина потока;

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

6

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

fij – поток по дуге (i, j), или дуговой поток.

Здесь первая сумма берется по дугам, ведущим в узел j, а вторая сумма — по дугам, ведущим из узла j. Заметим, что ограничения в формуле (1.1) выражают тот факт, что в каждый узел (кроме источника и стока) приходит столько потока, сколько из него выходит (условие сохранения). Ограничение в формуле (1.2) означает, что поток fij по дуге ограничен пропускной способностью дуги qij.

Кроме того, пусть каждой дуге (i, j) поставлена в соответствие дуговая стоимость cij, т.е. стоимость доставки единицы потока из узла i в узел j по дуге (i, j). Необходимо найти поток из источника в сток, имеющий заданную, величину и обладающий минимальной стоимостью.

Формально задача ставится следующим образом:

при ограничениях (1.1) и (1.2).

1.2 Описание методов решения поставленной задачи

Задача о минимальной стоимости была поставлена в 50-ее гг. Данцигом и Фалкерсоном. В 1961 году Басакер и Гоуэн опубликовали алгоритм для ее решения. Для текущего потока в графе строится остаточная сеть, которая для каждой пары вершин содержит прямое ребро, если поток строго меньше пропускной способности и обратное – если поток ненулевой. Стоимость прямого ребра равна стоимости исходного ребра, стоимость обратного ребра – ей противоположна. Алгоритм состоит в выполнении итераций: пока в остаточной сети существует цепь из истока в сток минимальной стоимости, увеличивать поток вдоль этой цепи.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

7

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Алгоритм Басакера–Гоуэна.

Шаг 0. Положить все дуговые потоки и величину потока v = 0.

Шаг 1. Определить модифицированные дуговые стоимости , зависящие от величины найденного потока по формуле (1.4):

Шаг 2. Найти кратчайшую цепь (т. е. цепь минимальной стоимости) из s в t, используя дуговые стоимости cij\*, найденные на шаге 1. Затем пропускать по этой цепи поток до тех пор, пока она не перестанет быть кратчайшей цепью. Получить величину нового потока, прибавив к величине старого величину потока, текущего по рассматриваемой цепи. Если величина нового потока равна v, то конец. В противном случае перейти к шагу 1.

Этот алгоритм обладает следующим интересным свойством: каждый раз на шаге 2 получается поток из источника в сток, обладающий минимальной стоимостью. Таким образом, последовательно получаются потоки минимальной стоимости величины р = 1, 2, . . ., v. По этой причине рассмотренный

алгоритм можно классифицировать как двойственный.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

8

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

В 1967 г. Клейн предложил другой способ отыскания максимального потока минимальной стоимости, основанный на «вычеркивании» циклов отрицательной стоимости. В алгоритме сначала находится произвольный максимальный поток, а затем итеративно уменьшается его стоимость в произвольном цикле с отрицательной стоимостью в остаточной сети. Процедура продолжается до тех пор, пока не останется ни одного такого цикла.

Алгоритм Клейна.

Шаг 1. Найти произвольный допустимый поток величины v из источника s в сток t. Это может быть сделано подбором или с помощью решения задачи о максимальном потоке (в которой надо проводить вычисления до тех пор, пока величина потока не станет равной v).

Шаг 2. Определить модифицированные дуговые стоимости.

Шаг 3. Используя величины cij\*, найти в сети цикл отрицательной стоимости. Для нахождения отрицательных циклов можно воспользоваться алгоритмом Флойда. Если отрицательного цикла нет, то найденный поток оптимален. Иначе увеличить поток по отрицательному циклу на величину, найденную по формуле (1.5):

δ = min{qij-fij, fij}, (1.5)

где минимум берется по всем дугам цикла, и перейти на шаг 2. (Если в сети существует несколько отрицательных циклов, то увеличить поток по каждому из них.)

Так как этот алгоритм с самого начала дает допустимый поток величины v, то его можно классифицировать как прямой алгоритм.

Легко видеть, что оба рассмотренных алгоритма позволяют найти поток величины v, если это число v не превышает величины максимального потока. Менее очевидно, что эти алгоритмы позволяют найти оптимальный поток. Доказательство этого факта основано на следующей теореме, которая может

рассматриваться как основная теорема в теории потоков минимальной стоимости.

Теорема. Поток величины v оптимален тогда и только тогда, когда в сети с модифицированными дуговыми стоимостями cij\* не существует отрицательных циклов.

Полиномиальный алгоритм для задачи о потоке минимальной стоимости был впервые предложен в 1987 г. Гольдбергом и Тарьяном. Алгоритм Голдберга-Тарьяна основан на алгоритме Клейна, однако вычеркиваются не произвольные циклы, а циклы с минимальной средней стоимостью.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

9

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Для программной реализации был выбран алгоритм Басакера-Гоуэна, на основе которого составлена блок-схема (см. приложение В, рисунок В.1).

1.3 Описание программы

Программа «Поиск потока минимальной стоимости» рассчитана на выполнение под управлением операционной системы семейства Microsoft Windows. В качестве объектно-ориентированной среды программирования была выбрана хорошо известная Borland Delphi 7, в основе которой лежит язык Object Pascal. Delphi – это комбинация нескольких важнейших технологий:

- высокопроизводительный компилятор;

- объектно-ориентированная модель компонент;

- визуальное (а, следовательно, и скоростное) построение приложений из программных прототипов;

- масштабируемые средства для построения баз данных.

Программа «Поиск потока минимальной стоимости» предназначена для автоматического решения задачи поиска потока минимальной стоимости на ЭВМ.

Описание функций составных частей программы приведено в таблице 1.1. Схема взаимосвязей основных составных частей программы изображена на рисунке 1.1.

Таблица 1.1 – Описание модулей программы

|  |  |
| --- | --- |
| Модуль | Назначение |
| Project1.dpr | Главный модуль проекта, подключающий все  остальные модули программы |
| Main.pas | Модуль основного окна программы, реализующий ввод из файла и запускающий алгоритм решения |
| Input.pas | Модуль ручного ввода начальных данных |
| FloydSearch.pas | Модуль поиска по алгоритму Флойда, являющийся составной частью алгоритма Басакера-Гоуэна |
| SplashScreen.pas | Модуль, реализующий пункт меню «О программе» |

Project1

Main

SplashScreen

FloydSearch

Input

Рисунок 1.1 – Взаимосвязь модулей программы

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

10

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Входная и выходная информация.

В качестве входной информации в программу поступают сведения о количестве вершин в сети, номерах источника и стока, величине искомого потока, а также матрицы максимальных пропускных способностей ребер сети и стоимости перевозок по ним. Примеры входной информации введенной с клавиатуры и считанной из файла приведены в приложении А.

В качестве выходной информации пользователь получает величину минимальной стоимости искомого потока и его распределение в сети в виде таблицы. Выходная информация представлена в приложении Б.

Руководство пользователя.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

11

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Для запуска программы необходимо зайти в папку программы, которую можно расположить в любой месте жесткого диска или съемного носителя, и запустить исполняемый файл «mincostflow.exe» путем двойного щелчка по иконке программы, либо одиночного с последующим нажатием клавиши Enter.

При открытии главное окно программы расползается на весь экран. Сразу под заголовком вы можете обнаружить строку меню, через пункты которого осуществляется управление программой (рис. 1.2).

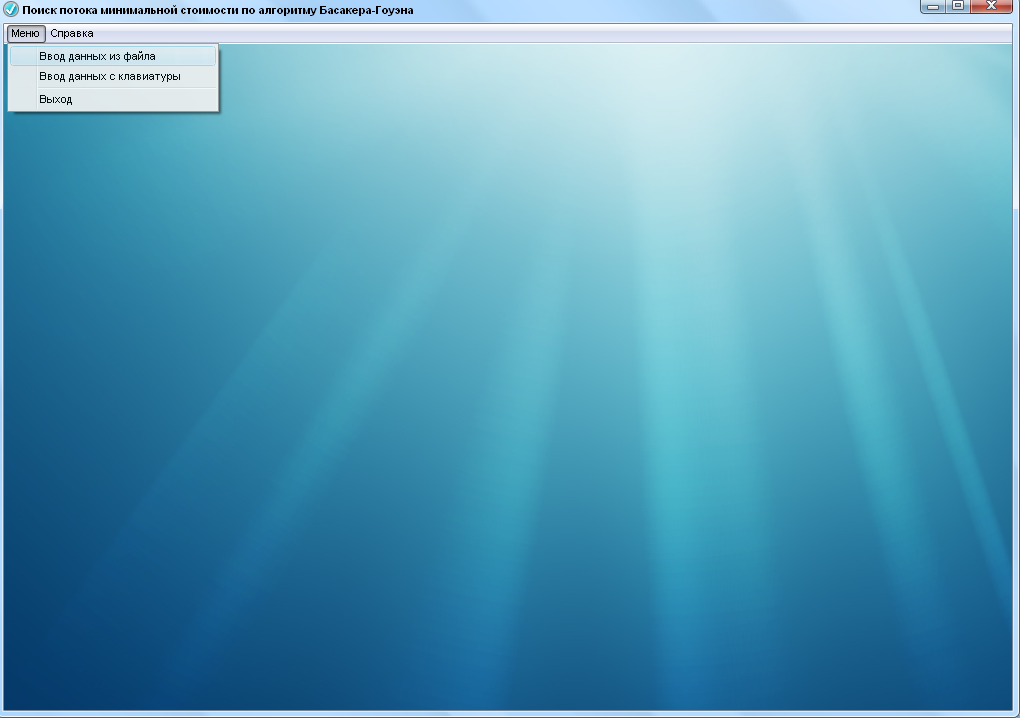


Рисунок 1.2 – Меню главного окна программы

В пункте меню «Меню» располагаются такие элементы как «Ввод данных из файла», «Ввод данных с клавиатуры» и «Выход». При помощи последнего осуществляется закрытие данной программы. А с помощью первых двух пользователь осуществляет ввод входной информации. Либо указывая с помощью диалогового окна файл, содержащий условия задачи в установленной форме,

либо последовательно вбивая их с клавиатуры при помощи мастера, реализованного в отдельном окне. В случае с ручным вводом после набора всех данных имеется возможность сохранить их в файл при нажатии по кнопке «Сохранить данные в файл», либо продолжить решение задачи без их сохранения путем нажатия по кнопке «Произвести расчеты» (рис. 1.3).

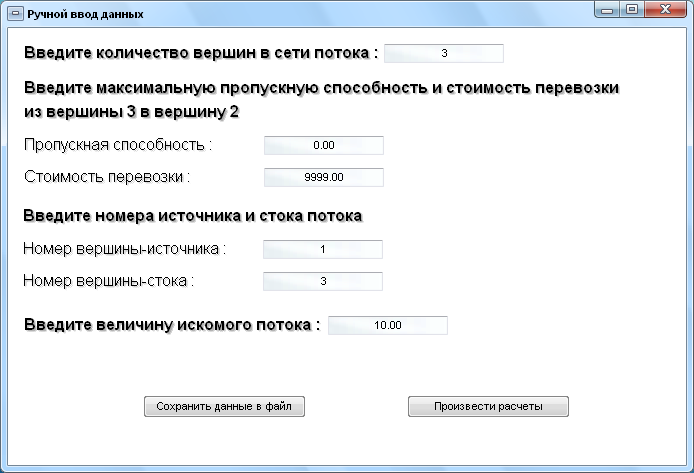


Рисунок 1.3 – Ввод данных с клавиатуры

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

12

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Решения задачи не займет много времени. После окончания работы алгоритма на главное окно программы будет выведена таблица, характеризующая распределение искомого потока минимальной стоимости в сети. А саму стоимость можно будет увидеть в разделе «Информация о потоке», в которой также выведется сообщение, когда искомый поток не возможно достичь на имеющейся сети. В таком случае результатом будет максимальный поток минимальной стоимости (рис 1.4). В случае если количество вершин больше 20 у таблицы распределения потока появляются полосы прокрутки (рис. 1.5).

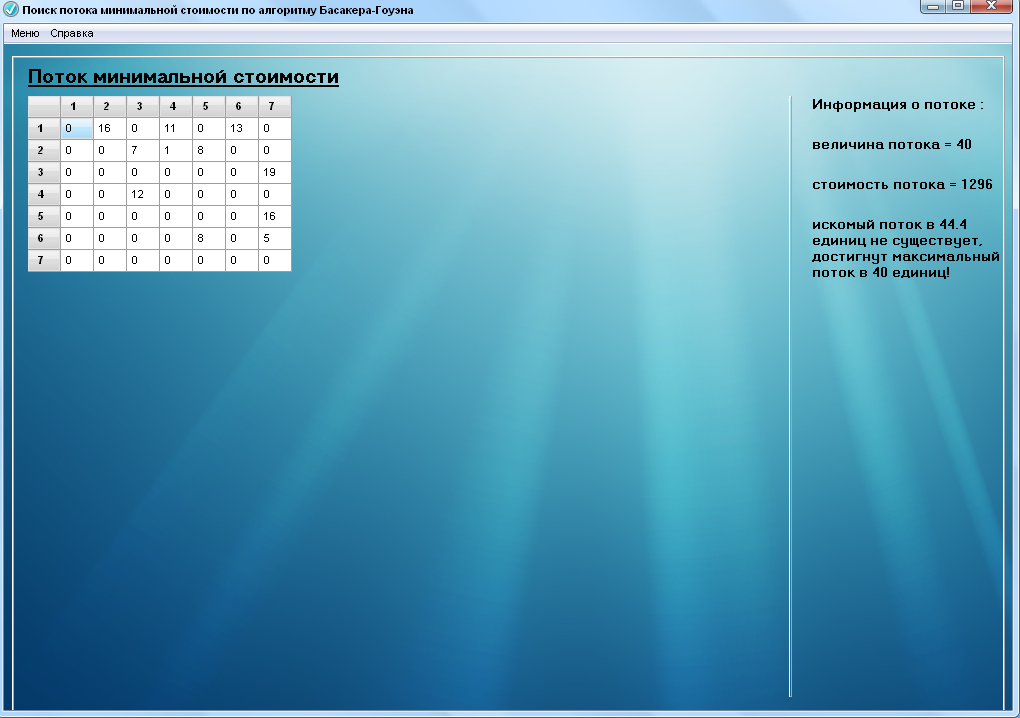


Рисунок 1.4 – Вывод результатов решения задачи

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

13

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

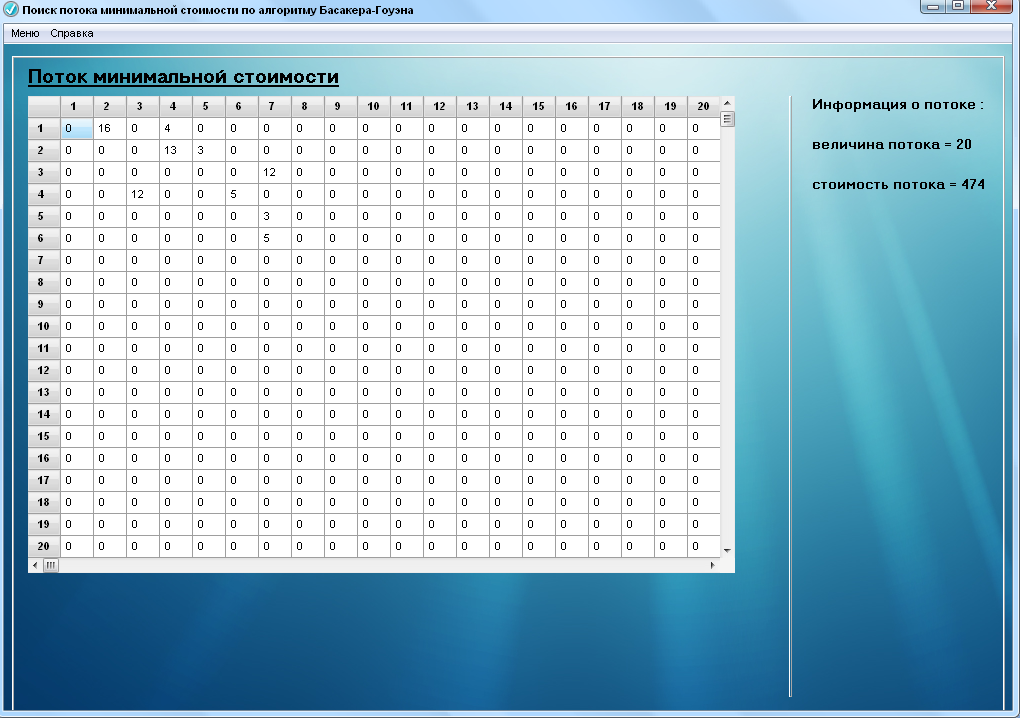


Рисунок 1.5 – Полосы прокрутки

2 Общая часть

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

14

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

2.1 Технология разработки программы

Данная программа разрабатывалась в несколько этапов:

- получение задания на курсовое проектирование

- сбор сведений о поставленной задаче

- выбор реализуемого в программе алгоритма

- определение структуры и типов данных

- написание на выбранном языке алгоритма

- построение интерфейса программы

- отладка и тестирование

- сдача программы

2.2 Описание процесса отладки и испытания программы

Изм

Лист

№ докум.

Подп.

Дата

КР 230105.ММ.00.00.00 ПЗ

Лист

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Процесс отладки сопровождал написание каждой строчки программного кода. Благодаря чему программе не потребовалось существенных переработок кода после каждого выявления случаев некорректной работы программы.

Порядок испытаний проходил в следующем порядке:

- поиск и отладка синтаксических ошибок;

- корректность расчетов проводимых в программе;

- проверка входной информации;

- корректность отображения выходной информации;

- проверка программы на устойчивость к сбоям в работе.

Корректность расчетов, проводимых в программе, проверялась путем расчета исходных данных, вначале вручную, а затем с помощью программы. После чего сравнивались результаты.

Расчет тестового примера:

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

15

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Рассмотрим граф G, изображенный на рисунке 2.1, где первое число пометки у каждой дуги равно ее пропускной способности, а второе – стоимости. Требуется найти поток от s к t минимальной стоимости со значением 20.

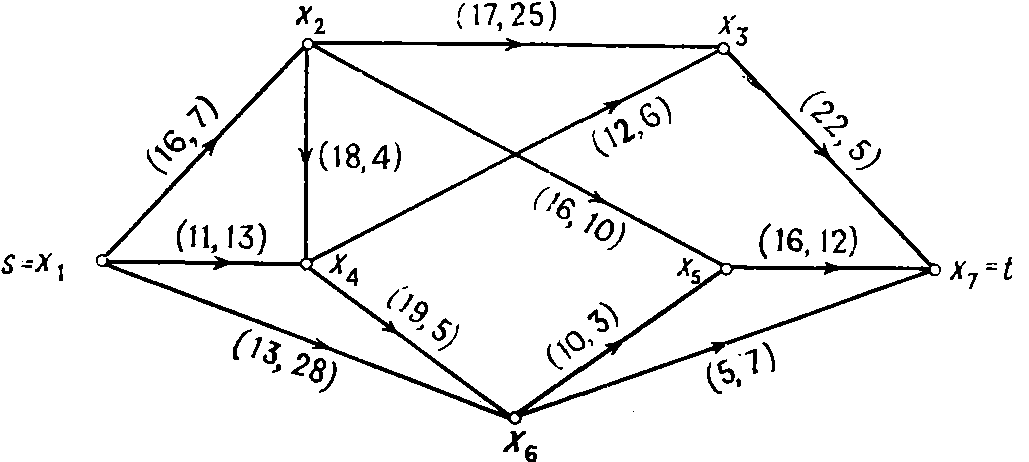


Рисунок 2.1 – Граф G

Для выявления циклов с отрицательной стоимостью на шаге 3 алгоритма Клейна используем алгоритм кратчайшей цепи Флойда.

Шаг 1. Алгоритм максимального потока дает первый допустимый поток, изображенный на рисунке 2.2. Стоимость этого потока равна 732.

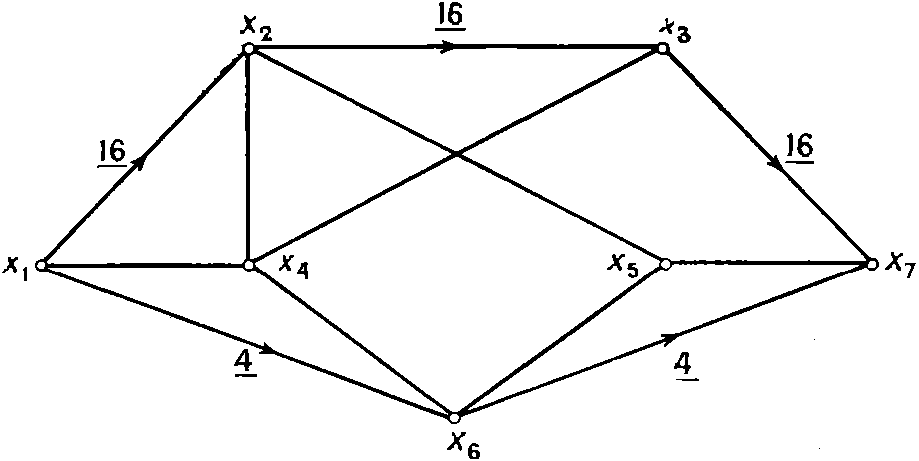


Рисунок 2.2 – Начальный поток

Шаг 2. Исходя из этого потока, построим инкрементальный граф, как показано на рисунке 2.3, где цикл отрицательной стоимости изображен пунктиром.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

16

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

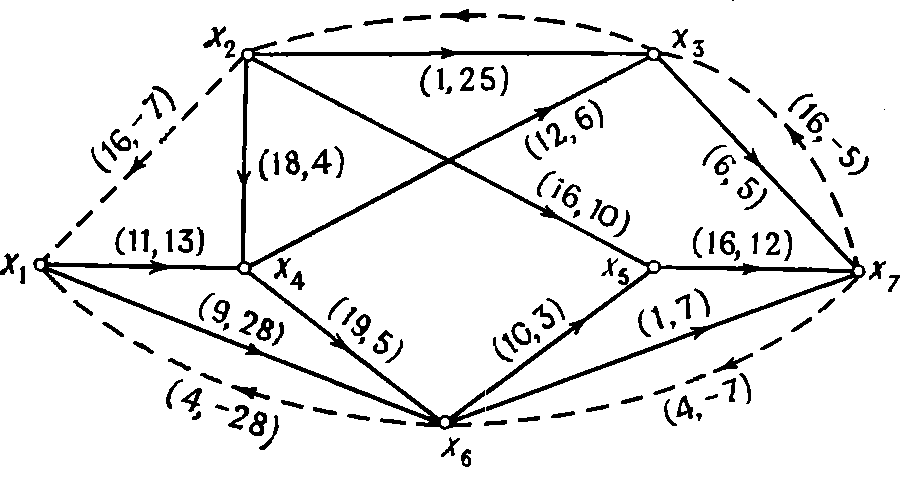


Рисунок 2.3 – Инкрементальный граф

Шаг 3. Начиная с матрицы стоимостей (рис. 2.4) и применяя алгоритм Флойда, получаем следующие матрицы наименьших стоимостей вместе с соответствующими им матрицами цепей. Матрицы после первой итерации приведены на рисунке 2.5, после второй – на рисунке 2.6, после третьей – на рисунке 2.7.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 | x6 | x7 |
| x1 | 0 | ∞ | ∞ | 13 | ∞ | 28 | ∞ |
| x2 | -7 | 0 | 25 | 4 | 10 | ∞ | ∞ |
| x3 | ∞ | -25 | 0 | ∞ | ∞ | ∞ | 5 |
| x4 | ∞ | ∞ | 6 | 0 | ∞ | 5 | ∞ |
| x5 | ∞ | ∞ | ∞ | ∞ | 0 | ∞ | 12 |
| x6 | -28 | ∞ | ∞ | ∞ | 3 | 0 | 7 |
| x7 | ∞ | ∞ | -5 | ∞ | ∞ | -7 | 0 |

Рисунок 2.4 – Начальная матрица стоимостей

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 | x6 | x7 |
| x1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| x2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 |
| x3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| x4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| x5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| x6 | 6 | 6 | 6 | 1 | 6 | 6 | 6 |
| x7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 | x6 | x7 |
| x1 | 0 | ∞ | ∞ | 13 | ∞ | 28 | ∞ |
| x2 | -7 | 0 | 25 | 4 | 10 | 21 | ∞ |
| x3 | ∞ | -25 | 0 | ∞ | ∞ | ∞ | 5 |
| x4 | ∞ | ∞ | 6 | 0 | ∞ | 5 | ∞ |
| x5 | ∞ | ∞ | ∞ | ∞ | 0 | ∞ | 12 |
| x6 | -28 | ∞ | ∞ | -15 | 3 | 0 | 7 |
| x7 | ∞ | ∞ | -5 | ∞ | ∞ | -7 | 0 |

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

17

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Рисунок 2.5 – Матрицы в конце первой итерации

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 | x6 | x7 |
| x1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| x2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 |
| x3 | 2 | 3 | 3 | 2 | 2 | 1 | 3 |
| x4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 |
| x5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| x6 | 6 | 6 | 6 | 1 | 6 | 6 | 6 |
| x7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 | x6 | x7 |
| x1 | 0 | ∞ | ∞ | 13 | ∞ | 28 | ∞ |
| x2 | -7 | 0 | 25 | 4 | 10 | 21 | ∞ |
| x3 | -32 | -25 | 0 | -21 | -15 | -4 | 5 |
| x4 | ∞ | ∞ | 6 | 0 | ∞ | 5 | ∞ |
| x5 | ∞ | ∞ | ∞ | ∞ | 0 | ∞ | 12 |
| x6 | -28 | ∞ | ∞ | -15 | 3 | 0 | 7 |
| x7 | ∞ | ∞ | -5 | ∞ | ∞ | -7 | 0 |

Рисунок 2.6 – Матрицы в конце второй итерации

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 | x6 | x7 |
| x1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| x2 | 2 | 2 | 2 | 2 | 2 | 1 | 3 |
| x3 | 2 | 3 | 3 | 2 | 2 | 1 | 3 |
| x4 | 2 | 3 | 4 | 2 | 2 | 1 | 3 |
| x5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| x6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| x7 | 2 | 3 | 7 | 2 | 2 | 1 | 7 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | x1 | x2 | x3 | x4 | x5 | x6 | x7 |
| x1 | 0 | ∞ | ∞ | 13 | ∞ | 28 | ∞ |
| x2 | -7 | 0 | 25 | 4 | 10 | 21 | 30 |
| x3 | -32 | -25 | 0 | -21 | -15 | -4 | 5 |
| x4 | -26 | -19 | 6 | -15 | -9 | 2 | 11 |
| x5 | ∞ | ∞ | ∞ | ∞ | 0 | ∞ | 12 |
| x6 | -28 | ∞ | ∞ | -15 | 3 | 0 | 7 |
| x7 | -37 | -30 | -5 | -26 | -20 | -9 | 0 |

Рисунок 2.7 – Матрицы в конце третьей итерации

В последней итерации элемент c4, 4 получается отрицательным, со значением -15, которое показывает существование цикла отрицательной стоимости, содержащий вершину x4. Этот цикл, найденный по матрицы цепей, имеет вид (x4, x3, x2, x4).

Шаг 4. По формуле (1.4) находим δ =12.

Новый поток, полученный после введения в цикл циркулярного потока δ, изображен на рисунке 2.8. Стоимость нового потока равна 552. Возвращаясь к шагу 2, находим новый инкрементальный граф, показанный на рисунке 2.9.

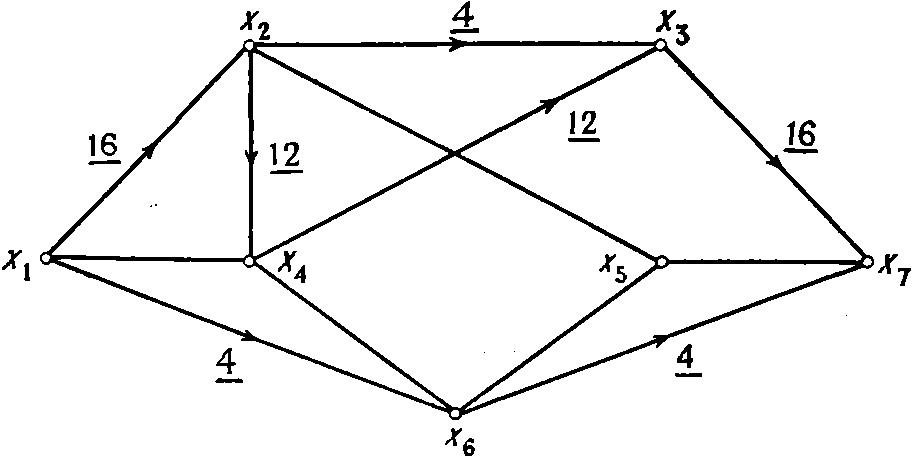


Рисунок 2.8 – Улучшенный поток со стоимостью 552

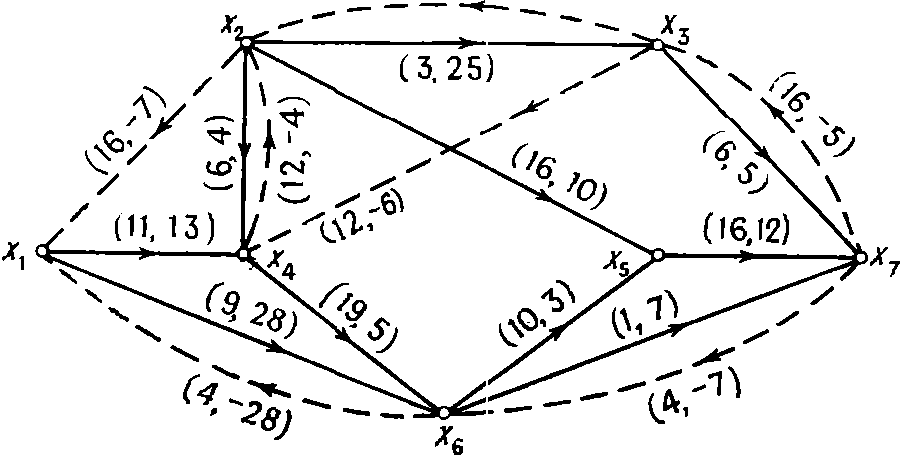


Рисунок 2.9 – Инкрементальный граф для потока на рис. 2.8

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

18

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Шаг 3. Обнаружен отрицательный цикл (x6, x1, x4, x6) со стоимостью -10.

Шаг 4. δ = min [4, 11, 19] = 4.

Новый поток со стоимостью 512 изображен на рисунке 2.10.

Шаг 2. Инкрементальный граф, соответствующий новому потоку, изображен на рисунке 2.11.

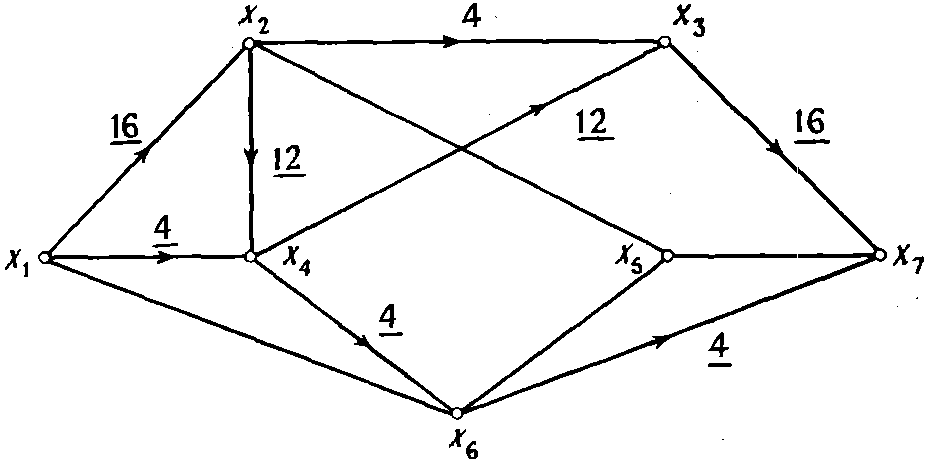


Рисунок 2.10 – Улучшенный поток со стоимостью 512

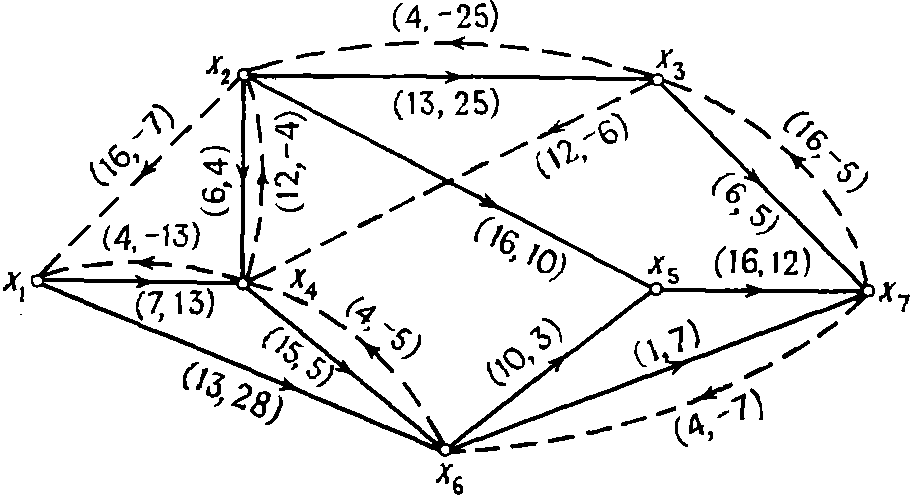


Рисунок 2.11 – Инкрементальный граф для потока на рисунке 2.10

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

19

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Шаг 3. Обнаружен отрицательный цикл (x7, x3, x2, x5, x7) со стоимостью -8

Шаг 4. δ = min [16, 4, 16, 16] = 4.

Новый поток со стоимостью 480 изображен на рисунке 2.12.

Шаг 2. Инкрементальный граф, соответствующий новому потоку, изображен на рисунке 2.13.

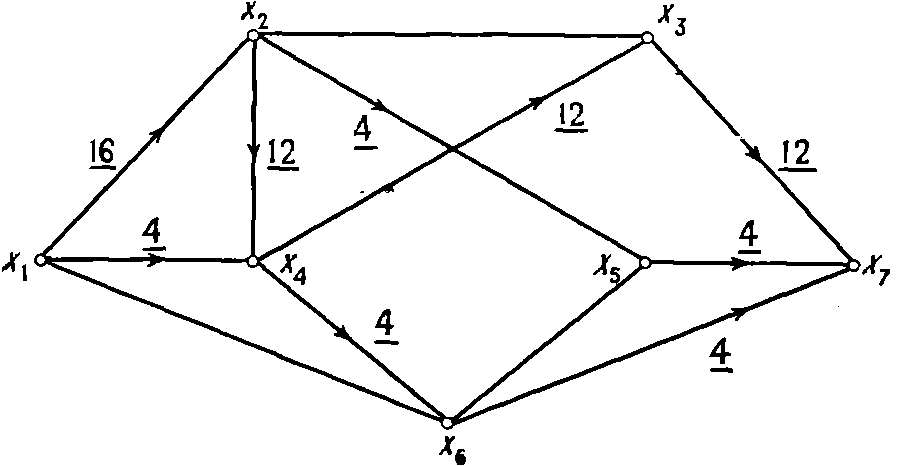


Рисунок 2.12 – Улучшенный поток со стоимостью 480

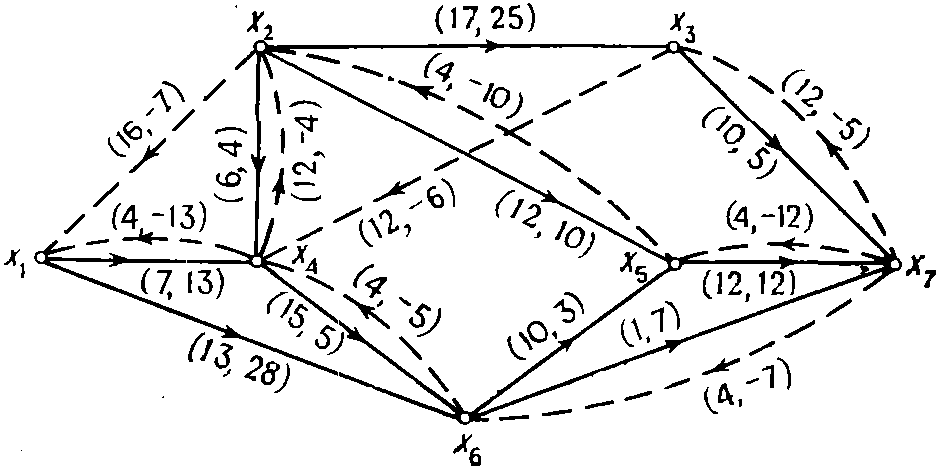


Рисунок 2.13 – Инкрементальный граф для потока на рисунке 2.12

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

20

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Шаг 3. Обнаружен отрицательный цикл (x1, x4, x6, x7, x5, x2, x1) со стоимостью -4.

Шаг 4. δ = min [7, 15, 1, 4, 4, 16] = 1.

Новый поток со стоимостью 476 изображен на рисунке 2.14.

Шаг 2. Инкрементальный граф, соответствующий новому потоку, изображен на рисунке 2.15.

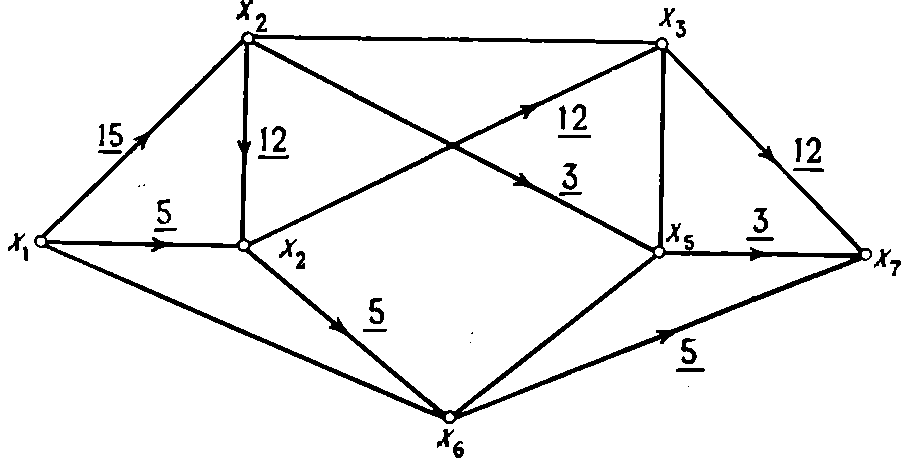


Рисунок 2.14 – Улучшенный поток со стоимостью 476

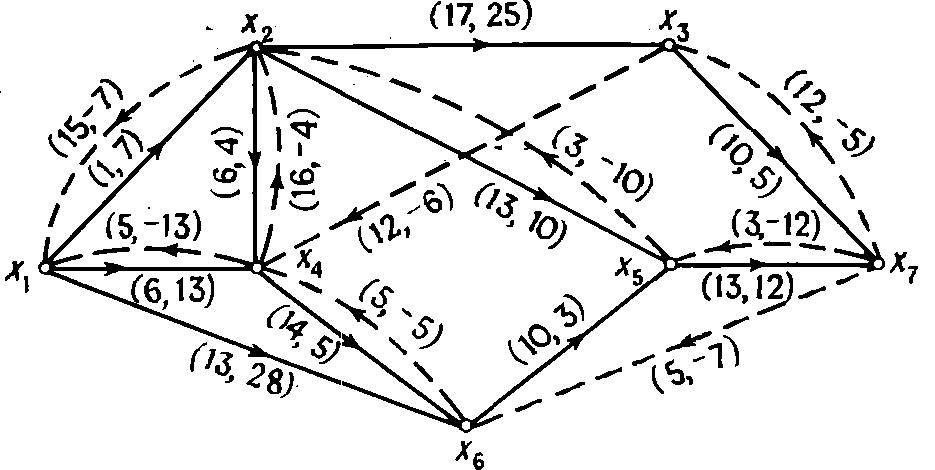


Рисунок 2.15 – Инкрементальный граф для потока на рисунке 2.14

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

21

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Шаг 3. Обнаружен отрицательный цикл (x1, x2, x4, x1) со стоимостью -2.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

22

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Шаг 4. δ = min [1, 6, 5] = 1.

Новый поток со стоимостью 474 изображен на рисунке 2.16.

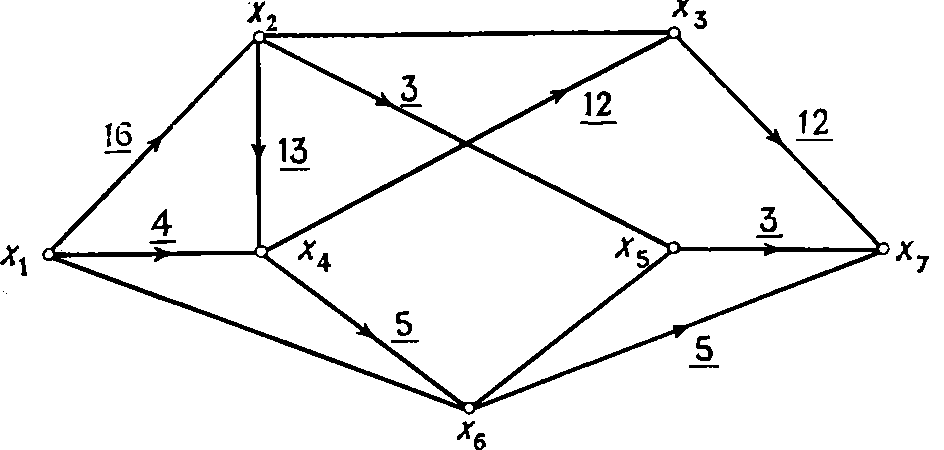


Рисунок 2.16 – Поток минимальной стоимости

В последнем инкрементальном графе нет никаких циклов с отрицательной стоимостью, и поэтому поток, изображенный на рисунке 2.16, является потоком наименьшей стоимости со значением 20 и его стоимость равна 474.

Сравнив результаты работы программы и ответ, полученный в результате решения тестового примера вручную, убеждаемся, что программа работает корректно (рис 2.17). Другие тестовые задачи приведены в приложении Г.

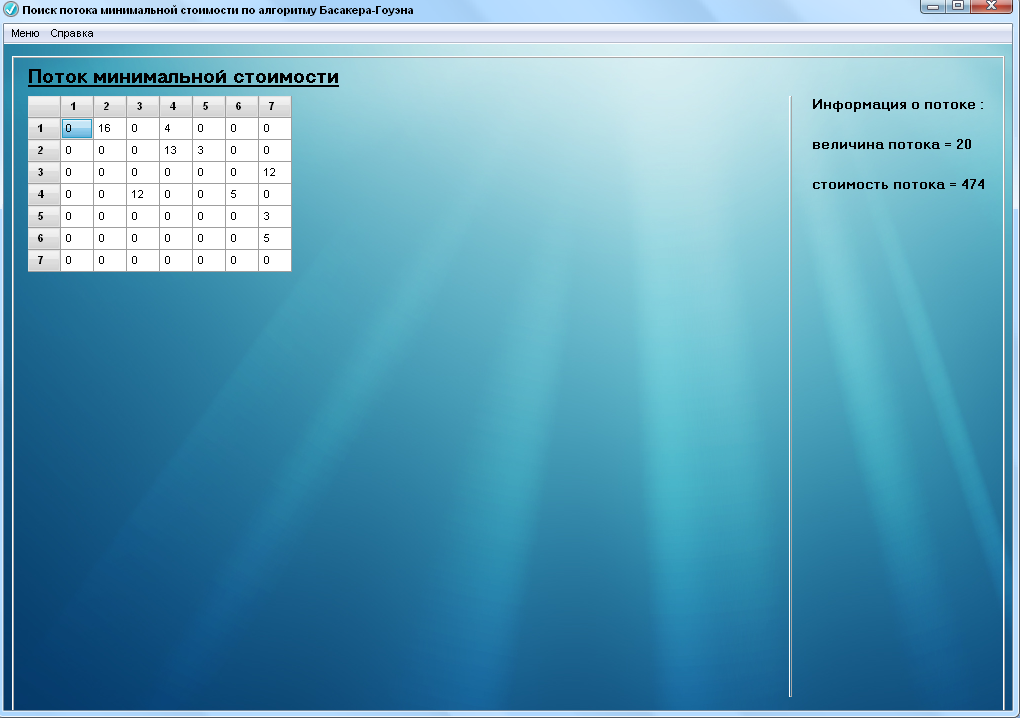


Рисунок 2.17 – Расчет программой тестового примера

2.3 Надежность программного изделия

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

23

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Для надежной работы программы необходимо следующее:

- контроль входных данных, которые вводятся в ЭВМ;

- контроль заполнения всех необходимых полей;

- невозможность изменения пользователем итоговых данных при работе программы.

На стадии проектирования в соответствии с вышеизложенными требованиями, были проведены следующие действия:

- для контроля входных данных были использованы специальные компоненты визуальной разработки, позволяющие ограничить тип данных;

- разработана форма поэтапного ввода всех необходимых данных;

- были изучены возможности языка системы Borland Delphi 7 для легкости доступа к итоговым данным.

На стадии отладки были проверены все реализованные функции на соответствие установленным требованиям.

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

24

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Проверка программы на устойчивость к сбоям в работе или функциональная устойчивость проводилась путем многократных испытаний программы, включающих в себя защиту всех полей ввода информации от символов, букв, а также отрицательных значений (рис. 2.18).

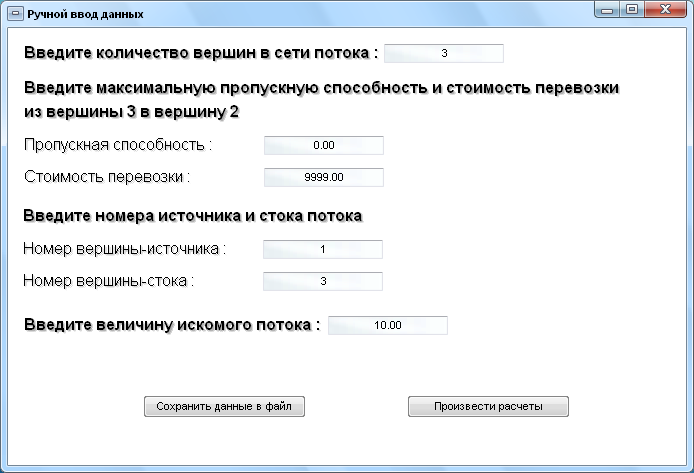
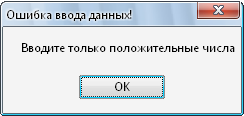


Рисунок 2.18 – Сообщение о некорректном вводе

ЗАКЛЮЧЕНИЕ

В результате курсового проектирования была написана программа «Поиск потока минимальной стоимости», производящая вычисления направленные на автоматизированный поиск определенного потока минимальной стоимости в заданной пользователем сети. Программа представлена в виде исполняемого модуля, имеющего графический интерфейс пользователя, выполненный в соответствии с требованиями, предъявляемыми к графическому интерфейсу программного обеспечения ОС Windows. В комплект с данной программой входит набор тестовых примеров, которые показывают правильность ее функционирования при различных исходных данных.

При разработке программы были закреплены навыки программирования в среде Borland Delphi 7, изучены государственные стандарты по разработке и оформлению программной документации. Были изучены математические методы раздела теории графов «Потоки в сетях».

Таким образом, по итогам проделанной работы можно сделать следующие выводы: в ходе выполнения курсового проекта были систематизированы и закреплены полученные теоретические и практические умения по дисциплине «Математические методы» и другим дисциплинам, была проведена подготовка к итоговой аттестации

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

25

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

ЛИТЕРАТУРА

Изм

Лист

№ докум.

Подп.

Дата

КП 230105.ММ.00.00 ПЗ

Лист

26

Инв. №подл.

Подпись и дата

Взам. инв. №

Инв. № дубл.

Подпись и дата

Основная литература

1. Кристофидес Н. Теория графов. Алгоритмический подход. М.: Мир, 1978. – 427 с.

2. Липский В. Комбинаторика для программистов. М.: Мир, 1988. – 200 с.

3. Майника Э. Алгоритмы оптимизации на графах и сетях. М.: Мир, 1981. – 323 с.

4. Новиков Ф.А. Дискретная математика для программистов. Спб: Питер, 2000.-304 с.

5. Оре О. Теория графов. М.: Наука, 1980. – 336 стр.

6. Пападимитриу Х., Стайглиц К. Комбинаторная оптимизация. Алгоритмы и сложность. М.: Мир, 1985. – 501 с.

7. Свами М., Тхуласираман К. Графы, сети и алгоритмы. М.: Мир, 1984. – 455 с.

8. Форд Л., Фалкерсон Д. Потоки в сетях. М.: Мир, 1966. – 273 с.

9. Ху T. Целочисленное программирование и потоки в сетях. М.: Мир, 1974. – 513 с.

Нормативная литература

1. ГОСТ 19.201-78 – ЕСПД (Единая система программной документации).

2. ГОСТ 19.301-79 – ЕСПД (Единая система программной документации).

3. ГОСТ 19.402-78 – ЕСПД (Единая система программной документации).

ПРИЛОЖЕНИЕ А

Входная информация

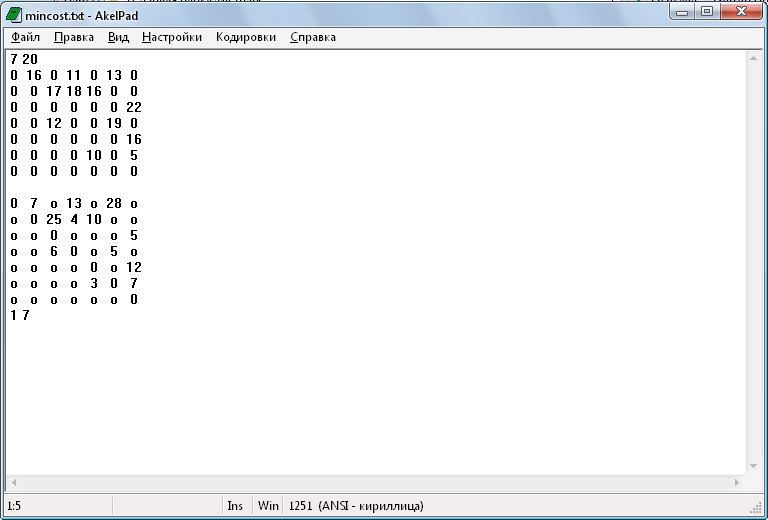


Рисунок А.1 – Входная информация

ПРИЛОЖЕНИЕ Б

Выходная информация

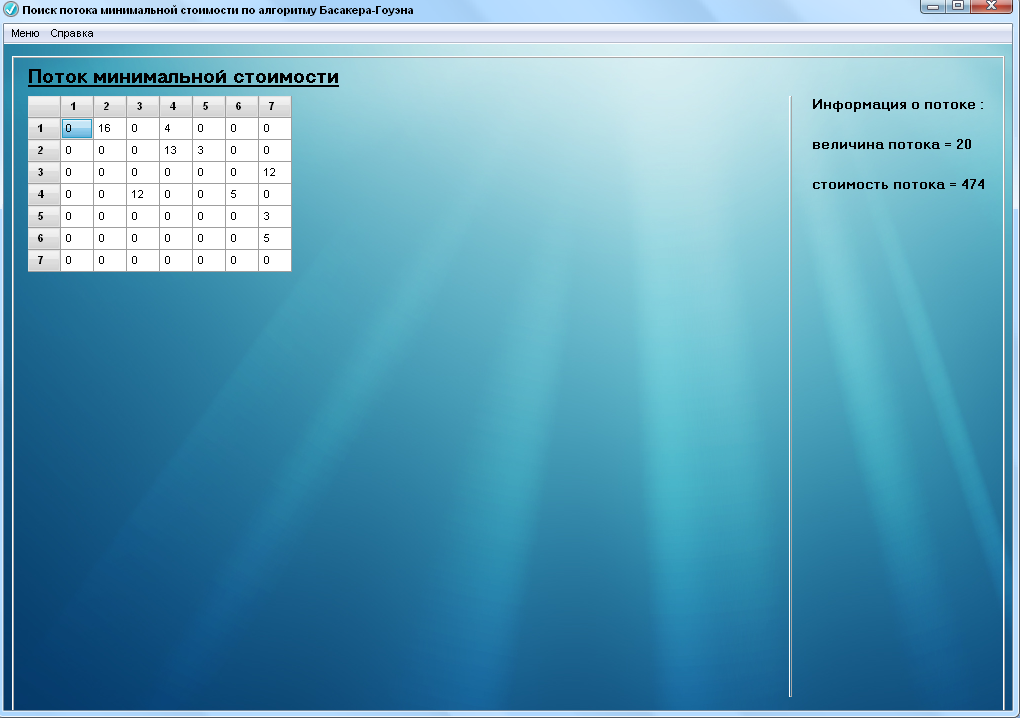


Рисунок Б.1 – Выходная информация

ПРИЛОЖЕНИЕ В

Блок-схема программы

Рисунок В.1 – Блок-схема Main.pas

начало

Ввод массивов данных q[i, j]

c[i, j], количество вершин n,

искомого потока p, номеров

стока ss и истока tt

f[i, j]:=0; x:=0; y:=0

i:=1,n; j:=1,n

f[j, i]>0

cc[i, j]:=-c[j, i]

cc:=c

f[i, j]=q[i, j]

cc[i, j]:=oo

(floyd(n, cc))

and (h[s, t]<>0)

i:=s; j:=h[s, t]; b:=q[i, j]-f[i, j]

i:=j; j:=h[i, t]

d[i, j]>0

a:=f[j, i]

a:=q[i, j]-f[i, j]

b:=min(b, a)

j=t

**+**

**+**

**+**

**-**

**-**

1

3

**+**

2

**-**

**+**

Продолжение приложения В

Продолжение Рисунка В.1

x:=x+b; y:=y+d[s, t]\*b

i:=j; j:=h[i, t]

q[j, i]>0

f[i, j]:=f[i, j]+

b\*sign(d[i, j])

f[j, i]:=f[j, i]+

b\*sign(d[i, j])

j=t

**+**

**-**

**-**

1

x:=p

2

**+**

3

x=p

**-**

**+**

Вывод матрицы потока

минимальной стоимости

f[i, j] и его стоимости y

конец

(x+b)>p

b:=p-x

**+**

j:=s

Окончание приложения В

начало

i:=1,n; j:=1,n

d[i, j]:=c[i, j]

(k<>i)and(d[i, k]<>oo)and

(k<>j)and(d[k, j]<>oo)and

(d[i, j]>(d[i, k]+d[k, j]))

floyd:=true

c[i, j]=oo

h[i, j]:=j

h[i, j]:=0

**+**

**-**

k:=1

i:=1,n; j:=1,n

h[i, j]:=h[i, k];

d[i, j]=d[i, k]+d[k, j]

j:=1,n

d[j, j]<0

floyd:=false; k:=n

k:=k+1

k>n

конец

**-**

**+**

**+**

**+**

Рисунок В.2 – Блок-схема FloydSearch.pas

ПРИЛОЖЕНИЕ Г

Тестовые примеры

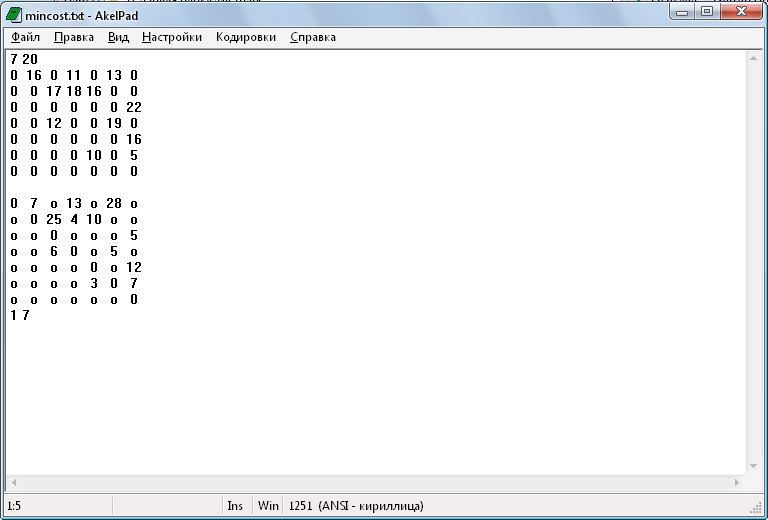


Рисунок Г.1 – Входная информация первого тестового примера

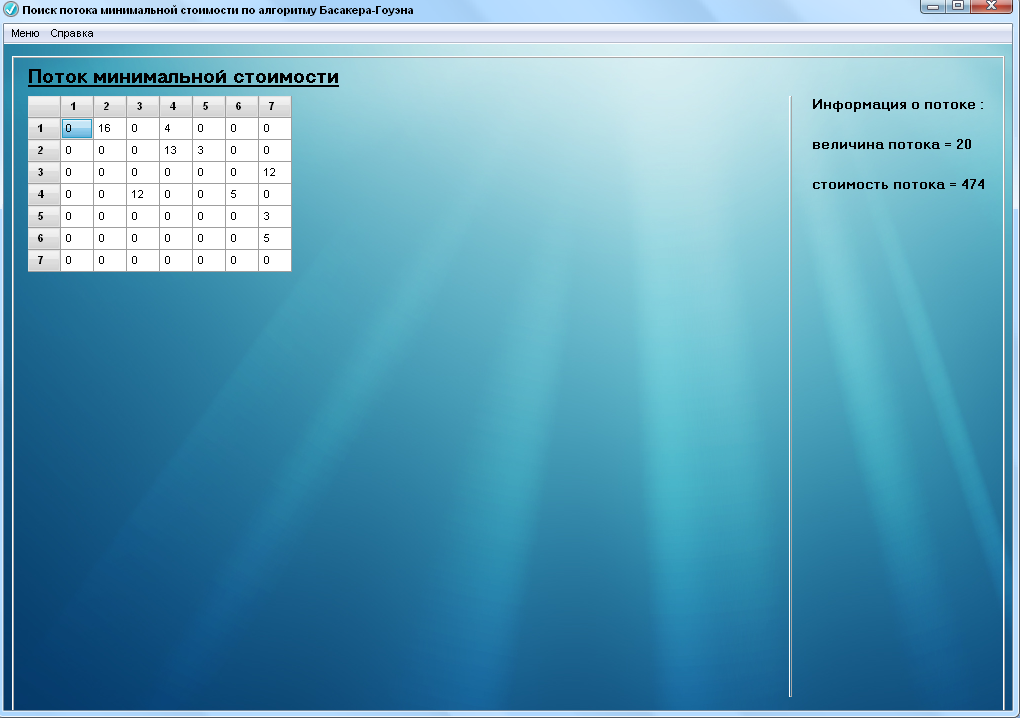


Рисунок Г.2 – Выходная информация первого тестового примера

Продолжение приложения Г

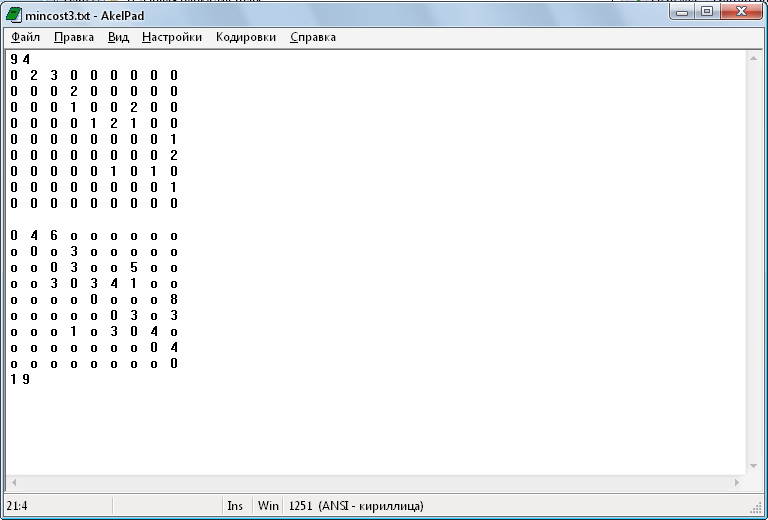


Рисунок Г.3 – Входная информация второго тестового примера

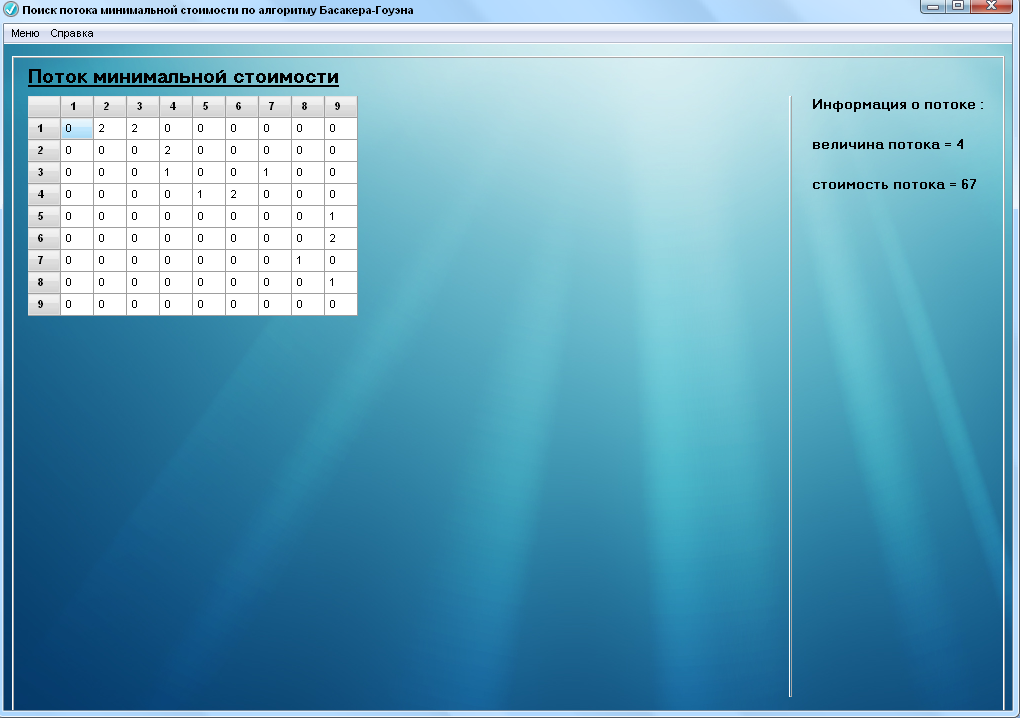


Рисунок Г.4 – Выходная информация второго тестового примера

Окончание приложения Г

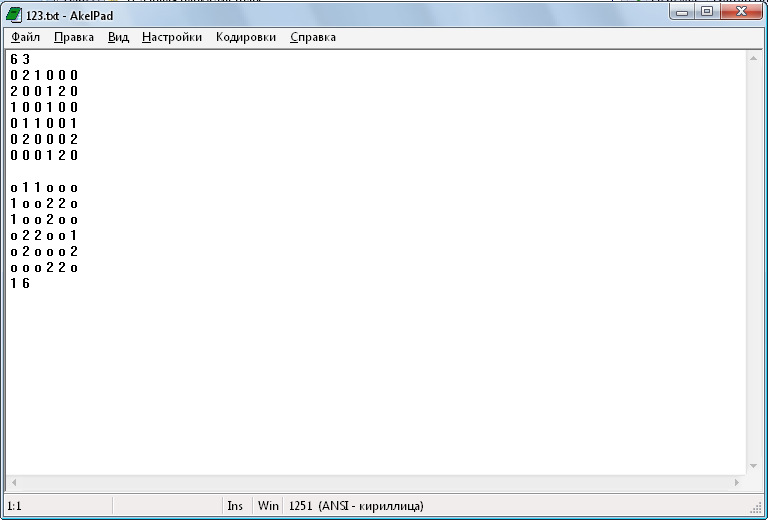


Рисунок Г.5 – Входная информация третьего тестового примера

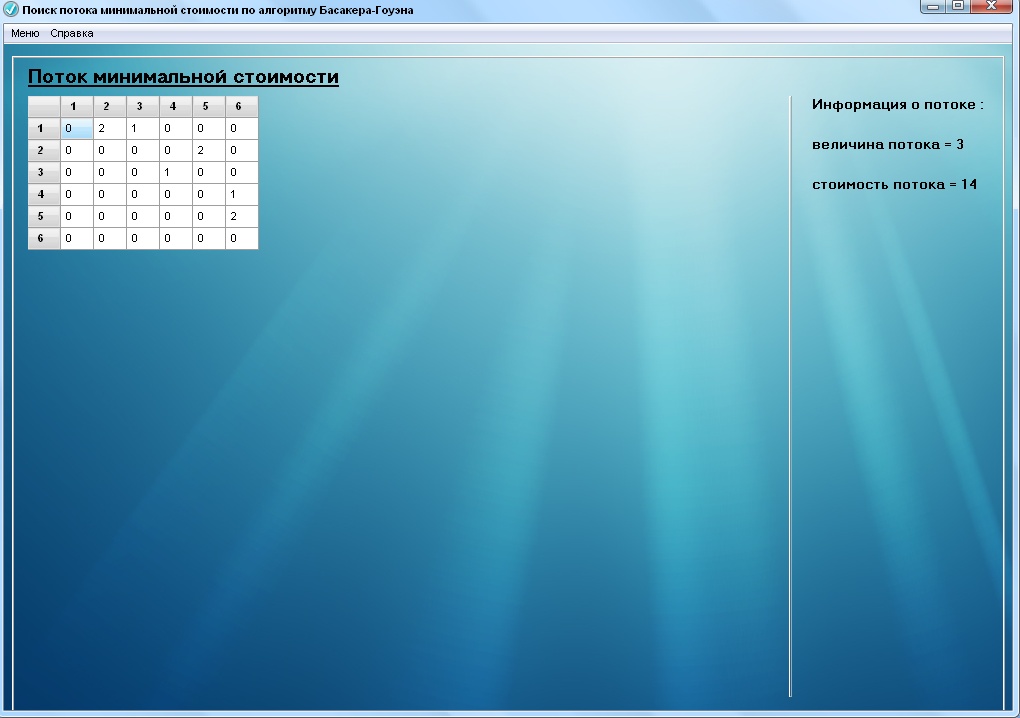


Рисунок Г.6 – Выходная информация третьего тестового примера

ПРИЛОЖЕНИЕ Д

Текст программы

program Project1;

uses Forms,

Main in 'Main.pas' {MainForm},

FloydSearch in 'FloydSearch.pas',

SplashScreen in 'SplashScreen.pas' {Splash},

Input in 'Input.pas' {InputForm};

{$R \*.res}

begin Application.Initialize;

Application.Title:='MinCostFlow';

Application.CreateForm(TMainForm, MainForm);

Application.CreateForm(TSplash, Splash);

Application.Run;

end.

unit Main;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Dialogs,

Forms, Menus, SplashScreen, FloydSearch, Grids, ExtCtrls, StdCtrls, XPMan,

SkinCtrls, DynamicSkinForm, SkinData, SkinGrids, spSkinShellCtrls, Input,

Math;

type TMainForm=class(TForm)

Result: TPanel;

Label1: TLabel;

Label2: TLabel;

Bevel1: TBevel;

Продолжение приложения Д

Label3: TLabel;

Label4: TLabel;

Label5: TLabel;

spDynamicSkinForm1: TspDynamicSkinForm;

spSkinData1: TspSkinData;

spCompressedSkinList1: TspCompressedSkinList;

spSkinMainMenuBar1: TspSkinMainMenuBar;

spSkinMainMenu1: TspSkinMainMenu;

N9: TMenuItem;

N10: TMenuItem;

N11: TMenuItem;

N12: TMenuItem;

N13: TMenuItem;

N14: TMenuItem;

N15: TMenuItem;

N16: TMenuItem;

StringGrid1: TspSkinStringGrid;

spSkinScrollBar1: TspSkinScrollBar;

spSkinScrollBar2: TspSkinScrollBar;

OpenDialog1: TspSkinOpenDialog;

procedure N10Click(Sender: TObject);

procedure N13Click(Sender: TObject);

procedure N16Click(Sender: TObject);

procedure N11Click(Sender: TObject);

procedure FormCreate(Sender: TObject);

procedure pro;

private

public

end;

var txt: text;

Продолжение приложения Д

MainForm: TMainForm;

p, x, y, a, b: extended;

i, j, n, s, t, k, l: integer;

q, c, cc, f: matrix;

st: string;

ch: char;

implementation

{$R \*.dfm}

procedure TMainForm.pro;

begin for i:=1 to n do for j:=1 to n do begin f[i, j]:=0;

if (i=t)or(j=s) then q[i, j]:=0 end;

x:=0; y:=0;

repeat cc:=c;

for i:=1 to n do for j:=1 to n do

if f[j, i]>0 then cc[i, j]:=-c[j, i];

for i:=1 to n do for j:=1 to n do

if (f[i, j]=q[i, j])and(q[i, j]<>0) then cc[i, j]:=oo;

if (floyd(n, cc))and(h[s, t]<>0) then begin

i:=s;

j:=h[s, t];

b:=q[i, j]-f[i, j];

repeat i:=j; j:=h[i, t];

if d[i, j]>0 then a:=q[i, j]-f[i, j] else a:=f[j, i];

b:=min(b, a)

until j=t; j:=s;

if (x+b)>p then b:=p-x;

x:=x+b;

y:=y+d[s, t]\*b;

Продолжение приложения Д

repeat i:=j; j:=h[i, t];

if q[i, j]>0 then

f[i, j]:=f[i, j]+b\*sign(d[i, j]) else

f[j, i]:=f[j, i]+b\*sign(d[i, j]) until j=t;

for i:=2 to n do for j:=1 to i-1 do if (f[i, j]<>0)and(f[j, i]<>0) then begin

f[i, j]:=f[i, j]+f[j, i];

f[j, i]:=f[i, j] end end else x:=p until x=p;

x:=0;

with MainForm do

with StringGrid1 do begin Result.Visible:=true;

ColCount:=n+1;

RowCount:=n+1;

if n<20 then begin

spSkinScrollBar1.Visible:=false;

spSkinScrollBar2.Visible:=false;

Height:=RowCount\*(DefaultRowHeight+GridLineWidth);

Width:=ColCount\*(DefaultColWidth+GridLineWidth) end else begin

spSkinScrollBar1.Visible:=true;

spSkinScrollBar2.Visible:=true;

Width:=692;

Height:=462 end;

for i:=1 to n do begin x:=x+f[1, i];

Cells[i, 0]:=' '+inttostr(i);

Cells[0, i]:=' '+inttostr(i);

for j:=1 to n do Cells[i, j]:=floattostr(f[j, i]) end;

Label3.Caption:='величина потока = '+floattostr(x);

Label4.Caption:='стоимость потока = '+floattostr(y);

if x=p then Label5.Caption:='' else

Label5.Caption:='искомый поток в '+floattostr(p)+#13'единиц не существует,'#13'достигнут максимальный'#13'поток в '+floattostr(x)+' единиц!' end end;

Продолжение приложения Д

procedure TMainForm.N10Click(Sender: TObject);

begin if OpenDialog1.Execute then begin

assignfile(txt, OpenDialog1.FileName);

reset(txt);

readln(txt, n, p); read(txt, ch);

for i:=1 to n do for j:=1 to n do begin st:='';

repeat st:=st+ch;

if eoln(txt) then ch:=' ' else read(txt, ch) until ch=' ';

if i=j then q[i, j]:=0 else

if st[1]='o' then q[i, j]:=oo else q[i, j]:=strtofloat(st);

if eoln(txt) then readln(txt);

if not((i=n)and(j=n)) then repeat read(txt, ch) until ch<>' ' end;

if eoln(txt) then readln(txt); read(txt, ch);

for i:=1 to n do for j:=1 to n do begin st:='';

repeat st:=st+ch;

if eoln(txt) then ch:=' ' else read(txt, ch) until ch=' ';

if i=j then c[i, j]:=0 else

if st[1]='o' then c[i, j]:=oo else c[i, j]:=strtofloat(st);

if eoln(txt) then readln(txt);

if not((i=n)and(j=n)) then repeat read(txt, ch) until ch<>' ' end;

readln(txt, s, t);

closefile(txt);

pro end end;

procedure TMainForm.N13Click(Sender: TObject);

begin close end;

procedure TMainForm.N16Click(Sender: TObject);

begin Splash.Show;

Splash.Update;

Splash.Hide end;

Продолжение приложения Д

procedure TMainForm.N11Click(Sender: TObject);

begin Application.CreateForm(TInputForm, InputForm);

for i:=1 to co do for j:=1 to co do c[i, j]:=oo;

InputForm.spSkinNumericEdit3.Value:=oo;

InputForm.ShowModal;

InputForm.Destroy end;

procedure TMainForm.FormCreate(Sender: TObject);

begin DecimalSeparator:='.' end;

end.

unit FloydSearch;

interface

uses Math;

const co = 100; oo: extended = 9999;

type matrix=array[1..co, 1..co] of extended;

var d: matrix;

i, j, k: integer;

h: array[1..co, 1..co] of integer;

function floyd(n: integer; c: matrix): boolean;

implementation

function floyd(n: integer; c: matrix): boolean;

begin floyd:=true;

for i:=1 to n do for j:=1 to n do begin d[i, j]:=c[i, j];

if c[i, j]=oo then h[i, j]:=0 else h[i, j]:=j end;

k:=1;

repeat for i:=1 to n do for j:=1 to n do

if (k<>i)and(d[i, k]<>oo)and(k<>j)and(d[k, j]<>oo)and

((d[i, j]=oo)or(d[i, j]>(d[i, k]+d[k, j])))then begin h[i, j]:=h[i, k];

Продолжение приложения Д

d[i, j]:=d[i, k]+d[k, j] end;

for j:=1 to n do if d[j, j]<0 then begin floyd:=false;

k:=n end;

k:=k+1 until k>n end;

end.

unit Input;

interface

uses Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,

Dialogs, SkinData, DynamicSkinForm, SkinCtrls, SkinExCtrls, StdCtrls,

Mask, SkinBoxCtrls, ExtCtrls, spSkinShellCtrls, FloydSearch;

type TInputForm = class(TForm)

spDynamicSkinForm1: TspDynamicSkinForm;

spSkinNumericEdit1: TspSkinNumericEdit;

spSkinShadowLabel1: TspSkinShadowLabel;

spSkinButton1: TspSkinButton;

spSkinShadowLabel2: TspSkinShadowLabel;

spSkinShadowLabel3: TspSkinShadowLabel;

spSkinShadowLabel4: TspSkinShadowLabel;

spSkinShadowLabel5: TspSkinShadowLabel;

spSkinNumericEdit2: TspSkinNumericEdit;

spSkinNumericEdit3: TspSkinNumericEdit;

spSkinButton2: TspSkinButton;

Panel1: TPanel;

spSkinButton3: TspSkinButton;

Panel2: TPanel;

spSkinShadowLabel6: TspSkinShadowLabel;

spSkinShadowLabel7: TspSkinShadowLabel;

spSkinShadowLabel8: TspSkinShadowLabel;

Продолжение приложения Д

spSkinButton4: TspSkinButton;

spSkinNumericEdit4: TspSkinNumericEdit;

spSkinNumericEdit5: TspSkinNumericEdit;

spSkinButton5: TspSkinButton;

Panel3: TPanel;

spSkinShadowLabel9: TspSkinShadowLabel;

spSkinNumericEdit6: TspSkinNumericEdit;

spSkinButton6: TspSkinButton;

spSkinButton7: TspSkinButton;

spSkinButton8: TspSkinButton;

SaveDialog1: TspSkinSaveDialog;

procedure spSkinButton1Click(Sender: TObject);

procedure spSkinButton2Click(Sender: TObject);

procedure spSkinButton3Click(Sender: TObject);

procedure spSkinButton4Click(Sender: TObject);

procedure spSkinButton5Click(Sender: TObject);

procedure spSkinButton6Click(Sender: TObject);

procedure spSkinButton7Click(Sender: TObject);

procedure spSkinButton8Click(Sender: TObject);

private

public

end;

var sss: TStrings;

InputForm: TInputForm;

iii, ii, jjj, jj: integer;

implementation

uses Main;

{$R \*.dfm}

Продолжение приложения Д

procedure TInputForm.spSkinButton1Click(Sender: TObject);

begin n:=round(spSkinNumericEdit1.Value);

iii:=1;

jjj:=2;

spSkinButton1.Visible:=false;

Panel1.Visible:=true end;

procedure TInputForm.spSkinButton2Click(Sender: TObject);

begin

q[iii, jjj]:=spSkinNumericEdit2.Value;

c[iii, jjj]:=spSkinNumericEdit3.Value;

if jjj<n then begin inc(jjj);

if (jjj=iii)and(jjj<>n) then inc(jjj) end else if iii<n then begin inc(iii);

jjj:=1 end;

if (jjj=n)and(iii=n) then begin dec(jjj);

Panel2.Visible:=true;

spSkinButton2.Visible:=false;

spSkinButton3.Visible:=false;

spSkinNumericEdit4.Value:=1;

spSkinNumericEdit5.Value:=n end else begin

spSkinNumericEdit2.Value:=q[iii, jjj];

spSkinNumericEdit3.Value:=c[iii, jjj];

spSkinShadowLabel3.Caption:='из вершины '+inttostr(iii)+' в вершину '+inttostr(jjj) end end;

procedure TInputForm.spSkinButton3Click(Sender: TObject);

begin

q[iii, jjj]:=spSkinNumericEdit2.Value;

c[iii, jjj]:=spSkinNumericEdit3.Value;

if jjj>1 then begin dec(jjj);

Продолжение приложения Д

if (jjj=iii)and(jjj<>1) then dec(jjj) end else if iii>1 then begin dec(iii);

jjj:=n end;

if (jjj=1)and(iii=1) then begin inc(jjj);

Panel1.Visible:=false;

spSkinButton1.Visible:=true end else begin

spSkinNumericEdit2.Value:=q[iii, jjj];

spSkinNumericEdit3.Value:=c[iii, jjj];

spSkinShadowLabel3.Caption:='из вершины '+inttostr(iii)+' в вершину '+inttostr(jjj) end end;

procedure TInputForm.spSkinButton4Click(Sender: TObject);

begin spSkinButton4.Visible:=false;

s:=round(spSkinNumericEdit4.Value);

t:=round(spSkinNumericEdit5.Value);

spSkinButton5.Visible:=false;

Panel3.Visible:=true end;

procedure TInputForm.spSkinButton5Click(Sender: TObject);

begin Panel2.Visible:=false;

spSkinButton2.Visible:=true;

spSkinButton3.Visible:=true end;

procedure TInputForm.spSkinButton6Click(Sender: TObject);

begin p:=spSkinNumericEdit6.Value;

spSkinButton6.Visible:=false;

spSkinButton7.Enabled:=true;

spSkinButton8.Enabled:=true end;

procedure TInputForm.spSkinButton7Click(Sender: TObject);

begin sss:=TStringList.Create;

Окончание приложения Д

try sss.Append(inttostr(n)+' '+floattostr(p));

for ii:=1 to n do begin st:='';

for jj:=1 to n-1 do st:=st+floattostr(q[ii, jj])+' ';

st:=st+floattostr(q[ii, n]);

sss.Append(st) end; st:='';

sss.Append(st);

for ii:=1 to n do begin st:='';

for jj:=1 to n-1 do if c[ii, jj]=oo then st:=st+'o ' else

st:=st+floattostr(c[ii, jj])+' ';

if c[ii, n]=oo then st:=st+'o' else st:=st+floattostr(c[ii, n]);

sss.Append(st) end;

sss.Append(inttostr(s)+' '+inttostr(t));

if SaveDialog1.Execute then sss.SaveToFile(SaveDialog1.FileName)

finally sss.Free end end;

procedure TInputForm.spSkinButton8Click(Sender: TObject);

begin MainForm.pro;

close end;

end.