МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ УКРАИНЫ

ХАРКОВСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ РАДИОЭЛЕКТРОНИКИ

Кафедра информатики

***КУРСОВАЯ РАБОТА***

Тема: “Программная реализация алгоритма Дейкстры (построение цепей минимальной длины)”

по дисциплине «Программирование»

# ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Выполнил: Руководители:

Студент группы КИ-05-4 Руденко Д. А.

Петров О. В. Машталир С. В.

Харьков 2006

**РЕФЕРАТ**

Записка объяснительная к курсовой работе: 23с., 5 рис., 1 табл., 5 разделов, 3 приложения.

Объект исследования – граф с взвешенными дугами.

Цель работы – разработка демонстрационной программы использования алгоритма Дейкстры.

Метод исследования – изучение литературы, составление и отладка программы на компьютере.

Данную программу можно использовать для поиска кратчайших расстояний между двумя точками.

Программа составлена на языке С++в среде Microsoft Visual C++ 6.0. Ключевые слова: АЛГОРИТМ ДЕЙКСТРЫ, ГРАФ, ВЕРШИНА, РЕБРО, ВЕС, ПУТЬ, МАССИВ, МЕТКА, ПРОГРАММА, ТИП, ОПЕРАТОР, ФУНКЦИЯ, ЦИКЛ, МАТРИЦА СМЕЖНОСТИ.

**СОДЕРЖАНИЕ**

|  |  |
| --- | --- |
| Введение………………………………………………………....…… | 4 |
| 1 Постановка задачи и сфера её применения…..………………...... | 6 |
| 2 Теоретическая часть…………………………………….………..... | 7 |
| 2.1 Общие сведения о графах……………………………...……. | 7 |
| 2.2 Алгоритм Дейкстры….……………………………………... | 9 |
| 3 Особенности работы в среде ……………………….……………. | 10 |
| 4 Программная реализация………………………………….……. | 11 |
| 4.1 Описание алгоритма и структуры программы…………….. | 11 |
| 4.2 Описание программных средств……………………………. | 13 |
| 5 Инструкция пользователя…………………………………………. | 15 |
| Заключение….…………………………………………………….…. | 16 |
| Перечень ссылок……………………………………………………... | 17 |
| Приложение А Текст программы………………………………….. | 18 |
| Приложение Б Результат………..……………………………….….. | 22 |
| Приложение В Схема программной реализации алгоритма Дейкстры….. | 23 |

**ВВЕДЕНИЕ**

Благодаря своему широкому применению, теория о нахождении кратчайших путей в последнее время интенсивно развивается.

Нахождение кратчайшего пути – жизненно необходимо и используется практически везде, начиная от нахождения оптимального маршрута между двумя объектами на местности (например, кратчайший путь от дома до университета), в системах автопилота, для нахождения оптимального маршрута при перевозках, коммутации информационного пакета в Internet и т.п.

Кратчайший путь рассматривается при помощи некоторого математического объекта, называемого графом.

Существуют три наиболее эффективных алгоритма нахождения кратчайшего пути:

* алгоритм Дейкстры (используется для нахождения оптимального маршрута между двумя вершинами);
* алгоритм Флойда (для нахождения оптимального маршрута между всеми парами вершин);
* алгоритм Йена (для нахождения k-оптимальных маршрутов между двумя вершинами).

Указанные алгоритмы легко выполняются при малом количестве вершин в графе. При увеличении их количества задача поиска кратчайшего пути усложняется. Здесь на помощь приходит современная техника

Компьютерные средства и информационные технологии повысили возможности такого всеохватывающего метода изучения и создания, как моделирования объектов, явлений и процессов – как тех, что существуют в природе, так и тех, что создаются человеком искусственно.

Количество объектов усложнялись, увеличивались, и натурное моделирование (макеты сооружений) стало невыгодным, неэкономным. Поэтому для изучения начали применять математику. Использование математических моделей – уравнения, неравенства, формулы и тому подобное называется математическим моделированием, для развития и приспособления которого нужны были эффективные численные методы.

Реализовать большой потенциал математического моделирования невозможно без мощных средств автоматизации вычислений, которыми являются компьютеры. Благодаря появлению компьютеров и развитию информационных технологий создаются методы и средства компьютерного моделирования, способные решать сложные практические задачи, такие как управление большими энергетическими системами, создание достоверных прогнозов погоды или урожая, моделирование региональных и общегосударственных систем, проектирование самолетов, кораблей и т. п. Компьютерная модель – это размещенная в компьютере совокупность средств, что реализуют концепцию вычисления.

Для реализации компьютерной модели, большое значение имеет такое научное направление, как программирование. Без него компьютер это просто набор различных устройств, микросхем, который не может быть полезным.

Большие программы из-за своей сложности нередко содержат ошибки, которые могут стать причиной материального ущерба, а иногда и угрожать жизни людей (например, при управлении авиаполётами). В результате борьбы с проблемой сложности программного кода были выработаны три новые концепции программирования:

а) объектно-ориентированное программирование (ООП);

б) унифицированный язык моделирования (UML);

в) специализированные средства разработки программного обеспечения;

Из всех объектно-ориентированных языков С++ является наиболее широко используемым. И именно с его помощью в данном курсовом проекте реализуется алгоритм Дейкстры.

**1 ПОСТАНОВКА ЗАДАЧИ И СФЕРА ЕЁ ПРИМЕНЕНИЯ**

Основной задачей данного курсового проекта является программная реализация алгоритма поиска кратчайшего пути между двумя любыми вершинами графа.

Программа должна работать так, чтобы пользователь вводил количество вершин и длины рёбер графа, а после обработки этих данных на экран выводился кратчайший путь между двумя заданными вершинами и его длина. Необходимо предусмотреть различные исходы поиска, чтобы программа не выдавала ошибок и работала правильно.

Данная программа может использоваться в дискретной математике для исследования графов или в качестве наглядного пособия, демонстрирующего применение алгоритма Дейкстры на практике.

**2 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ**

2.1 Сведения о графах

Граф G (рис.2.1.1) задается множеством точек (вершин) х1, х2,..., хn. (которое обозначается через Х) и множеством линий (ребер) а1, а2,...,аm. (которое обозначается символом А), соединяющих между собой все или часть этих точек. Таким образом, граф G полностью задается (и обозначается) парой (Х, А). Если ребра из множества А ориентированы, что обычно показывается стрелкой, то они называются дугами, и граф с такими ребрами называется ориентированным графом.

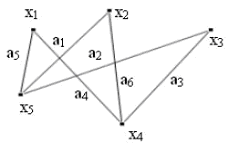
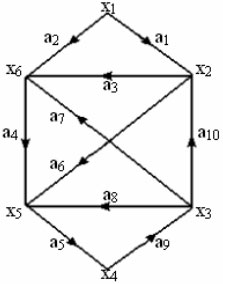
 

рис.2.1.2

рис.2.1.1

Например, если дорога имеет не двух-, а одностороннее движение то направление этого движения будет показано стрелкой.

Если ребра не имеют ориентации, то граф называется неориентированным, (двухстороннее движение).

В ориентированном графе дуга обозначается упорядоченной парой, состоящей из начальной и конечной вершин, ее направление предполагается заданным от первой вершины ко второй.

Путем (или ориентированным маршрутом) ориентированного графа называется последовательность дуг, в которой конечная вершина всякой дуги, отличной от последней, является начальной вершиной следующей.

Так, на рис. 2.1.2 путями являются последовательности дуг:

а6, а5, а9, а8, а4. (1)

а1, а6, а5, а9. (2)

а1, а6, а5, а9, а10, а6, а4. (3)

Ориентированной цепью (орцепью) называется такой путь, в котором каждая дуга используется не больше одного раза.

Простой орцепью называется такой путь, в котором каждая вершина используется не более одного раза. Например, путь (2).

Маршрут есть неориентированный “двойник” пути, и это понятие рассматривается в тех случаях, когда можно пренебречь направленностью дуг в графе. Таким образом, маршрут есть последовательность ребер ä1, ä2,..., äq, в которой каждое ребро аi, за исключением первого и последнего ребер, связано с ребрами аi-1 и аi+1 своими концевыми вершинами. Последовательности дуг:

ä2, ä4, ä8, ä10, (4)

ä2, ä7, ä8, ä4, ä3, (5)

ä10 , ä7 , ä4 , ä8 , ä7 , ä2. (6)

в графе, изображенном на рис. 2.1.2, являются маршрутами; две точки над символом дуги означают, что ее ориентацией пренебрегают, т.е. дуга рассматривается как неориентированное ребро. Также путь или маршрут можно изображать последовательностью вершин. Например, путь (1) будет выглядеть следующем образом: х2, х5, х4, х3, х5, х6. Иногда дугам графа приписываются числа, называемыевесом, стоимостью, или длиной этой дуги. В этом случае граф называется графом с взвешенными дугами*.* А если вес приписывается вершинам графа, то тогда получается граф с взвешенными вершинами. Если в графе веса приписаны и дугам и вершинам, то он называется просто взвешенным*.* При рассмотрении пути µ представленного последовательностью дуг (ä1, ä2,..., äq), за его вес принимается число l(µ), равное сумме весов всех дуг, входящих в µ, т.е.



**2.2 Алгоритм Дейкстры**

Алгоритм Дейкстры строит кратчайшие пути, ведущие из исходной вершины графа к остальным вершинам этого графа (если таковые имеются).

В процессе работы алгоритма последовательно помечаются рассмотренные вершины графа. Причем вершина, помеченная последней (на данный момент) расположена ближе к исходной вершине, чем все непомеченные, но дальше, чем все помеченные.

Сначала помечается исходная вершина; следующей, очевидно, будет помечена вершина, ближайшая к исходной, и смежная с ней.

Пусть на каком-то шаге уже помечено несколько вершин. Известны кратчайшие пути, ведущие из исходной вершины к помеченным. Для каждой из непомеченных вершин проделаем следующее:

1. Рассмотрим все дуги, ведущие из помеченных вершин в одну непомеченную. Каждая такая дуга является последней дугой на пути из исходной вершины в эту непомеченную.
2. Выберем из этих путей кратчайший. А затем выберем среди них самый короткий ко всем непомеченным вершинам, и пометим вершину, к которой он ведет.

Алгоритм завершится, когда будут помечены все достижимые вершины.

В результате работы алгоритма Дейкстры строится Дерево кратчайших путей.

**3 ОСОБЕННОСТИ РАБОТЫ В СРЕДЕ**

При написании программы использовалась среда Microsoft Visual C++ 6.0. Данная среда позволяет писать приложения на языке C++.

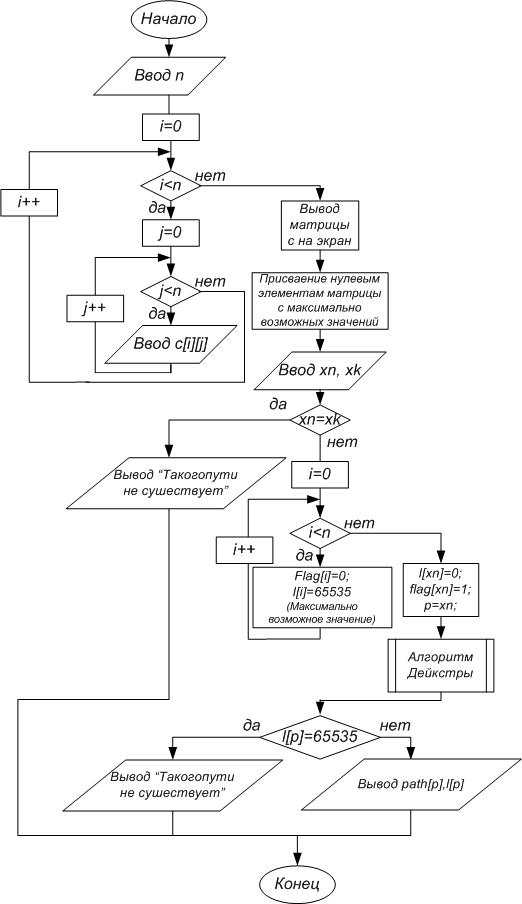
В ходе написания программы все операторы и служебные слова языка С++ выделяются другим цветом, чтобы отличать их от переменных, заданных программистом. Среда Microsoft Visual C++ 6.0 содержит встроенный компилятор.

Окно программы разделено на несколько частей. Вверху находится стандартная панель – Standart, с которой можно сохранить исходный текст программы на диск, открыть новый документ, скопировать или вставить текст, отменить последнее действие, или найти текст. Слева находятся панели Object TreeView и Object Inspector, на которых показаны объекты, которые используются в данной программе, и их свойства. В центре окна программы расположен текстовый редактор, в котором следует писать программу. Внизу – панель Output, в которой показывается сообщения, если в программе содержатся ошибки – компилятор сообщает вид ошибки и строку, в которой она допущена, достаточно сделать двойной клик левой клавишей мыши на описании ошибки в Output, чтобы переместиться на строку, содержащую ошибки.

Программа создана в консольном режиме – в режиме, не имеющем графического интерфейса.

**4 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ**

**4.1 Описание алгоритма и структуры программы**



**Рис. 4.1.1**

Программа выводит минимальный путь между двумя указанными вершинами в графе и его длину.

При запуске программы на экран выводится запрос о вводе весов рёбер исследуемого графа. Данные, введённые пользователем, отображаются в виде матрицы смежности, в которой не существующие рёбра обозначаются нулями. После указанным рёбрам присваивается значение 65535, которое принимается за бесконечность.

Следующим этапом выполнения программы является запрос о вводе номеров вершин, между которыми необходимо узнать путь. В случае, если начальная и конечная вершины совпадают, отображается соответствующее сообщение и работа программы завершается. В противном случае выполняется непосредственно алгоритм Дейкстры, схема которого приведена в приложении В.

Результатом программы является вывод на экран вершин, через которые проходит минимальный путь, а также вывод длины маршрута. Если пути между заданными точками не существует – выводится соответствующее сообщение.

**4.2 Описание использованных программных средств**

Таблица 4.2.1–Описание переменных

|  |  |  |
| --- | --- | --- |
| Переменная | Тип | Описание |
| n | int | Количество точек (вершин) грифа |
| i,j | int | Счётчики |
| p | int | Номер кратчайшего пути и наименьшей длины пути |
| xn | int | Номер начальной точки (вершины) |
| xk | int | Номер конечной точки (вершины) |
| flag[11] | int | Массив, i-й элемент которого имеет значение 0, когда i-й путь и расстояние временные, и принимает значение 1, когда i-й путь и расстояние становятся постоянными |
| c[11][11] | word (unsigned int) | Массив i-j элемент которого содержит расстояние между i-й и j-й точками (вершинами)  Замечание:   1. с[i][i]=∞ 2. c[i][j]=c[j][i] |
| s[80] | char | Строчная переменная, которая содержит промежуточные значения пути |
| path[80][11] | char | Массив строк, который содержит пути  Замечание:  После прохождения обработки по алгоритму Дейкстры p-й элемент массива содержит кратчайший путь. |
| l[11] | word (unsigned int) | Массив, который содержит длины путей (path)  Замечание:  После прохождения обработки по алгоритму Дейкстры p-й элемент массива содержит длину кратчайшего пути. |

Кроме стандартных функций из библиотек iostream.h, string.h, stdio.h, conio.h были использованы также следующие функции.

* word minim(word x, word y) – функция, которая возвращает минимальное из x и y.

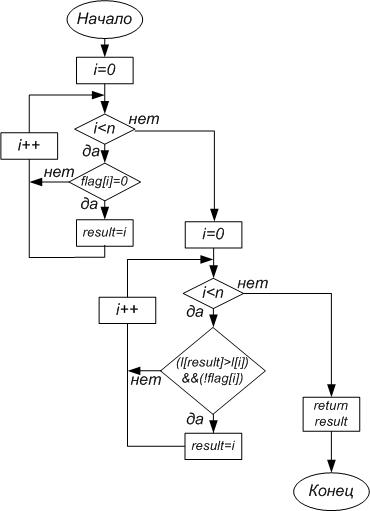


Рис. 4.2.1

* int min(int n) – функция, которая возвращает номер элемента массива l[i] минимальной «неотмеченной» длиной пути(flag[i]=0).

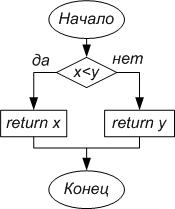


Рис. 4.2.2

**5 ИНСТРУКЦИЯ ПОЛЬЗОВАТЕЛЯ**

При запуске программы на экране появится окно с инструкциями. Выполняйте эти инструкции, а именно:

1. Введите количество вершин исследуемого графа.
2. Введите веса рёбер (положительное число). В программе расстояния от хi до xi+1 и xi+1 до хi считаются равными, а расстояния от хi до хi – не существующими. Если ребра между указанными вершинами не существует, введите 0 (ноль).

На экран выводится матрица смежности, отображающая введённую информацию.

1. Введите номер вершины, от которой начинается искомый путь.
2. Введите номер вершины, в которой путь заканчивается.
3. Чтоб завершить работу программы после получения результата нажмите Enter.

**ЗАКЛЮЧЕНИЕ**

Таким образом, в процессе создания данного проекта разработана программа, реализующая алгоритм Дейкстры в Microsoft Visual C++ 6.0. Её недостатком является примитивный пользовательский интерфейс. Это связано с тем, что программа работает в консольном режиме, не добавляющем к сложности языка сложность программного оконного интерфейса

Также были углублены знания, полученные в процессе выполнения лабораторных работ по предмету «Программирование».

**ПЕРЕЧЕНЬ ССЫЛОК**

1. Бондарев В.М. Программирование на С++.–Х: «Компания СМИТ», 2004
2. Страуструп Бьярн Язык программирования С++(2 ч).–«К:ДиаСофт», 1993
3. Хаханов В.И., Чумаченко С.В. Дискретная математика (теоретическое и практическое содержание курса).–Кафедра АПВТ, 2002
4. Алгоритм Дейкстры
5. Конспект лекций.

**Приложение А**

Текст программы

#include<iostream.h>

#include<string.h>

#include<stdio.h>

#include<stdlib.h>

#include<conio.h>

#define word unsigned int

int i, j, n, p, xn, xk;

int flag[11];

word c[11][11], l[11];

char s[80], path[80][11];

int min(int n)

{

int i, result;

for(i=0;i<n;i++)

if(!(flag[i])) result=i;

for(i=0;i<n;i++)

if((l[result]>l[i])&&(!flag[i])) result=i;

return result;

}

word minim(word x, word y)

{

if(x<y) return x;

return y;

}

void main()

{

cout<<"Vvedite kolichestvo tochek: ";

cin>>n;

for(i=0;i<n;i++)

for(j=0;j<n;j++) c[i][j]=0;

for(i=0;i<n;i++)

for(j=i+1;j<n;j++)

{

cout<<"Vvedite rasstoyanie ot x"<<i+1<<" do x"<<j+1<<": ";

cin>>c[i][j];

}

cout<<" ";

for(i=0;i<n;i++) cout<<" X"<<i+1;

cout<<endl<<endl;

for(i=0;i<n;i++)

{

printf("X%d",i+1);

for(j=0;j<n;j++)

{

printf("%6d",c[i][j]);

c[j][i]=c[i][j];

}

printf("\n\n");

}

for(i=0;i<n;i++)

for(j=0;j<n;j++)

if(c[i][j]==0) c[i][j]=65535; //бесконечность

cout<<"Vvedite nachalnuy tochku: ";

cin>>xn;

cout<<"Vvedite konechnuy tochku: ";

cin>>xk;

xk--;

xn--;

if(xn==xk)

{

cout<<"Nachalnaya I konechnaya tochki sovpadayt."<<endl;

getch();

return;

}

for(i=0;i<n;i++)

{

flag[i]=0;

l[i]=65535;

}

l[xn]=0;

flag[xn]=1;

p=xn;

itoa(xn+1,s,10);

for(i=1;i<=n;i++)

{

strcpy(path[i],"X");

strcat(path[i],s);

}

do

{

for(i=0;i<n;i++)

if((c[p][i]!=65535)&&(!flag[i])&&(i!=p))

{

if(l[i]>l[p]+c[p][i])

{

itoa(i+1,s,10);

strcpy(path[i+1],path[p+1]);

strcat(path[i+1],"-X");

strcat(path[i+1],s);

}

l[i]=minim(l[i],l[p]+c[p][i]);

}

p=min(n);

flag[p]=1;

}

while(p!=xk);

if(l[p]!=65535)

{

cout<<"Put: "<<path[p+1]<<endl;

cout<<"Dlina puti: "<<l[p]<<endl;

}

else

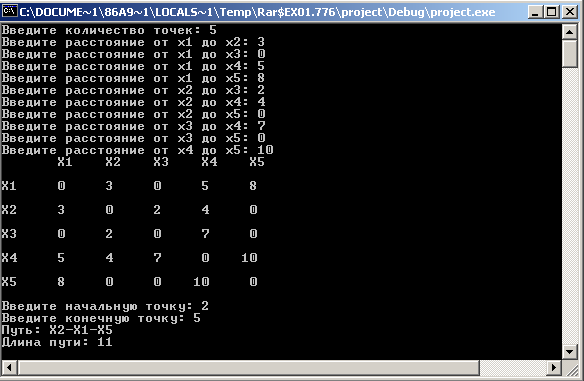
cout<<"takogo puti ne syshestvuet!"<<endl;

getch();

}

**Приложение Б**

Результат



**Приложение В**

Схема программной реализации алгоритма Дейкстры

