**МИНОБРНАУКИ РОССИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**

**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**

**(БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

дисциплина: Дискретная математика

Лабораторная работа № 4.4

тема: «Кратчайшие пути во взвешенном орграфе»

|  |  |
| --- | --- |
|  | Выполнил: ст. группы ПВ-21  Зановская А. И.  Проверил: Рязанов Ю.Д. |

Белгород

2017

**Цель занятия**: изучить алгоритм Дейкстры нахождения кратчайших путей между вершинами взвешенного орграфа, научиться рационально использовать его при решении различных задач.

**Вариант 5**

Найти кратчайший путь во взвешенном орграфе от вершины x до вершины y, проходящий сначала через вершину v, а затем — через вершину w. Вывести найденный путь и его длину.

**Задания**

1. Изучить алгоритм Дейкстры нахождения кратчайших путей между вершинами взвешенного орграфа.

2. Используя алгоритм Дейкстры, разработать и реализовать алгоритм решения задачи (см. варианты заданий).

3. Подобрать тестовые данные. Результат представить в виде диаграммы графа.

#include <stdio.h>

#include <stdlib.h>

#define MAX 1000

#define N 6

void Graph(int g[][N])

{

FILE \*f;

f=fopen("1.txt","r");

int i,j;

while(!feof(f))

{

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

{

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

{

fscanf(f,"%i",&g[i][j]);

printf("%i ",g[i][j]);

}

printf("\n");

}

}

printf("\n");

fclose(f);

}

void derevo(int V[], int T[], int v1, int v2)

{// printf ("\n\n");

int i,j,k;//printf ("%d %d \n", v1, v2);

while(v1!=v2) //пока не дошли до заданной вершины

{

j=0;

for(i=0; i<N, j==0; i++) //j - не обнаружена нужная вершина

{

if(T[i]==v1 && V[i]==1)

{

printf("%i->%i ", v1, i+1);

j=i;

}

}

v1=j+1;

if (v1==2 )//&& T[v2-1]==N)

{

printf("%i->%i ", j, v2);

return;

}

}

}

int deikstr2(int v1, int v2, int g[][N], int T[], int V[], int \*dl)

{

//v1 v2 - начальная и конечная вершины, g - граф

//V - множество вершин орграфа

// Т - массив для хранения дерева кратчайших путей

//D = d(xi) - кратчайшее расстояние от вершины v1 к вершине xi

int min,f,i, tmp=(\*dl);

int D[N]; // минимальное расстояние

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

{

D[i]=MAX;

V[i]=0;

T[i]=-1;

}

D[v1]=0;

V[v1]=1;

T[v1]=0;

while(v1!=v2 && f)

{

f=0;

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

{

if(g[v1][i] && D[v1]+g[v1][i]<D[i])

{

D[i]=D[v1]+g[v1][i];

T[i]=v1+1;

}

}

min=MAX;

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

if(V[i]==0 && min>D[i])

{

min=D[i];

v1=i;

f=1;

}

V[v1]=1;

}

int j;

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

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

if(D[i] == D[j] && D[i] != MAX)

return 0;

tmp+=min;

(\*dl)=tmp;

return 1;

}

int main()

{

int g[N][N];

Graph(g);

int T[N];

int V[N];

int i,v1,z,v2,w, dl=0;

printf("vvod v1-z-w-v2\n");

scanf("%i%i%i%i",&v1,&z,&w,&v2);

printf("\n");

//от первой до z

if(!deikstr2(v1-1,z-1,g,T,V, &dl))

{

printf ("\nnot found");

return 0;

}

printf ("\n");

derevo(V,T,v1,z);

//от z до w

if(!deikstr2(z-1,w-1,g,T,V, &dl))

{

printf ("\nnot found");

return 0;

}

printf ("\n");

derevo(V,T,z,w);

//от w до последней

if(!deikstr2(w-1,v2-1,g,T,V, &dl))

{

printf ("\nnot found");

return 0;

}

printf ("\n");

derevo(V,T,w,v2);

printf("%i", dl);

return 0;

}

**Результаты работы программы**





