**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 №4.6

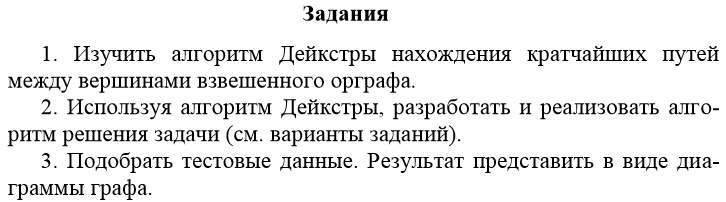
тема: «Кратчайшие пути во взвешенном орграфе»

|  |  |
| --- | --- |
|  | Выполнил: ст. группы ПВ-21  Донцов Александр Алексеевич  Проверил: Рязанов Ю.Д. |

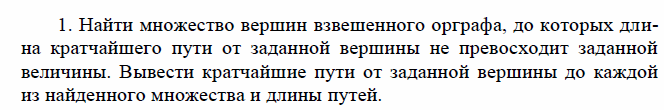
Белгород

2018

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



**Вариант 1**



**Код программы:**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include "stdio.h"

#include "windows.h"

#include <iostream>

int\*\* init\_matr(int n);

void output\_matr(int \*\*a, int n);

int\* djkstra(int \*\*a, int n);

void output(int \*rez, int n);

void out(int \*rez, int n, int num);

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int n = 5;

int \*\*a = init\_matr(n);

output\_matr(a, n);

int \*rez = djkstra(a, n);

output(rez, n);

free(rez);

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

free(a[i]);

free(a);

getchar();

getchar();

getchar();

return 0;

}

// Инициализация матрицы связей

int\*\* init\_matr(int n) {

int temp;

int \*\*a = (int\*\*)malloc(sizeof(int\*) \* n);

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

a[i] = (int\*)malloc(sizeof(int) \* n);

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

scanf("%d", &temp);

a[i][j] = temp;

}

}

return a;

}

// Вывод матрицы смежности

void output\_matr(int \*\*a, int n) {

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

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

printf("%5d ", a[i][j]);

printf("\n");

}

}

//алгоритм Дейкстры

int\* djkstra(int \*\*a, int n) {

int\* rez = (int\*)malloc(sizeof(int) \* n \* 2);

int start;//станция отправки

int u;//самый минимальный не проверенный

int min;//сохранение минимального значения

int \*was = (int\*)malloc(sizeof(int)\*n);//пройденные станции

for (int i = 0; i < n; i++) {//создание пустых массивов данных

rez[i] = INT\_MAX;//установка максимального значения

rez[n + i] = -1;//пустое значение для предыдущей станции

was[i] = 0;//установка ложного значения

}

printf("Введите станцию отправки: ");

scanf("%d", &start);//введение станции отпавки

//start -= 1;//установка на массив

rez[start] = 0;

for (int count = 0; count < n; count++) {

min = \_CRT\_INT\_MAX;//максимальное значение инт

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

if (!was[i] && (rez[i] <= min)) {

min = rez[i];

u = i;//Ищем значение старта

}

was[u] = 1;

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

if (!was[i] && a[u][i] != 0 && rez[u] != INT\_MAX && (rez[u] + a[u][i] < rez[i])) {

rez[i] = rez[u] + a[u][i];

rez[i + n] = u;

}

}

}

free(was);

return rez;

}

//вывод пути

void output(int \*rez, int n) {

int end;

int flag = 0;

printf("Введите длинну пути: ");

scanf("%d", &end);

//end -= 1;

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

if (rez[i] <= end && rez[i] != 0) {

out(rez, n, i);

printf(" = %d\n", rez[i]);

flag = 1;

}

if (!flag)

printf("Путей нет");

}

//рекурсия

void out(int \*rez, int n, int num) {

if (num == -1)

return;

else {

out(rez, n, rez[n + num]);

printf("%d->", num);

}

}

**Тестовые данные**

