

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФГБОУ ВО «Брянский государственный технический университет»

Факультет информационных технологий.

Кафедра «Информатика и программное обеспечение»

Отчет по лабораторной работе

«Алгоритмы на нагруженных графах»

Выполнил:

студент гр. О-18-ПРИ-РПС-Б

Подгорняк А.А.

Проверил:

к.т.н., проф. В. К. Гулаков

Брянск 2020

**СОДЕРЖАНИЕ**

[1. ЗАДАНИЕ 3](#_Toc43895288)

[2. Описание программы 4](#_Toc43895289)

[3. Демонстрация программы 7](#_Toc43895290)

[4. ПРИЛОЖЕНИЕ 8](#_Toc43895291)

## ЗАДАНИЕ

Рассмотрим транспортную сеть, включающую мосты. Будем предполагать, что при превышении, грузоподъемности некоторого моста последний разрушается. Определить максимальный вес груза. который может быть транспортирован в рассматриваемой сети из пункта А в пункт В без превышения грузоподъемности находящихся на маршруте движения транспорта мостов. *Указание*. Если поставить в соответствие мостам дуги определенного графа и задать длины дуг равными соответствующим ограничениям на нагрузку моста, то задача сводится к поиску такого пути между вершинами А и В, в котором длина кратчайшей дуги максимальна.

## Описание программы

Программа представляет собой консольное приложение, которое решает поставленную задачу

Как и в предыдущей программе матрица заполняется пользователем вручную. Далее используется алгоритм поиска нужного пути, который берет идеи у BFS алгоритма. Последовательно проходим от первой до последней вершины, записывая суммарный вес пути и уровень вершины, а также предыдущую вершину, чтобы потом записать весь путь.

|  |
| --- |
| class pick  {  public:  int lvl = 0;  int weight = 0;  int name = 1;  int prev=-1;    };  void SearchPath(int a[80][80],int n, vector<pick> &Picks)  {  for (int i = 0; i < n - 1; i++)//вершина  {  for (int j = i+1; j < n ; j++)//все смежные вершины  {  if (a[i][j] > 0&& Picks[j].weight< Picks[i].weight+ a[i][j])  {  Picks[j].lvl = Picks[i].lvl + 1;  Picks[j].name = j+1;  Picks[j].weight = Picks[i].weight + a[i][j];  Picks[j].prev = i+1;    }  }  }  } |

Листинг 1

## Демонстрация программы

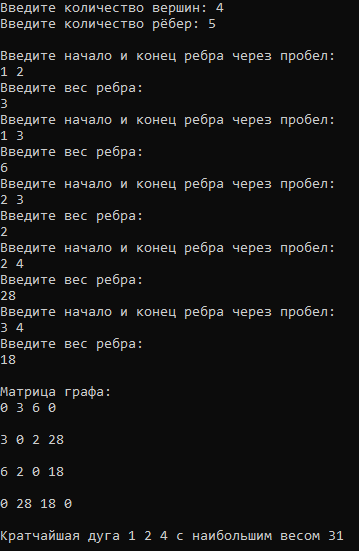


Рис. Пример работы программы

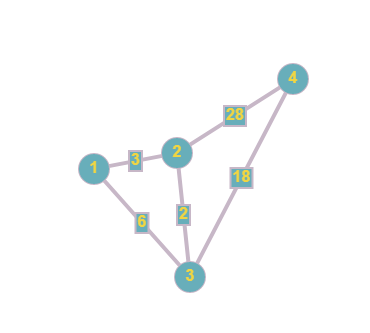


Рис. 2. Граф для поиска

## ПРИЛОЖЕНИЕ

Исходный код всей программы(Листинг 2)

|  |
| --- |
| #include <vector>  #include <iostream>  #include <string>  using namespace std;  class pick  {  public:  int lvl = 0;  int weight = 0;  int name = 1;  int prev=-1;    };  void SearchPath(int a[80][80],int n, vector<pick> &Picks)  {  for (int i = 0; i < n - 1; i++)//вершина  {  for (int j = i+1; j < n ; j++)//все смежные вершины  {  if (a[i][j] > 0&& Picks[j].weight< Picks[i].weight+ a[i][j])  {  Picks[j].lvl = Picks[i].lvl + 1;  Picks[j].name = j+1;  Picks[j].weight = Picks[i].weight + a[i][j];  Picks[j].prev = i+1;    }  }  }  }  void Results(vector<pick> Picks)  {  int flag=100000,i;  flag = Picks[Picks.size() - 1].lvl;  i= Picks[Picks.size() - 1].name;  string res;  while (flag != -1)  {  res+= to\_string(Picks[i-1].name);  i = Picks[i - 1].prev;  flag--;  }  cout << "Кратчайшая дуга ";  for (int i = res.size() - 1; i >= 0; i--)  {  cout << res[i] << " ";  }  cout<< "с наибольшим весом "<< Picks[Picks.size() - 1].weight<<endl<<endl;  }  void main()  {  int n, m, x, y,p, a[80][80];  setlocale(LC\_ALL, "rus");  cout << "Введите количество вершин: ";  cin >> n;  vector<pick> Picks(n);  cout << "Введите количество рёбер: ";  cin >> m;  for (int i = 0; i < n; i++)  for (int j = 0; j < n; j++)  {  a[i][j] = 0;  }  cout << endl;    for (int i = 0; i < m; i++)  {  cout << "Введите начало и конец ребра через пробел: " << endl;  cin >> x >> y;  cout << "Введите вес ребра: " << endl;  cin >> p;  a[x-1][y-1] =p;  a[y-1][x-1] = p;  }  cout << endl;    cout << "Матрица графа: " << endl;  for (int i = 0; i < n; i++)  {  for (int j = 0; j <n; j++)  {  cout << a[i][j] << " ";  }  cout << endl<<endl;  }  SearchPath(a, n, Picks);  Results(Picks);  } |

Листинг 2