РОССИЙСКОЙ ФЕДЕРАЦИИ
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

Лабораторная работа №4.4

по дисциплине: Дискретная математика
тема: «Кратчайшие пути во взвешенном орграфе»

Выполнил: ст. группы ПВ-211
Чувилко Илья Романович

Проверили:
Рязанов Юрий Дмитриевич
Бондаренко Татьяна Владимировна

Белгород 2022 г.

Вариант №24

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

Выполнение работы:

№1. Изучить алгоритм Дейкстры нахождения кратчайших путей между вершинами взвешенного орграфа.

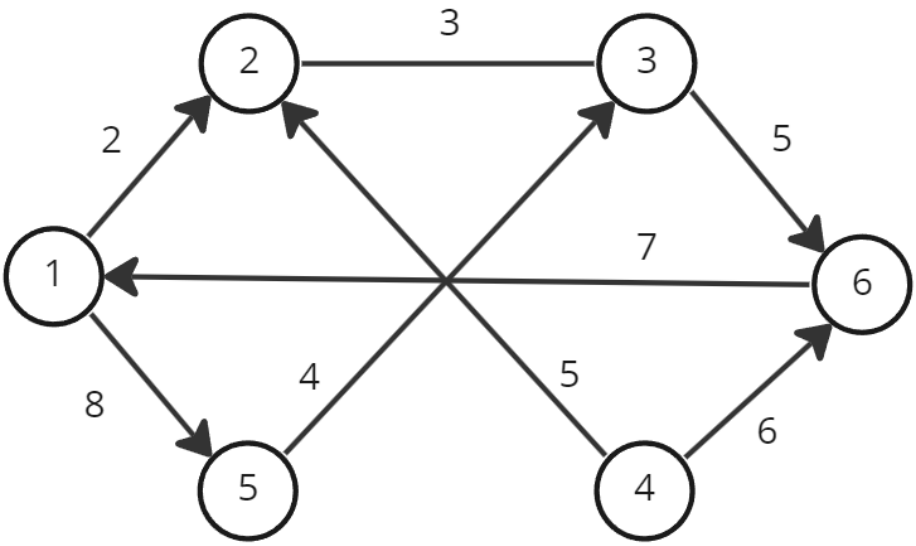
№2. Используя алгоритм Дейкстры, разработать и реализовать алгоритм решения задачи: «Найти кратчайший путь во взвешенном орграфе от вершины x до вершины y, проходящий сначала через вершину v, а затем — через вершину w. Вывести найденный путь и его длину».

```
#include <iostream>
#include <vector>

using namespace std;
using GraphRow = vector<int>;
using Graph = vector<GraphRow>;


void outputDijkstraRoute(const vector<int> &V,
                         const vector<int> &T,
                         size_t from, const size_t to) {
  while (from != to) {
    size_t j = 0;
    for (size_t i = 0; i < T.size() and j == 0; i++)
      if (T[i] == from and V[i] == 1) {
        cout << from << " ->" << i + 1 << endl;
        j = i;
      }
    from = j + 1;
  }
}

bool dijkstra_(const Graph &g, int v1, int v2,
               vector<int> &T, vector<int> &V,
               int &dl) {
  int tmp = dl, min;
  vector<int> D(g.size(), INT_MAX);
  D[v1 - 1] = 0;
  V.assign(g.size(), 0);
  V[v1 - 1] = 1;
  T.assign(g.size(), -1);
  T[v1 - 1] = 0;

  bool f = true;
  while (v1 != v2 and f) {
    f = false;
    for (size_t i = 0; i < g.size(); i++) {
      if (g[v1 - 1][i]) {
        D[i] = std::min(D[i], D[v1 - 1] + g[v1 - 1][i]);
        T[i] = v1;
      }
    }
    min = INT_MAX;
    for (size_t i = 0; i < g.size(); i++)
      if (V[i] == 0 and min > D[i]) {
```

```
        min = D[i];
        v1 = i + 1;
        f = true;
      }
    V[v1 - 1] = 1;
  }

  if (min == INT_MAX)
    return true;

  for (size_t i = 0; i < D.size() and D[i] != INT_MAX; i++)
    for (size_t j = i + 1; j < D.size(); j++)
      if (D[i] == D[j])
        return true;

  dl += min;
  return false;
}

void dijkstra(const Graph &g, const vector<int> &needRoute) {
  int dl = 0;
  vector<int> T(g.size());
  vector<int> V(g.size());
  for (size_t i = 0; i < needRoute.size() - 1; ++i) {
    if (dijkstra_(g, needRoute[i],
                  needRoute[i + 1], T, V, dl) or dl < 0) {
      cout << "No way between: " << needRoute[i] <<
           " и " << needRoute[i + 1] << endl;
      return;
    } else
      outputDijkstraRoute(V, T, needRoute.at(i), needRoute.at(i + 1));
  }

  cout << endl << "minimum distance: " << dl << endl;
}

int main() {
  Graph v({{0, 2, 0, 0, 8, 0},
           {0, 0, 3, 0, 0, 0},
           {0, 3, 0, 0, 0, 5},
           {0, 5, 0, 0, 0, 6},
           {0, 0, 4, 0, 0, 0},
           {7, 0, 0, 0, 0, 0}});

  Graph g(v);
  g.output();
  cout << endl;
  vector<int> needRoute = {4, 1, 3, 5};
  dijkstra(g, needRoute);

  return 0;
}
```

№3. Подобрать тестовые данные. Результат представить в виде диаграммы графа.

Тестовый граф:



Результат работы программы:

```
C:\BGTU\BGTU\DisMat\lab_4_4\Code\cmake-build-debug\Code.exe
0 2 0 0 8 0
0 0 3 0 0 0
0 3 0 0 0 5
0 5 0 0 0 6
0 0 4 0 0 0
7 0 0 0 0 0

4->6
6->1
1->2
2->3
3->6
6->1
1->5

minimum distance: 38

Process finished with exit code 0
```

Вывод: изучили алгоритм Дейкстры нахождения кратчайших путей между вершинами взвешенного орграфа, научиться рационально использовать его при решении различных задач.