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 №4.5

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.

Цель работы: изучить алгоритмы нахождения кратчайших путей между каждой парой вершин во взвешенном орграфе, научиться использовать их при решении различных задач.

№1. Изучить алгоритмы нахождения кратчайших путей между каждой парой вершин во взвешенном орграфе.

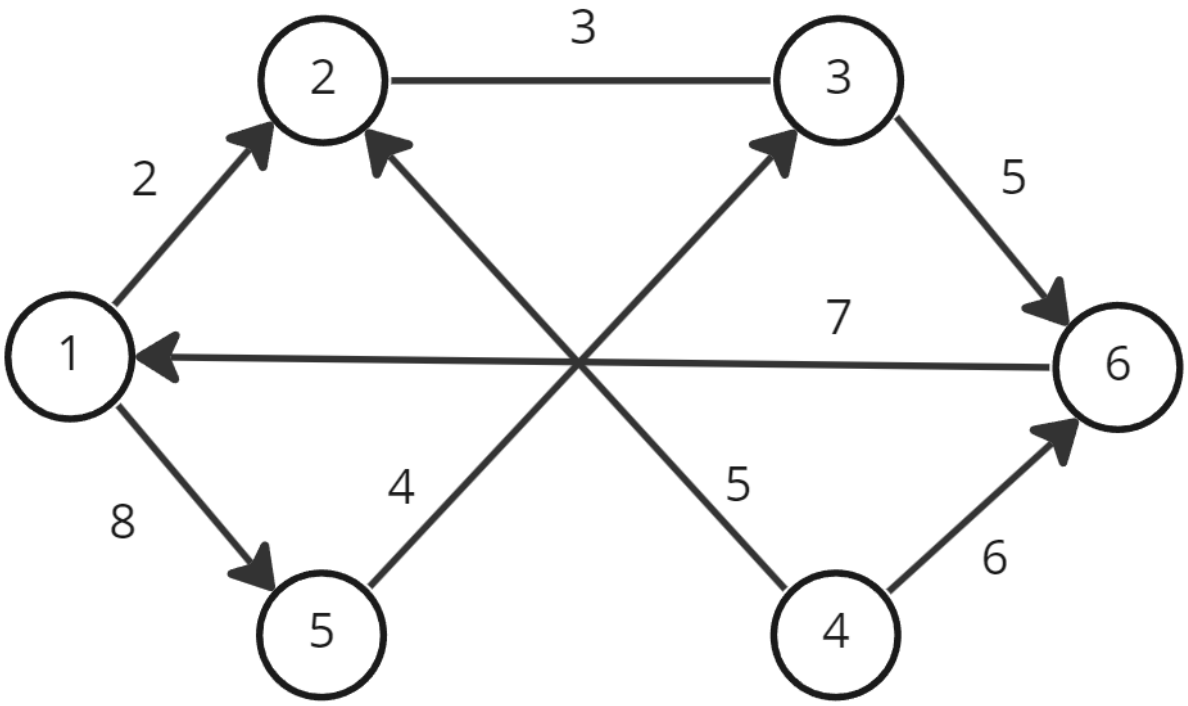
№2. Разработать и реализовать алгоритм решения задачи: “Во взвешенном орграфе найти все такие вершины vi , что сумма кратчайших расстояний от всех вершин орграфа до vi меньше суммы кратчайших расстояний от vi до всех вершин орграфа.”

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

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

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;
}

int 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 << "Нет пути между: " << needRoute[i] <<
           " и " << needRoute[i + 1] << endl;
      return NULL;
    }
  }
  return dl;
}

void OutputAllGoodVert(Graph &G) {
  vector<vector<int>> Lengths(G.size(), vector<int>(G.size()));

  for (int i = 0; i < G.size(); ++i) {
    vector<int> needRoute = {i, i + 1};
    Lengths[i][i+1] = dijkstra(G, needRoute);
  }

  for (int i = 0; i < G.size(); ++i) {
    int fromVertSum = 0;
    int toVertSum = 0;

    for (int j = 0; j < G.size(); ++j) {
      if (Lengths[i][j] == numeric_limits<int>::max())
        Lengths[i][j] = 0;

      if (Lengths[j][i] == numeric_limits<int>::max())
        Lengths[j][i] = 0;

      fromVertSum += Lengths[i][j];
      toVertSum += Lengths[j][i];
    }

    if (toVertSum < fromVertSum)
      cout << i + 1 << ' ';
  }
}

int main() {
  Graph graph({{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.output();
  cout << endl;
  OutputAllGoodVert(graph);

  return 0;
}
```

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

Тестовый граф:



Результат работы программы:

```
C:\BGTU\BGTU\DisMat\lab_4_5\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

Process finished with exit code 0
```

Вывод: изучили алгоритмы нахождения кратчайших путей между каждой парой вершин во взвешенном орграфе, научились использовать их при решении различных задач.