

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
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 ВТ-211
Руденко Ксения Ильинична

Проверили:
Рязанов Юрий Дмитриевич
Бондаренко Татьяна Владимировна

Белгород, 2022 г.

Лабораторная работа №4.4

«Кратчайшие пути во взвешенном орграфе»

Цель работы: изучить алгоритм Краскала построения покрывающего леса, научиться использовать его при решении различных задач.

Содержание отчета:

- Тема лабораторной работы;
- Цель лабораторной работы;
- Задания к лабораторной работе;
- Выводы.

Задания к лабораторной работе: вариант №5.

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».

```
1 void print_dijkstra_route (const vector<int> &V,  
2                             const vector<int> &T,  
3                             size_t v1, const size_t v2) {  
4     while (v1 != v2) {  
5         size_t j = 0;  
6         for (size_t i = 0; i < T.size() and j == 0; i++)  
7             if (T[i] == v1 and V[i] == 1) {  
8                 cout << v1 << "->" << i + 1 << endl;  
9                 j = i;  
10            }  
11  
12            v1 = j + 1;  
13        }  
14    }  
15  
16 bool dijkstra_ (const Graph &g, int v1, int v2,  
17                 vector<int> &T, vector<int> &V,  
18                 int &dl) {  
19     int tmp = dl, min;  
20     vector<int> D(g.rows, INT_MAX); D[v1 - 1] = 0;
```

```

21 V.assign(g.rows, 0); V[v1 - 1] = 1;
22 T.assign(g.rows, -1); T[v1 - 1] = 0;
23
24 bool f = true;
25 while (v1 != v2 and f) {
26     f = false;
27     for (size_t i = 0; i < g.rows; i++) {
28         if (g.values[v1 - 1][i]) {
29             D[i] = std::min(D[i],
30                             D[v1 - 1] + g.values[v1 - 1][i]);
31             T[i] = v1;
32         }
33     }
34
35     min = INT_MAX;
36     for (size_t i = 0; i < g.rows; i++)
37         if (V[i] == 0 and min > D[i]) {
38             min = D[i];
39             v1 = i + 1;
40             f = true;
41         }
42
43     V[v1 - 1] = 1;
44 }
45
46 if (min == INT_MAX)
47     return true;
48
49 for (size_t i = 0; i < D.size() and D[i] != INT_MAX; i++)
50     for (size_t j = i + 1; j < D.size(); j++)
51         if (D[i] == D[j])
52             return true;
53
54 dl += min;
55 return false;
56 }
57
58 void dijkstra(const Graph &g, const vector<int> &needRoute) {
59     int dl = 0; vector<int> T(g.rows); vector<int> V(g.rows);
60     for (size_t i = 0; i < needRoute.size() - 1; ++i) {
61         if (dijkstra_(g, needRoute[i],
62                       needRoute[i + 1], T, V, dl) or dl < 0) {
63             cout << "Нет пути между: " << needRoute[i] << " и "
64                  << needRoute[i + 1] << endl;
65             return;
66         } else
67             print_dijkstra_route(V, T, needRoute.at(i),
68                                  needRoute.at(i + 1));
69     }
70
71     cout << endl << "Кратчайшее расстояние: " << dl << endl;
72 }
73
74 int main() {
75     vector<vector<int>> v7 = {{0, 1, 9, 0, 0, 0},
76                               {0, 0, 7, 2, 0, 0},
77                               {0, 0, 0, 0, 1, 0},
78                               {0, 0, 4, 0, 8, 2},
79                               {0, 0, 0, 0, 0, 5},
80                               {0, 0, 0, 0, 4, 0}};

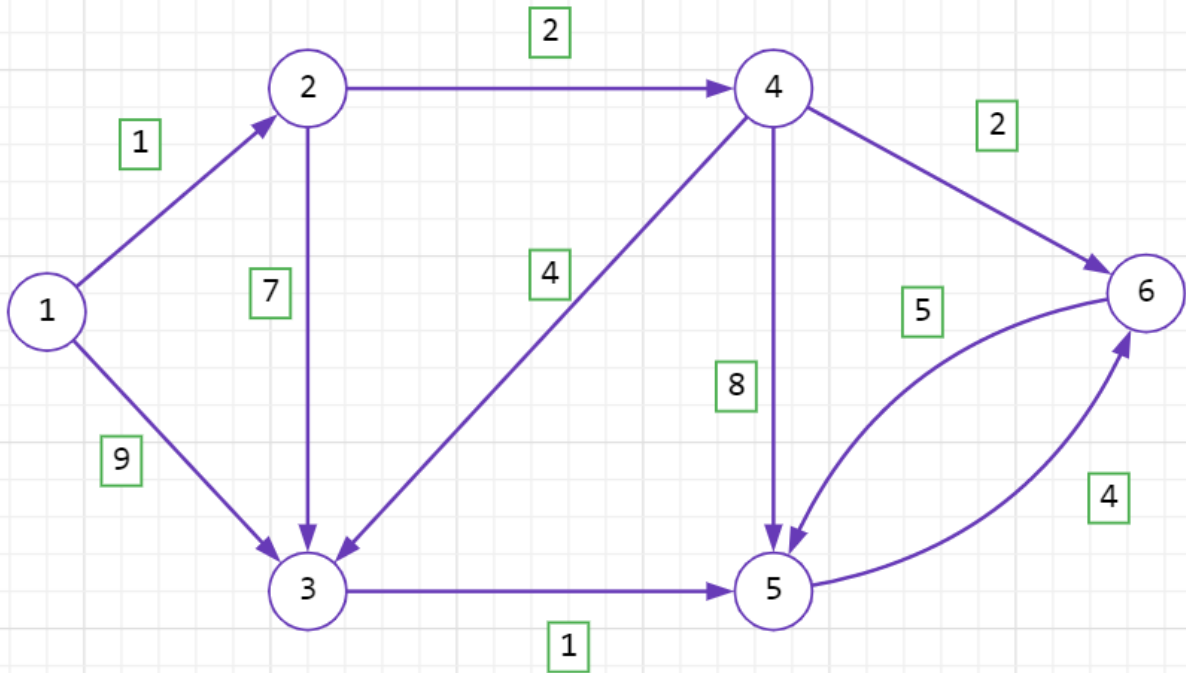
```

```

81     Graph g(v7);
82     g.output(); cout << endl;
83
84     vector<int> needRoute = {1, 4, 3, 6};
85     dijkstra(g, needRoute);
86
87     return 0;
88 }

```

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



The screenshot shows a C++ IDE with the following components:

- Project Explorer:** Shows a project named 'main_plus' with files like 'main.cpp', 'graphs.h', 'relation.h', 'sort_functions.cpp', 'useful_functions.cpp', 'useful_functions.h', 'ChokeList.h', and 'main.exe'.
- Code Editor:** Displays the implementation of the graph and the Dijkstra algorithm. The code includes the definition of the graph structure, the Dijkstra function, and the main function that initializes the graph and finds the shortest path from node 1 to node 6.
- Run Output:** Shows the output of the program, which is a 6x6 adjacency matrix representing the graph weights. The matrix is:

0	1	9	0	0	0
0	0	7	2	0	0
0	0	0	0	1	0
0	0	4	0	8	2
0	0	0	0	0	5
0	0	0	0	4	0
- Console:** Displays the shortest path found: 'Кратчайшее расстояние: 13' and 'Process finished with exit code 0'.

Вывод: изучили алгоритм Краскала построения покрывающего леса, научились использовать его при решении различных задач.