Министерство цифрового развития, связи

и массовых коммуникаций Российской Федерации

Сибирский Государственный Университет Телекоммуникаций и Информатики

СибГУТИ

Кафедра прикладной математики и кибернетики

**РАСЧЕТНО-ГРАФИЧЕСКАЯ РАБОТА**

Выполнил: студент гр. ИП-014

Михеев Д.М.

Преподаватель: Авдеев К.И.

Новосибирск 2022

Оглавление

[Постановка задачи 3](#_Toc122469540)

[Теоретическая часть 4](#_Toc122469541)

[Код программы 5](#_Toc122469542)

[Результаты тестирования 7](#_Toc122469543)

# Постановка задачи

***Задание:***

Нахождение кратчайшего расстояния (Форд-Беллман).

*Формат входных данных:*

Матрица весов, список ребер либо другой альтернативный вариант. В лабораторной разумно делать ввод не с клавиатуры, а из файла.

*Формат выходных данных:*

Сами кратчайшие пути и их длины.

# Теоретическая часть

Формируем массив – минимально возможная стоимость переезда (перехода, перевозки) из вершины в на каждом этапе работы нашего алгоритма.

Первоначально он задается как = (0,∞, ∞,…,∞).

Затем пересчитываем стоимости всех вершин по формуле

=

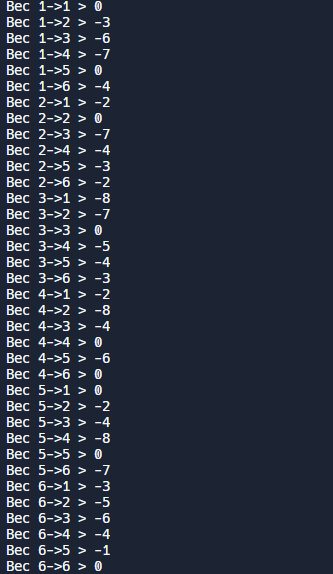
до тех пор, пока система не стабилизируется (так называемое *транзитивное замыкание*). В результате мы получим стоимости переезда из каждой вершины графа до заданной и эти стоимости будут минимально возможными.

# Код программы

1. #include <iostream>
2. #include <fstream>
4. #define inf 100000
5. using namespace std;
6. struct Edges {
7. int u, v, w;
8. };
9. const int SIZE=6;
10. int a[SIZE][SIZE] = {
11. {0, -3, -6, -7, 0, -4},
12. {-2, 0, -7, -4, -3, -2},
13. {-8, -7, 0, -5, -4, -3},
14. {-2, -8, -4, 0, -6, 0},
15. {0, -2, -4, -8, 0, -7 },
16. {-3, -5, -6, -4, -1, 0},
18. };
19. const int Vmax = 1000;
20. const int Emax = Vmax \* (Vmax - 1) / 2;
21. int i, j, n, e, start;
22. Edges edge[Emax];
23. int d[Vmax];
25. void bellman\_ford(int n, int s) {
26. int i, j;
28. for (i = 0; i < n; i++)
29. d[i] = inf;
30. d[s] = 0;
32. for (i = 0; i < n - 1; i++)
33. for (j = 0; j < e; j++)
34. if (d[edge[j].v] + edge[j].w < d[edge[j].u])
35. d[edge[j].u] = d[edge[j].v] + edge[j].w;
37. for (i = 0; i < n; i++)
38. if (d[i] == inf)
39. cout << endl
40. << start << "->" << i + 1 << "="
41. << "Not";
42. else
43. cout << endl << start << "->" << i + 1 << "=" << d[i];
44. }
45. int main() {
46. setlocale(LC\_ALL, "Rus");
47. int w;
48. n=SIZE;
49. e = 0;
50. for (i = 0; i < n; i++)
51. for (j = 0; j < n; j++) {
52. cout << "Вес " << i + 1 << "->" << j + 1 << " > " << a[i][j] << endl;
53. if (a[i][j] != 0) {
54. edge[e].v = i;
55. edge[e].u = j;
56. edge[e].w = a[i][j];
57. e++;
58. }
59. }
61. cout << "Стартовая вершина > ";
62. cin >> start;
63. cout << "Список кратчайших путей:";
64. bellman\_ford(n, start - 1);
65. }

# Результаты тестирования

Стоимости вершин на каждом шаге:



Кратчайшие расстояния:

