

Отчет по расчетной работе
по дисциплине
ПиОИВИС

Казаченко Вадим
(Гр. 321702)

Дата сдачи: 27 декабря 2023 г.

1 Цель:

Ознакомиться с основами теории графов, способами представления графов, базовыми алгоритмами для работы с разными видами графов.

2 Условия задания

Найти ребреный граф для неориетированного. Граф задается списком смежности.

3 Базовые понятия

- **Граф** — совокупность двух множеств - множества вершин и множества рёбер.
- **Неориентированный граф** — граф ребра которого не имеют направления
- **Список смежности** — один из способов представления графа в виде коллекции списков вершин. Каждой вершине графа соответствует список, состоящий из «соседей» этой вершины.

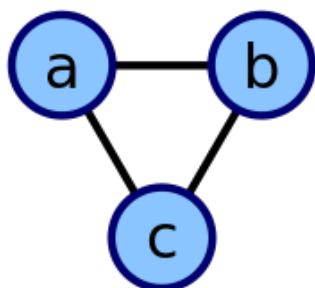


Рис. 1: Неориентированный граф

4 Описание работы программы

- 1. Программа открывает файл для чтения данных графа.
- 2. Программа последовательно считывает строки из файла и обрабатывает их.
- 3. Для каждой строки, программа извлекает номер вершины и список смежных вершин.
- 4. Программа создает и заполняет вектор 'spisok', который представляет список смежности графа.

- 5. Программа выводит матрицу смежности графа и информацию о смежных ребрах.
- 6. По завершении работы, программа закрывает файл и выводит сообщение о том, что файл не найден, если не удалось открыть файл.

5 Код

```

1  #include <fstream>
2  #include <iostream>
3  #include <vector>
4  #include <sstream>
5  using namespace std;
6
7  int vertshina;
8  vector<pair<int, int>> rebra;
9
10 void find_sosed_craya(const vector<vector<int>>& spisok
11 ) {
12     int num = 1;
13
14     //
15     for (int i = 0; i < vertshina; i++) {
16         for (int j = 0; j < spisok[i].size(); j++) {
17             int vertshina1 = i + 1;
18             int vertshina2 = spisok[i][j];
19             rebra.push_back(make_pair(vertshina1,
20                                     vertshina2)); //
21
22             num++;
23         }
24     }
25
26     cout << "-----" << endl << endl;
27     cout << "          :" << endl;
28     for (int i = 0; i < rebra.size(); i++) {
29         cout << i + 1 << " : " << rebra[i].first << "
30             - " << rebra[i].second << endl; //
31     }

```

```

28     cout << " _-----_" << endl << endl;
29 }
30
31 int main() {
32     vector<vector<int>> spisok;
33
34     setlocale(LC_ALL, "RU");
35     ifstream file_graph("graph.txt");
36
37     if (file_graph.is_open()) {
38         string line;
39         while (getline(file_graph, line)) {
40             stringstream ss(line);
41             string vertshina_str;
42             getline(ss, vertshina_str, ':');
43             int vertex = stoi(vertshina_str);
44
45             spisok.resize(max(spisok.size(),
46                             static_cast<size_t>(vertex)));
47
48             string adjacent_vertices_str;
49             getline(ss, adjacent_vertices_str);
50
51             stringstream ss_adj(adjacent_vertices_str);
52             string adjacent_vertex_str;
53             while (ss_adj >> adjacent_vertex_str) {
54                 int adjacent_vertex = stoi(
55                     adjacent_vertex_str);
56                 spisok[vertex - 1].push_back(
57                     adjacent_vertex); //
58
59             }
60         }
61
62         file_graph.close();
63
64         vertshina = spisok.size();

```

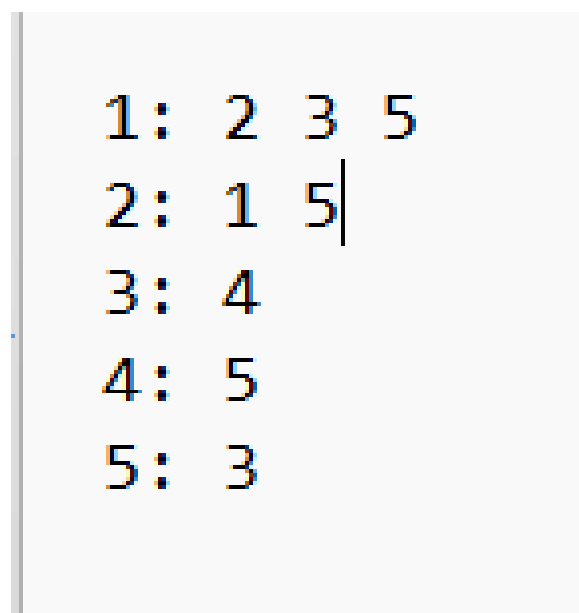
```

62         cout << "                                     ":" <<
           endl;
63     for (int i = 0; i < vertshina; i++) {
64         cout << "                                     " << i + 1 << ":" ";
65         for (int j = 0; j < spisok[i].size(); j++)
66             {
67                 cout << spisok[i][j] << " ";
68             }
69         cout << endl;
70     }
71     cout << endl;
72     find_sosed_craya(spisok);
73 }
74 else {
75     cout << "                                     ." << endl;
76 }
77
78 system("pause");
79 return 0;
80 }

```

6 Пример выполнения

1. Условие



```

1: 2 3 5
2: 1 5
3: 4
4: 5
5: 3

```

Рис. 2: Список смежности

2. Результат

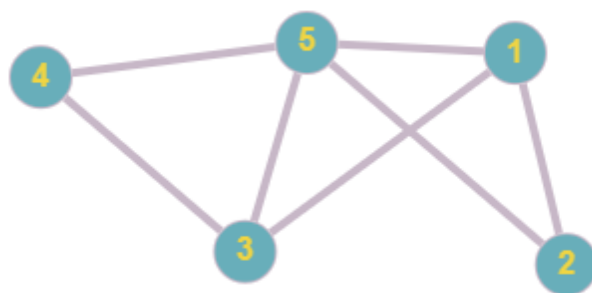


Рис. 3: Граф

```
Матрица смежности:
Вершина 1: 2 3 5
Вершина 2: 1 5
Вершина 3: 4
Вершина 4: 5
Вершина 5: 3

Смежные ребра:
1 : 1 - 2
2 : 1 - 3
3 : 1 - 5
4 : 2 - 1
5 : 2 - 5
6 : 3 - 4
7 : 4 - 5
8 : 5 - 3

Для продолжения нажмите любую клавишу . . .
```

Рис. 4: Enter Caption

Вывод

- изучены основы теории графов
- изучены способы представления графов
- изучены базовые алгоритмы для работы с графами