|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | **ИУК «Информатика и управление»** |
| **КАФЕДРА** | **ИУК4 «Программное обеспечение ЭВМ,** |
| **информационные технологии»** | |

**ЛАБОРАТОРНАЯ РАБОТА №3**

**«Реализация основных алгоритмов с графами»**

**ДИСЦИПЛИНА: «Типы и структуры данных»**



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. ИУК4-31Б | |  |  | ( | Суриков Н. С. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |
| Проверил: | |  |  | ( | Былинка М. И. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |

|  |  |
| --- | --- |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | |
|  | - Балльная оценка:  - Оценка: |

**Цель:** формирование практических навыков создания алгоритмов обработки графов.

**Задачи:**

1. Познакомиться со способами представления графов в памяти компьютера.
2. Изучить основные обходы графов.
3. Научиться составлять алгоритмы для нахождения кратчайших путей в графе.
4. Реализовать алгоритм согласно варианту.

**Вариант 25**

**Формулировка задания:**

*Используя нерекурсивную процедуру поиска в глубину, найти города, расположенные от города В на расстоянии, большем Т.*

**Листинг программы:**

**main.cpp**

1 *#include* *"Graph.h"*

2 *#include* *<iostream>*

3

4 int main()

5 {

6 Graph graph("input.txt");

7

8 char startCity;

9 int threshold;

10 std::cout << "Введите начальный город (A-E) и порог расстояния: ";

11 std::cin >> startCity >> threshold;

12

13 graph.findCitiesBeyondDistance(startCity, threshold);

14

15 return 0;

16 }

**Graph.cpp**

1 *#include* *"Graph.h"*

2 *#include* *<fstream>*

3 *#include* *<iostream>*

4 *#include* *<stack>*

5

6 Graph::Graph(const std::string &filename)

7 {

8 std::ifstream file(filename);

9 if (!file.is\_open())

10 {

11 std::cerr << "Ошибка открытия файла!" << std::endl;

12 return;

13 }

14

15 file >> numCities;

16 adjacencyMatrix.resize(numCities, std::vector<int>(numCities, 0));

17

18 for (int i = 0; i < numCities; ++i)

19 {

20 for (int j = 0; j < numCities; ++j)

21 {

22 file >> adjacencyMatrix[i][j];

23 }

24 }

25 file.close();

26 }

27

28 int Graph::cityIndex(char city)

29 {

30 return city - 'A';

31 }

32

33 char Graph::indexCity(int index)

34 {

35 return 'A' + index;

36 }

37

38 void Graph::dfs(int startCity, int threshold)

39 {

40 std::vector<bool> visited(numCities, false);

41 std::stack<std::pair<int, int>> stack; *// (city, currentDistance)*

42 stack.push({startCity, 0});

43 visited[startCity] = true;

44

45 while (!stack.empty())

46 {

47 auto [city, currentDistance] = stack.top();

48 stack.pop();

49

50 for (int i = 0; i < numCities; ++i)

51 {

52 if (adjacencyMatrix[city][i] > 0)

53 {

54 int newDistance = currentDistance + adjacencyMatrix[city][i];

55

56 if (newDistance > threshold && !visited[i])

57 {

58 std::cout << indexCity(i) << " ";

59 }

60

61 if (!visited[i])

62 {

63 visited[i] = true;

64 stack.push({i, newDistance});

65 }

66 }

67 }

68 }

69 }

70

71 void Graph::findCitiesBeyondDistance(char startCity, int threshold)

72 {

73 int startIndex = cityIndex(startCity);

74 std::cout << "Города на расстоянии больше " << threshold << " от города " << startCity << ": ";

75 dfs(startIndex, threshold);

76 std::cout << std::endl;

77 }

**Graph.h**

1 *#ifndef GRAPH\_H*

2 *#define GRAPH\_H*

3

4 *#include* *<string>*

5 *#include* *<vector>*

6

7 class Graph

8 {

9 public:

10 Graph(const std::string &filename);

11 void findCitiesBeyondDistance(char startCity, int threshold);

12

13 private:

14 std::vector<std::vector<int>> adjacencyMatrix;

15 int numCities;

16 void dfs(int startCity, int threshold);

17 int cityIndex(char city);

18 char indexCity(int index);

19 };

20

21 *#endif // GRAPH\_H*

**Результат работы:**

Матрица смежности

5

0 10 0 30 100

10 0 50 0 0

0 50 0 20 10

30 0 20 0 60

100 0 10 60 0

Введите начальный город (A-Z) и порог расстояния: A 30

Города на расстоянии больше 30 от города A: E C

Введите начальный город (A-Z) и порог расстояния: A 10

Города на расстоянии больше 10 от города A: D E C

Введите начальный город (A-Z) и порог расстояния: A 130

Города на расстоянии больше 130 от города A:

**Вывод:** в ходе работы были сформированы практические навыки создания алгоритмов обработки графов.