Министерство науки и высшего образование Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ИОБРАЗОВАТЕЛЬНОЕ УЧЕРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЕ

Национальный исследовательский университет ИТМО

МЕГАФАКУЛЬТЕТ ТРАНСЛЯЦИОННЫХ ИНФОРСМАЦИОННЫХ ТЕХНОЛОГИЙ

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ПРОГРАММИРОВАНИЯ

ЛАБОРАТОРНЯ РАБОТОТА №5

По дисциплине Введение в цифровую культуры и программирования

Название работы: Работа с графом

Выполнил: Тарасов Михаил Евгеньевич

Проверил: Страдина Марина Владимировна

Санкт- Петербург, 2020 г.

Задание:

Граф содержит 1000 вершин, задан списком рёбер, которые перечислены в файле graphedges148.txt
Вопрос 1. Сколько в нём рёбер?
Изолятом называется вершина, не связанная ни с одной другой вершиной.
Вопрос 2. Сколько в графе изолятов? Выведите полный список, упорядоченный по возрастанию
Степенью вершины называется количество ребёр, которые связывают её с другими вершинами.
Вопрос 3. Найдите вершину(вершины) с самой большой степенью.
Кратчайший путь - минимальная сумма рёбер, составляющих путь от одной вершины к другой.
Компонента связности - это максимальный связный подграф. Диаметр - это самый длинный кратчайший путь
Вопрос 4. Найдите диаметр компоненты связности графа
Вопрос 5. Найдите кратчайший путь от A до B. A: 938 B: 333
Вопрос 6. Найдите кратчайший путь от C до D. C: 446 D: 467
Вопрос 7. Найдите кратчайший путь от E до F. E: 60 F: 448
* ответ должен включать в себя длину пути и последовательность вершин
** если путь отсутствует, то сделайте соответствующую пометку.
Удалите из графа следующие вершины: [833, 515, 228, 107, 751, 212, 509]
Вопрос 8. Сколько ребёр в графе?
Вопрос 9. Сколько в графе изолятов? Выведите полный список, упорядоченный по возрастанию
Вопрос 10. Найти вершину(вершины) с самой большой степенью
Вопрос 11. Найдите диаметр компоненты связности графа
Вопрос 12. Найдите кратчайший путь от A до B
Вопрос 13. Найдите кратчайший путь от C до D
Вопрос 14. Найдите кратчайший путь от E до F

Ответы:

Вопрос 1: Количество вершин в графе: 2600
Вопрос 2: Изоляторы нашего графа: 77 346 376 654 876 970. Количество изоляторов в графе: 6
Вопрос 3: Максимальная степень: 12. В вершины с этими степенями: 748 751.
Все вершины, с которыми связан элемент с наибольшим кол-вом рёбер:
748) 5 188 194 278 347 447 505 579 619 781 850 930
751) 43 81 149 187 364 393 428 475 515 625 643 705

Код на C++:

```cpp
#include <iostream>
#include <fstream>
#include <vector>
#include <set>

using namespace std;

struct Graph
{
    vector<int> cur_top;
    vector<int> linked;
};

int main()
{
    ifstream fin( s: "graphedges148.txt");

    int N = 1000;
    int top, linked_top, edge_count = 0, insulator = 0;
    int check_exist; // проверка на входит или нет
    vector <Graph> arr_top(N); // массив вершин
    set <int> set_insulator; // множество изоляторов(элементов у которых нет соединений)
    vector <int> pow_top(N); // степень вершины (кол-во связей)

    while (fin >> top) // пока есть вершины на ввод
    {
        fin >> linked_top; // вводим вершины и связанную с ней вершину
        arr_top[top].linked.push_back(linked_top);
        arr_top[linked_top].linked.push_back(top);
        // вставка в массив неповторяющихся элементов
        set_insulator.insert(top);
        set_insulator.insert(linked_top);
        // увеличение степени вершины
        pow_top[top]++;
        pow_top[linked_top]++;
        // запись элементов, с которыми связана вершина
        // запись элементов, с которыми связана вершина
        check_exist = 0;
        for (int i = 0; i < arr_top[top].cur_top.size(); i++)
            if (arr_top[top].cur_top[i] == linked_top)
                check_exist++;
        if (!check_exist)
            arr_top[top].cur_top.push_back(linked_top);
        // запись вершин, с которыми связано с другой стороны
        for (int i = 0; i < arr_top[linked_top].cur_top.size(); i++)
            if (arr_top[linked_top].cur_top[i] == top)
                check_exist++;
        if (!check_exist)
            arr_top[linked_top].cur_top.push_back(top);
        // счётчик вершин графа
        edge_count++;
    }
    cout << "Kol-vo vershin v grafe: " << edge_count << endl << endl;
    cout << "Izolytiri v gafe:  " << endl;
    for (int i = 0; i < N; i++)
    {
        if (!set_insulator.count(i))
        {
            cout << i << " ";
            insulator++;
        }
    }
    cout << endl;
    for (int i = 0; i < N; i++)
        edge_count += arr_top[i].cur_top.size();
    cout << "Kol-vo izolytor v grafe: " << insulator << endl;
    cout << endl << endl;
    int max = 0, index;
    for (int i = 0; i < N; i++)
    {
        if (pow_top[i] > max)
```

```
71          {
72              max = pow_top[i];
73              index = i;
74          }
75      }
76      vector <int> max_pow( n: 0);
77      for (int i = 0; i < N; i++)
78          if (pow_top[i] == max)
79              max_pow.push_back(i);
80      cout << "Max pow: " << max << endl;
81      cout << "Vse vershini s etim pow: ";
82      for (int i = 0; i < max_pow.size(); i++)
83          cout << max_pow[i] << " ";
84      cout << endl << endl;
85      cout << "Vse vershini s kotorimi cvyzan element s max kol-vom vershin: " << endl;
86      for (int j = 0; j < max_pow.size(); j++)
87      {
88          cout << max_pow[j] << ") ";
89          for (int i = 0; i < arr_top[max_pow[j]].cur_top.size(); i++) {
90
91              cout << arr_top[max_pow[j]].cur_top[i] << " ";
92          }
93          cout << endl;
94      }
95      cout << endl << endl;
96      return 0;
97  }
98
```