BFS :: Give up on this task :: Work on this task later :: Back to task selection
Reward: Ⓝ **3.50**.

Characters: **2081**, Images: **1**, Exercises: **5** When you are ready to submit, click here: Submit for Review

---

Title

BFS(Поиск в ширину)

Subhdr

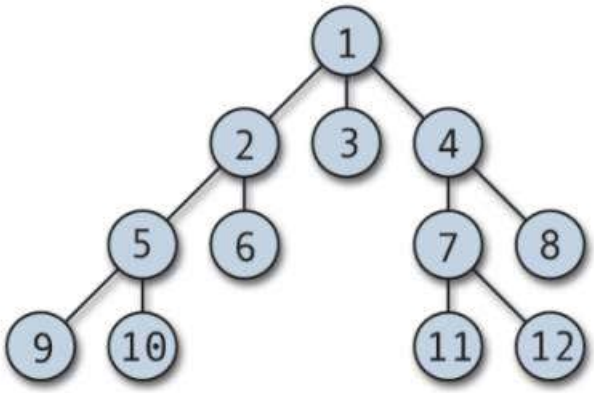
Определение

Text

BFS (breadth-first search) - это один из методов обхода графа. Алгоритм поиска в ширину систематически обходит все ребра графа для «открытия» всех вершин, достижимых из исходной вершины, вычисляя при этом расстояние (минимальное количество рёбер) от начальной вершины до каждой достижимой из нее вершины.

Поиск в ширину имеет такое название потому, что в процессе обхода мы идём вширь, то есть перед тем как приступить к поиску вершин на расстоянии i+1, выполняется обход вершин на расстоянии i.

Image



Subhdr

АЛГОРИТМ

Text

Задаем граф с помощью списка смежности v. Также нам понадобится массив d, в котором будут хранится расстояния от стартовой вершины до каждой вершины в графе, массив used, который будет отвечать за посещение вершины, задаем в переменной start стартовую вершину, допустим это будет первая с индексом 0, также нам понадобится такая структура как очередь q, куда помещаем текущие рассматриваемые вершины, помещаем стартовую вершину туда и помечаем ее в массиве used как посещенную, затем в цикле while пока наша очередь не пустая, достаем верхний элемент из нее, присваиваем переменной r, сразу же его удаляем. Затем надо перебрать все достижимые, из текущей, вершины присвоив их переменной to, дальше делаем проверку на посещение рассматриваемой вершины через if. Если же мы не посещали эту вершину, то добавляем ее в очередь и помечаем как посещенную и изменяем расстояние до этой вершины: d[to] = d[r] + 1. На этом алгоритм заканчивается, остается только вывести массив d, в котором и будут все расстояния до вершин от начальной, внизу представлена реализация на C++:

```
#include <iostream>
#include <queue>
#include <vector>

using namespace std;

vector<vector<int>> v(100000);
bool used[100000];
int d[100000];

int main()
```

```
{
    int n, m;
    cin >> n >> m;
    for(int i = 0; i < m; ++i)
    {
        int a, b;
        cin >> a >> b;
        a--; b--;
        v[a].push_back(b);
        v[b].push_back(a);
    }
    for(int i = 0; i < m; ++i)
        d[i] = 100000;

    int start = 0;
    used[start] = true;
    d[start] = 0;
    queue<int> q;
    q.push(start);
    while(!q.empty())
    {
        int r = q.front();
        q.pop();
        for(int i = 0; i < (int)v[r].size(); ++i)
        {
            int to = v[r][i];
            if(used[to])
                continue;
            q.push(to);
            used[to] = true;
            d[to] = d[r] + 1;
        }
    }

    for(int i = 0; i < n; ++i)
        cout << d[i] << " ";
    return 0;
}
```

Замечание: данный алгоритм находит кратчайшие расстояния для неориентированного графа.

Subhdr

СЛОЖНОСТЬ АЛГОРИТМА

Text

Временная сложность данного алгоритма будет O(n+m), где n - количество вершин, а m - количество ребер.

Exercise

Task Type: Photo / Picture Solution Type: Multiple Choice

Task Description: Какой минимальный путь из вершины 3 в вершину 5

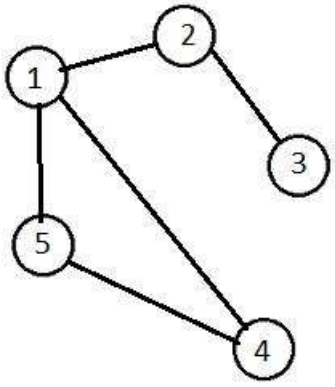
Correct solution

- (•) 3
- ( ) 4
- ( ) 1

Wrong solution (What is this?)

- ( ) 3
- (•) 4
- ( ) 1

Solution



Multiple Choice Options:

3 ×
4 ×
1 ×
Add

Explanation:
Из вершины 3 в вершину 5 можно попасть через ребра, которые соединяют вершины (2, 3); (2, 1); (1, 5). Т.е. минимальный расстояние это 3

Exercise

Task Type: Text Solution Type: Multiple Choice

Task

Программист Вася решил написать на C++ алгоритм BFS, внизу представлен кусок программы, который содержит цикл while, где просматривается вершины в очереди queue, но Вася допустил ошибку в коде, помогите ему определить в чем ошибка?

```
while(!q.empty())
    {
        int r = q.front();
        q.pop();
        for(int i = 0; i < (int)v[r].size(); ++i)
        {
            int to = v[r][i];
            if(used[to])
                continue;
            q.push(to);
            d[to] = d[r] + 1;
        }
    }
```

Solution

Correct solution
- ○ Строка q.pop() до цикла for не нужна
- ◉ Он забыл пометить в массиве used вершину to как посещенную
- ○ В строке d[to] = d[r] + 1 не надо добавлять 1

Wrong solution (What is this?)
- ◉ Строка q.pop() до цикла for не нужна
- ○ Он забыл пометить в массиве used вершину to как посещенную
- ○ В строке d[to] = d[r] + 1 не надо добавлять 1

Multiple Choice Options:

Строка q.pop() до цикла for ×
Он забыл пометить в масси ×
В строке d[to] = d[r] + 1 не н ×
Add

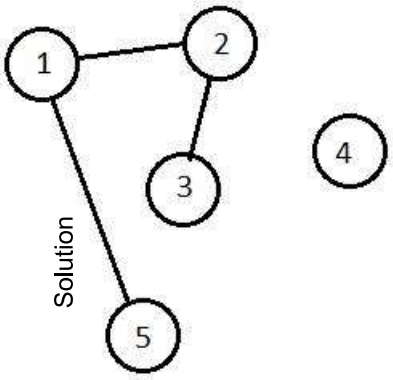
Explanation:
Когда мы взяли вершину to, ее надо пометить как посещенную, чтобы не рассматривать ее еще раз.

Exercise

Task Type: Photo / Picture Solution Type: Multiple Choice

Task Description: Как будет выглядеть массив с расстояниями от вершины 1

Correct solution



- (●) 0, 1, 2, 0, 1
- ( ) 1, 1, 2, 0, 1
- ( ) 0, 1, 2, 1, 2

Wrong solution (What is this?)

- ( ) 0, 1, 2, 0, 1
- (●) 1, 1, 2, 0, 1
- ( ) 0, 1, 2, 1, 2

Multiple Choice Options:

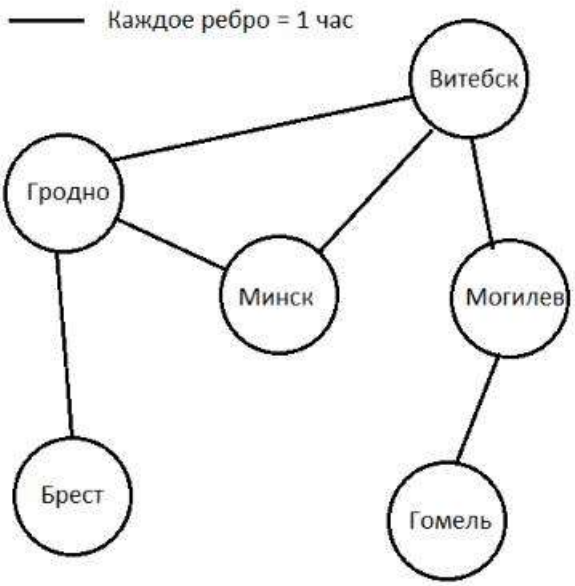
0, 1, 2, 0, 1 ×
1, 1, 2, 0, 1 ×
0, 1, 2, 1, 2 ×
Add

Explanation:
Второй вариант не подходит, т.к. расстояние от стартовой вершины 1 до самой себя должна быть 0.
Третий вариант не подходит, т.к. вершина под номером 4 не соединена с другими вершинами, значит до нее расстояние от 1 вершины должно быть 0.

Exercise

Task Type: Photo / Picture Solution Type: Multiple Choice

Task Description: За сколько времени можно добраться из Гомеля в Гродно



Solution

Correct solution

- ( ) 2 часа
- (●) 3 часа
- ( ) 4 часа

Wrong solution (What is this?)

- (●) 2 часа
- ( ) 3 часа
- ( ) 4 часа

Multiple Choice Options:

2 часа ×
3 часа ×
4 часа ×
Add

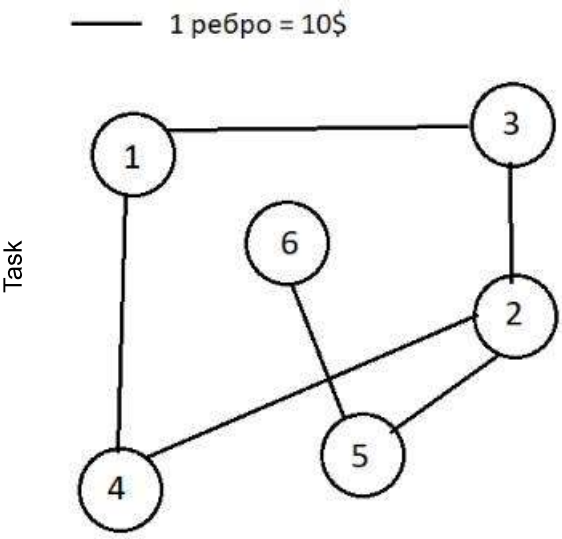
Explanation:
Из Гомеля в Гродно можно попасть по следующему пути: Гомель-Могилев-Витебск-Гродно, т.е. пройти по трем ребрам, учитывая, что каждое ребро - это 1 час пути, то все путешествие займет 3 часа.

Exercise

Task Type: Photo / Picture Solution Type: Text

Task Description: Сколько $ минимально будет стоить дорога из 1 вершины до 6

Task



Solution

Correct solution
40

Wrong solution (What is this?)
50

Explanation:
Кратчайший маршрут из 1 вершины до 6 будет таким: 1-3-2-5-6 или 1-4-2-5-6. В любом их этих двух маршрутов мы проходим 4 ребра, т.к. каждое ребро стоит 10$, то весь маршрут обойдется в 40$

Exercise