CSES Problem Set

Message Route

Submission details

Task:	Message Route
Sender:	tonykk
Submission time:	2024-11-10 11:59:27 +0200
Language:	Python3 (CPython3)
Status:	READY
Result:	ACCEPTED

Test results

test	verdict	time	
#1	ACCEPTED	0.02 s	»
#2	ACCEPTED	0.02 s	»
#3	ACCEPTED	0.02 s	»
#4	ACCEPTED	0.02 s	»
#5	ACCEPTED	0.02 s	»
#6	ACCEPTED	0.77 s	»
#7	ACCEPTED	0.77 s	»
#8	ACCEPTED	0.78 s	»
#9	ACCEPTED	0.76 s	»
#10	ACCEPTED	0.71 s	»
#11	ACCEPTED	0.02 s	»
#12	ACCEPTED	0.51 s	»

Code

```python
from collections import deque, defaultdict


def legrovidebb_ut_keresese():
    # Adatok beolvasása
    n, m = map(int, input().split())

    # Gráf inicializálása
    graf = defaultdict(list)
    for _ in range(m):
        a, b = map(int, input().split())
        graf[a].append(b)
        graf[b].append(a)

    # inicializálás (BFS)
    sor = deque([1])
    tavolsag = {1: 1}  # Az 1-es csúcsról indulunk, távolsá
```


Graph Algorithms

- Counting Rooms
- Labyrinth
- Building Roads
- Message Route
- Building Teams
- Round Trip
- Monsters
- Shortest Routes I

...

Your submissions

- 2024-11-10 11:59:27 ✓
- 2024-11-10 11:15:11 ✓
- 2024-11-10 11:13:50 ✗
- 2024-11-07 10:07:37 ✓
- 2024-11-07 10:03:59 ✗

```python
    elozo_csucs = {1: None}  # Az út visszakövetéséhez

    # BFS ciklus
    while sor:
        csucs = sor.popleft()

        # Ha elérjük a célt (az n-edik csúcsot), akkor megá
        if csucs == n:
            break

        # Szomszédok bejárása
        for szomszed in graf[csucs]:
            if szomszed not in tavolsag:  # Még nem jártunk
                tavolsag[szomszed] = tavolsag[csucs] + 1
                elozo_csucs[szomszed] = csucs
                sor.append(szomszed)

    # Ellenőrizzük, hogy elértük-e az n-edik csúcsot
    if n not in tavolsag:
        print("IMPOSSIBLE")
        return

    # Útvonal visszakövetése az 1-estől az n-edikig
    utvonal = []
    jelenlegi_csucs = n
    while jelenlegi_csucs is not None:
        utvonal.append(jelenlegi_csucs)
        jelenlegi_csucs = elozo_csucs[jelenlegi_csucs]

    utvonal.reverse()  # Az út visszafelé van, így megfordí
    print(tavolsag[n])  # Az út hossza
    print(" ".join(map(str, utvonal)))  # Az út kiírása



legrovidebb_ut_keresese()
```

SHARE CODE TO OTHERS

Test details

Test 1

Verdict: ACCEPTED

input

```
10 20
8 9
6 7
9 10
...
```

correct output

```
5
1 4 5 8 10
```

user output

```
5
1 4 5 8 10
```

Test 2

Verdict: ACCEPTED

input

```
10 20
3 4
7 8
3 6
2 4
...
```

correct output

```
5
1 2 4 7 10
```

user output

```
5
1 2 4 7 10
```

Test 3

Verdict: ACCEPTED

input

```
10 20
2 4
8 10
2 5
8 9
...
```

correct output

```
5
1 2 5 7 10
```

user output

```
5
1 2 5 7 10
```

Test 4

Verdict: ACCEPTED

input

```
10 20
1 2
4 5
3 6
8 10
...
```

correct output

```
4
1 2 6 10
```

user output

```
4
1 2 6 10
```

Test 5

Verdict: ACCEPTED

input

```
10 10
5 7
3 5
7 9
5 9
...
```

correct output

```
IMPOSSIBLE
```

user output

```
IMPOSSIBLE
```

Test 6

Verdict: ACCEPTED

input

```
100000 200000
91175 91206
80200 80201
93273 93296
17478 17523
...
```

correct output

```
3025
1 42 41 74 114 162 169 222 253...
```

user output

```
3025
1 42 41 74 114 162 169 222 253...
```

Test 7

Verdict: ACCEPTED

input

```
100000 200000
6944 6945
93198 93199
54590 54591
14273 14293
...
```

correct output

```
6602
1 2 22 25 49 74 94 96 97 110 1...
```

user output

```
6602
1 2 22 25 49 74 94 96 97 110 1...
```

Test 8

Verdict: ACCEPTED

input

```
100000 200000
39621 39622
10754 10795
86702 86703
15170 15184
...
```

correct output

```
2235
1 57 58 59 126 186 255 299 360...
```

user output

```
2235
1 57 58 59 126 186 255 299 360...
```

Test 9

Verdict: ACCEPTED

input

```
100000 200000
89513 89514
21947 21954
76082 76083
15033 15034
...
```

correct output

```
3702
1 2 32 33 77 76 107 139 140 16...
```

user output

```
3702
1 2 32 33 77 76 107 139 140 16...
```

Test 10

Verdict: ACCEPTED

input

```
100000 200000
44678 44704
84851 84861
16615 16641
32838 32876
...
```

correct output

```
IMPOSSIBLE
```

user output

```
IMPOSSIBLE
```

Test 11

Verdict: ACCEPTED

input

```
2 1
1 2
```

correct output

```
2
1 2
```

user output

```
2
1 2
```

Test 12

Verdict: ACCEPTED

input

```
99999 149997
1 2
2 3
3 4
4 5
...
```

correct output

```
50000
1 3 5 7 9 11 13 15 17 19 21 23...
```

user output

```
50000
1 3 5 7 9 11 13 15 17 19 21 23...
```