{% note info %} **摘要** Title: 851. spfa求最短路 Tag: spfa、最短路 Memory Limit: 64 MB Time Limit: 1000 ms {% endnote %}

Powered by:**NEFU AB-IN**

Link

@TOC

851. spfa求最短路

- 题意

> 给定一个 n 个点 m 条边的有向图，图中可能存在重边和自环， 边权可能为负数。 请你求出 1 号点到 n 号点的最短距离，如果无法从 1 号点走到 n 号点，则输出 impossible。 数据保证不存在负权回路。

- 思路

SPFA板子题（bellman-ford的队列优化）

核心思想：

  - 我更新过谁，再拿谁来更新别人
  - 队列中存的是待更新的点，如果队列里有某点，那么就不让它入队；如果队列里没有某点，那么让它入队，并更新状态；出队时也更新状态
  - 某个点可以被多次更新 **注意**：
  - 队列维护，只需维护顶点即可，因为不需优先队列根据distance排序
  - 图中不能存在负权回路

{% note info %} **为什么bellman-ford算法写成 if(dist[n] > 0x3f3f3f/2) return -1; spfa确是 if(dist[n] == 0x3f3f3f) return -1** 因为队列里都是由起点更新到的点，不存在bellman-ford算法中未更新的点同样被边更新的情况。因为本题不是直接遍历所有边，所以距离是正无穷的点不会被插入队列再去更新其他点。 {% endnote %}

{% note info %} **spfa和Dijkstra区别** spfa中一个点的距离可能被**多次更新**，因为有权值为负数的边。st[i]代表i是否在队列中 Dijktra中一个点的距离只被**更新一次**，每次确定一个点的距离不再改变。st[i]代表i是否已被更新为最短路 {% endnote %}

- 代码

```
'''
Author: NEFU AB-IN
Date: 2022-03-03 12:15:39
FilePath: \ACM\Acwing\851.py
LastEditTime: 2022-03-03 12:34:28
'''
```

```python
from collections import deque

N = int(1e5 + 10)
INF = int(2e9)
st, dist = [0] * N, [INF] * N
g = [[] for _ in range(N)]
q = deque()


def spfa(s):
    dist[s] = 0
    q.appendleft(s)
    st[s] = 1
    while q:
        u = q.pop()
        st[u] = 0
        for v, w in g[u]:
            if dist[v] > dist[u] + w:
                dist[v] = dist[u] + w
                if st[v] == 0:
                    q.appendleft(v)
                    st[v] = 1
    return dist[n]


n, m = map(int, input().split())
for i in range(m):
    x, y, z = map(int, input().split())
    g[x].append([y, z])
res = spfa(1)
if res == INF:
    print("impossible")
else:
    print(res)
```