

1. Introduction:

實作 Dijkstra's Algorithm 和 Bellman-Ford Algorithm 來找出最短路徑，以及 negative loop。

2. Implement Details:

Dijkstra's Algorithm:

先將所有 node 的距離設為無限大，然後再把起點的距離設為零，當路徑加上所在位置的 node 距離小於下一個前往 node 的距離，就把下一個 node 距離更新為路徑加上所在位置的 node 距離。

Bellman-Ford Algorithm:

大致上實作與 Dijkstra's Algorithm 相同，但是新增一個偵測有沒有 negative loop 的功能，偵測方法為在檢查一遍各點的距離，如果距離變化了，代表有 negative loop 存在，那就無法計算 shortest path 了。

3. Results:

Dijkstra's Algorithm:

Input (stdin)

```
5 6
0 4
0 1 5
0 2 3
1 4 6
2 3 2
2 4 2
4 3 4
```

Your Output (stdout)

```
5
```

Expected Output

```
5
```



Test Case #0



Test Case #1



Test Case #2



Test Case #3

Bellman-Ford Algorithm:

Input (stdin)

```
5 6
0 4
0 1 5
0 2 3
1 4 6
2 4 2
3 2 2
4 3 -5
```

Your Output (stdout)

```
Negative loop detected!
```

Expected Output

```
Negative loop detected!
```

Homework 3 - Bellman-Ford Algorithm

Problem

Submissions

Leaderboard

Discussions

Submitted in a few seconds • Score: 100.00

Status: Accepted

✓

Test Case #0

✓

Test Case #1

✓

Test Case #2

✓

Test Case #3

✓

Test Case #4

✓

Test Case #5

4. Discussion:

Bellman-Ford Algorithm 可以幫我們偵測是否有 negative loop，因為如果有 negative loop 就沒辦法找出 shortest path，但如果使用 Dijkstra's Algorithm 會沒有辦法發現這個問題，而得出錯誤的答案。

5. Conclusion:

這一次的 HW 是這學期最後的 DS 作業，這學期在 OOP AND DS 這堂課上學到非常多，希望我的程式能力也能因此進步。