Class4_05

1167 트리의 지름

문제

트리의 지름이란, 트리에서 임의의 두 점 사이의 거리 중 가장 긴 것을 말한다. 트리의 지름을 구하는 프로그램을 작성하시오.

입력

트리가 입력으로 주어진다. 먼저 첫 번째 줄에서는 트리의 정점의 개수 V가 주어지고 (2 ≤ V ≤ 100,000)둘째 줄부터 V개의 줄에 걸쳐 간선의 정보가 다음과 같이 주어진다. 정점 번호는 1부터 V까지 매겨져 있다.

먼저 정점 번호가 주어지고, 이어서 연결된 간선의 정보를 의미하는 정수가 두 개씩 주어지는데, 하나는 정점번호, 다른 하나는 그 정점까지의 거리이다. 예를 들어 네 번째 줄의 경우 정점 3은 정점 1과 거리가 2인 간선으로 연결되어 있고, 정점 4와는 거리가 3인 간선으로 연결되어 있는 것을 보여준다. 각 줄의 마지막에는 -1이 입력으로 주어진다. 주어지는 거리는 모두 10,000 이하의 자연수이다.

알고리즘

이거 그냥 3주? 4주? 그쯤 전에 풀었던

트리의 지름 이거랑 동일한듯

그때 트리의 지름은 아무노드에서 가장 먼노드 A 를 찾고 찾은 A 노드에서 가장 먼노드B까지의 거리가 트리의 지름 A- B 이다. 를 사용해서 풀었는데

이 문제는 굳이 아무노드라 하지 않아도, 루트가 주어지니 그 루트로 쭉 하면될듯

```
import sys
sys.setrecursionlimit(10**9)
input = sys.stdin.readline
v = int(input())
edges = [ [] for _ in range(v + 1)]
for i in range(1,v):
    parent,child,dis=map(int,input().split())
    edges[parent].append([child,dis])
    edges[child].append([parent,dis])
visited=[-1]*(v+1)

def dfs(cur, curdis):
    if visited[cur] != -1:
        return
    visited[cur] = curdis

    for next,dis in edges[cur]:
        dfs(next,curdis+dis)
```

```
# 임의의 1노드에서 dfs수행해서 가장 먼 노드 찾고
dfs(1,0)
maxnode=visited.index(max(visited))
# 그 노드를 기준으로 dfs다시
visited=[-1]*(v+1)
dfs(maxnode,0)

print(max(visited))
```

1991 트리순회

이진 트리를 입력받아 전위 순회(preorder traversal), 중위 순회(inorder traversal), 후위 순회(postorder traversal)한 결과를 출력하는 프로그램을 작성하시오.

알고리즘

이거도 뭐 그냥 재귀로 쭉 하면 될듯

3문자열 만들어 놓고 전위,중위, 후위에따라 문자열에 넣는 시점을 다르게

```
import sys
input=sys.stdin.readline
sys.setrecursionlimit(10**9)

graph=dict()

ans=["","",""]
for i in range(int(input())):
    parent,left,right=input().split()
    if left ==".":
        left = None
    if right == ".":
        right =None
    graph[parent]=[left,right]

def search(curnode):
    # 전위순회
    ans[0]+=curnode

    left ,right =graph[curnode]
    if left!= None:
        search(left)

    ans[1]+=curnode

    if right != None:
        search(right)

    ans[2]+=curnode
```

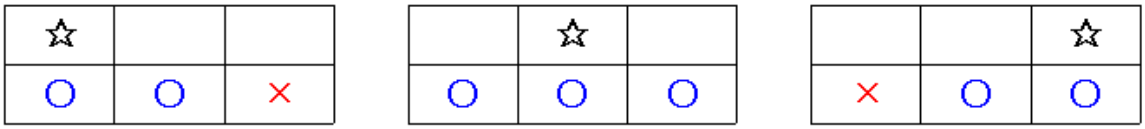
```
search("A")

print(*ans,sep= "\n")
```

2096 내려가기

N줄에 0 이상 9 이하의 숫자가 세 개씩 적혀 있다. 내려가기 게임을 하고 있는데, 이 게임은 첫 줄에서 시작해서 마지막 줄에서 끝나게 되는 놀이이다.

먼저 처음에 적혀 있는 세 개의 숫자 중에서 하나를 골라서 시작하게 된다. 그리고 다음 줄로 내려가는데, 다음 줄로 내려갈 때에는 다음과 같은 제약 조건이 있다. 바로 아래의 수로 넘어가거나, 아니면 바로 아래의 수와 붙어 있는 수로만 이동할 수 있다는 것이다. 이 제약 조건을 그림으로 나타내어 보면 다음과 같다.



별표는 현재 위치이고, 그 아랫 줄의 파란 동그라미는 원룡이가 다음 줄로 내려갈 수 있는 위치이며, 빨간 가위표는 원룡이가 내려갈 수 없는 위치가 된다. 숫자표가 주어져 있을 때, 얻을 수 있는 최대 점수, 최소 점수를 구하는 프로그램을 작성하시오. 점수는 원룡이가 위치한 곳의 수의 합이다.

알고리즘

그냥 dp로 첫줄부터 마지막줄 까지 하면될듯?

그냥 dp를 쭉 만들었더니 메모리 초과가 나서 어차피 이전 줄 만 현재 줄에 의미가 있으므로 3X2 칸만 활용해서

D[0 -> 이전줄 , 1-> 이번 줄] [0,1,2 -> 이 줄의 몇번째 칸]

D [0] [0]	**D [0] [1]**	**D [0] [2]**
D [1] [0]	D [1] [1]	D [1] [2]

이런식으로

D [0] [0]	D [0] [1]	D [0] [2]
D [1] [0]	D [1] [1]	D [1] [2]

```python
import sys

input = sys.stdin.readline

maxDp = {"pre": [0, 0, 0], "cur": [0, 0, 0]}
minDp = {"pre": [0, 0, 0], "cur": [0, 0, 0]}
for i in range(int(input())):
    temp = list(map(int, input().split()))

    maxDp["cur"][0] = max(maxDp["pre"][0], maxDp["pre"][1]) + temp[0]
    minDp["cur"][0] = min(minDp["pre"][0], minDp["pre"][1]) + temp[0]

    maxDp["cur"][1] = max(maxDp["pre"]) + temp[1]
    minDp["cur"][1] = min(minDp["pre"]) + temp[1]

    maxDp["cur"][2] = max(maxDp["pre"][1], maxDp["pre"][2]) + temp[2]
    minDp["cur"][2] = min(minDp["pre"][1], minDp["pre"][2]) + temp[2]

    maxDp["pre"] = [*maxDp["cur"]]
    minDp["pre"] = [*minDp["cur"]]

print(max(maxDp["cur"]), min(minDp["cur"]))
```