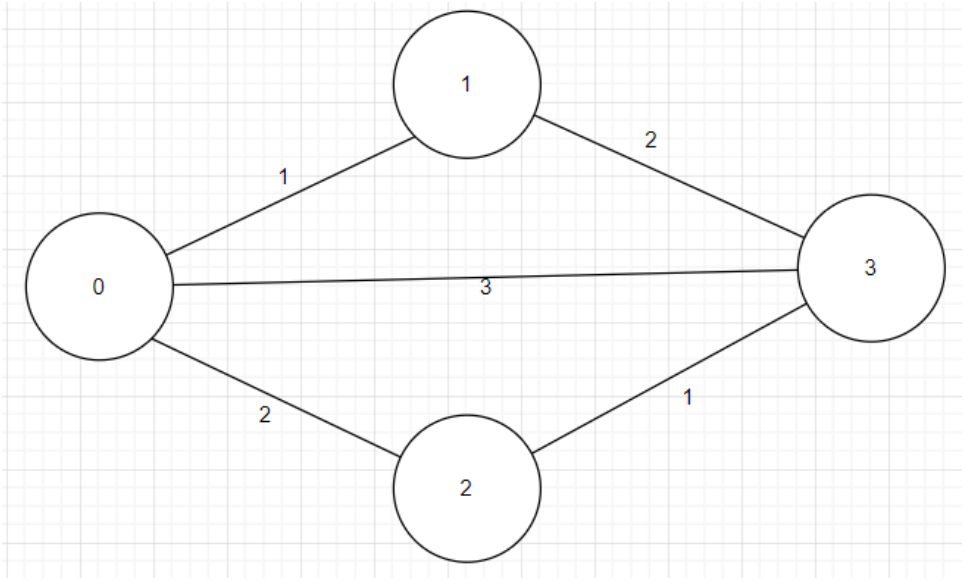

2 <= N <= 10000 , 1 <= M <= 10000, 1 <= W <= 100

Input for q2.1 :-

4 5 // N and M , N = no. Of nodes and M = no. Of edges.

0 1 1 // node1 and node2 and their weight.

1 3 2

0 3 3

0 2 2

2 3 1

3 // Destination Node

Its output will be:

0 1 3 //Path from 0 , Notice that 0->3 and 0->2->3 were also the shortest path to 3 but this is the lexicographically smallest

1 3 //Path from 1

2 3 //Path from 2

Input for q2.2:-

2 <= N <= 5 and 1 <= K <= 20 and 1<= W <= N

4 5 // N and M , N = no. Of nodes and M = no. Of edges.

0 1 1 // node1 and node2 and their weight.

1 3 2

0 3 3

0 2 2

2 3 1

3 // K means top k shortest paths

Total Paths arranged in ascending order:

1 weight:

0->1

2->3

Both 0-1 and 2-3 have same weight but 01 is lexicographically smaller

2 weight:

1->3

0->2

3 weight:

0->1->3

0->2->3

0->3

1->0->2

1->3->2

4 weight:

0->1->3->2

0->3->2

1->0->3

1->0->2->3

5 weight:

0->3->1

2->0->3

1->0->3->2

7 weight:

1->3->0->2

The output is:

0 1

2 3

1 3

Q2.3

Input have N and M where N is number of rows and M is number of columns.

'.' represents open path and '#' represents the closed path R means the position of a Rider.

You are starting from bottom right marked as 'A' and have to reach top left.

Notice that all the borders are also closed except for the top left part.

Constraints are : 1<= N,M <= 1000

Input:

7 8

########

.......#

#####..#

#.#.#..#

#R#..#.#

#.....A#

########

Output:

Yes

10