

Homework 4

1. When the robot runs into these obstacles, We should put the value 0 in that grid

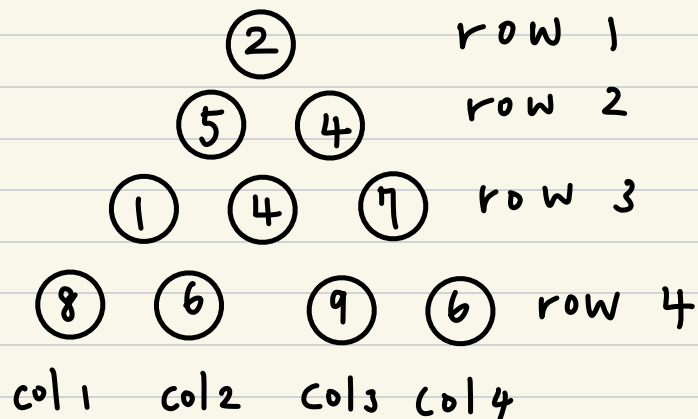
0	0	0	0	0	0
1	1	1	0	0	0
1	2	2	0	0	0
1	2	2	3	3	4
0	0	0	3	4	4

this table is the max number of coin collection

0	0	0	0	0	0
0	1	1	0	0	0
1	2	3	0	0	0
1	3	6	6	6	6
0	0	0	6	12	18

This table is for optimal path

2. First, We regard this triangle as a grid. the vertical one is row and horizontal is column.



We initial a new row dp to keep track of the summation of the total from bottom to top . $dp[8, 6, 9, 6]$

$$dp[col] = triangle[row][col] + \min(dp[col], dp[col+1])$$

$$\begin{aligned} dp[1] &= \Delta[3][1] + \min(dp[1], dp[2]) \\ &= 1 + \min(8, 6) = 7 \end{aligned}$$

$$\begin{aligned} dp[2] &= \Delta[3][2] + \min(dp[2], dp[3]) \\ &= 4 + \min(6, 9) = 10 \end{aligned}$$

$$\begin{aligned} dp[3] &= \Delta[3][3] + \min(dp[3], dp[4]) \\ &= 7 + \min(9, 6) = 13 \end{aligned}$$

$$dp[7, 10, 13, 6]$$

After sum of row 3 and row 4, we'll move forward to row 2,

$$\begin{aligned} dp[1] &= \Delta[2][1] + \min[dp[1], dp[2]] \\ &= 5 + \min(7, 10) = 12 \end{aligned}$$

$$\begin{aligned} dp[2] &= \Delta[2][2] + \min[dp[2], dp[3]] \\ &= 4 + \min(10, 13) = 14 \end{aligned}$$

$$dp[12, 14, 13, 6]$$

move forward to row 1

$$\begin{aligned} dp[1] &= \Delta[1][1] + \min(dp[1], dp[2]) \\ &= 2 + \min(12, 14) = 14 \end{aligned}$$

The smallest sum in this triangle is 14

3. The algorithm is based on recurrence relation. $C(n, k) = C(n-1, k-1) + C(n-1, k)$
It uses addition and division to calculate

The space of this algorithm is $O(1)$
it only uses constant amount of space
to store the variables used in recurrence.

The time complexity of the algorithm
iterate k times. so the $O(n)$ is
this algorithm's time complexity.

4. In this problem, You have n types of items, it means that it has an unlimited supply of n items and i is the index of n types of items. However, the used items are based on the capacity W .

We have to set the maximum items according to the capacity .

The time complexity for this algorithm is $O(n \times W)$, but this n isn't the original n , it is the maximum items based on W .

5. If the probability for each key is the same, we can use AVL tree to balance this binary search tree

This ensures that the tree remains balanced and maintain a logarithmic height

If $n = 2^x$, the average number of comparisons in a successful can be $\log n$.