

[Dashboard](#) / [Courses](#) / [2021-2022 EVEN](#) / [TTDSPY-M23](#) / [DAY -2 Coin Change Problem](#)
/ [M23-02-01- Automata Coding \(Weekly Challenge\) - SEB](#)

Started on	Thursday, 15 May 2025, 11:18 PM
State	Finished
Completed on	Thursday, 15 May 2025, 11:23 PM
Time taken	4 mins 36 secs
Grade	10.00 out of 10.00 (100%)

Question **1**
Correct
Mark 2.00 out of 2.00

Create a Python Function to find the total number of distinct ways to get a change of 'target' from an unlimited supply of coins in set 'S'.

For example:

Test	Input	Result
count(S, len(S) - 1, target)	3 4 1 2 3	The total number of ways to get the desired change is 4

Answer: (penalty regime: 0 %)

Reset answer

```
1 def count(S, n, target):
2     ##### Add Your Code Here #####
3     #Start here
4     if target == 0:
5         return 1
6     if target < 0 or n < 0:
7         return 0
8     incl = count(S, n, target - S[n])
9     excl = count(S, n - 1, target)
10    return incl + excl
11    #End here
12
13 if __name__ == '__main__':
14     S = []#[1, 2, 3]
15     n=int(input())
16     target = int(input())
17     for i in range(n):
18         S.append(int(input()))
19     print('The total number of ways to get the desired change is'
20         count(S, len(S) - 1, target))
```

	Test	Input	Expected	Got	
✓	count(S, len(S) - 1, target)	3 4 1 2 3	The total number of ways to get the desired change is 4	The total number of ways to get the desired change is 4	✓
✓	count(S, len(S) - 1, target)	3 11 1 2 5	The total number of ways to get the desired change is 11	The total number of ways to get the desired change is 11	✓

Passed all tests! ✓

Correct

Marks for this submission: 2.00/2.00.



Question **2**

Correct

Mark 2.00 out of 2.00

You are given an $n \times n$ grid representing a field of cherries, each cell is one of three possible integers.

- 0 means the cell is empty, so you can pass through,
- 1 means the cell contains a cherry that you can pick up and pass through, or
- -1 means the cell contains a thorn that blocks your way.

Return the maximum number of cherries you can collect by following the rules below:

- Starting at the position (0, 0) and reaching (n - 1, n - 1) by moving right or down through valid path cells (cells with value 0 or 1).
- After reaching (n - 1, n - 1), returning to (0, 0) by moving left or up through valid path cells.
- When passing through a path cell containing a cherry, you pick it up, and the cell becomes an empty cell 0.
- If there is no valid path between (0, 0) and (n - 1, n - 1), then no cherries can be collected.

For example:

Test	Result
obj.cherryPickup(grid)	5

Answer: (penalty regime: 0 %)

Reset answer

```
1 class Solution:
2     def cherryPickup(self, grid):
3         n = len(grid)
4         ##### Add your code here #####
5         #Start here
6         dp = [[-1] * (n + 1) for _ in range(n + 1)]
7         dp[1][1] = grid[0][0]
8         for m in range(1, (n << 1) - 1):
9             for i in range(min(m, n - 1), max(-1, m - n), -1):
10                for p in range(i, max(-1, m - n), -1):
11                    j, q = m - i, m - p
12                    if grid[i][j] == -1 or grid[p][q] == -1:
13                        dp[i + 1][p + 1] = -1
14                    else:
15                        dp[i + 1][p + 1] = max(dp[i + 1][p + 1], dp[i][i] + grid[i][j] + grid[p][q])
16                        if dp[i + 1][p + 1] != -1: dp[i + 1][p + 1] = dp[i + 1][p + 1]
17        return max(0, dp[-1][-1])
18        n,m=len(grid),len(grid[0])
19        dp = [[[-1] for i in range(m)] for j1 in range(n)]
20        #End here
21        return f(0, 0, m-1, dp)
```

	Test	Expected	Got	
✓	obj.cherryPickup(grid)	5	5	✓

Passed all tests! ✓

Correct

Marks for this submission: 2.00/2.00.



Question **3**
Correct
Mark 3.00 out of 3.00

Create a python function to compute the fewest number of coins that we need to make up the amount given.

For example:

Test	Input	Result
ob1.coinChange(s,amt)	3 11 1 2 5	3

Answer: (penalty regime: 0 %)

Reset answer

```
1 class Solution(object):
2     def coinChange(self, coins, amount):
3         ##### Add your Code Here #####
4         #End here
5         if amount == 0 :
6             return 0
7         if min(coins) > amount:
8             return -1
9         dp = [-1 for i in range(0, amount + 1)]
10        for i in coins:
11            if i > len(dp) - 1:
12                continue
13            dp[i] = 1
14            for j in range(i + 1, amount + 1):
15                if dp[j - i] == -1:
16                    continue
17                elif dp[j] == -1:
18                    dp[j] = dp[j - i] + 1
19                else:
20                    dp[j] = min(dp[j], dp[j - i] + 1)
21        return dp[amount]
```

	Test	Input	Expected	Got	
✓	ob1.coinChange(s,amt)	3 11 1 2 5	3	3	✓
✓	ob1.coinChange(s,amt)	3 12 1 2 5	3	3	✓
✓	ob1.coinChange(s,amt)	3 22 1 2 5	5	5	✓

Passed all tests! ✓

Correct

Marks for this submission: 3.00/3.00.



Question 4

Correct

Mark 3.00 out of 3.00

Create a Dynamic Programming python Implementation of Coin Change Problem.

For example:

Test	Input	Result
count(arr, m, n)	3	4
	4	
	1	
	2	
	3	

Answer: (penalty regime: 0 %)

Reset answer

```
1 def count(S, m, n):
2     table = [[0 for x in range(m)] for x in range(n+1)]
3     for i in range(m):
4         table[0][i] = 1
5     for i in range(1, n+1):
6         for j in range(m):
7             # Count of solutions including S[j]
8             #Start here
9             x = table[i - S[j]][j] if i-S[j] >= 0 else 0
10            # Count of solutions excluding S[j]
11            y = table[i][j-1] if j >= 1 else 0
12            # total count
13            table[i][j] = x + y
14        return table[n][m-1]
15    #End here
16 arr = []
17 m = int(input())
18 n = int(input())
19 for i in range(m):
20     arr.append(int(input()))
21 print(count(arr, m, n))
```

	Test	Input	Expected	Got	
✓	count(arr, m, n)	3	4	4	✓
		4			
		1			
		2			
		3			
✓	count(arr, m, n)	3	20	20	✓
		16			
		1			
		2			
		5			

Passed all tests! ✓

Correct

Marks for this submission: 3.00/3.00.

◀ M23-02-01- Automata Coding (Home Challenge)

Jump to...

⌵

M23-D2-01 Automata Fix(Weekly Challenge)-SEB ▶

