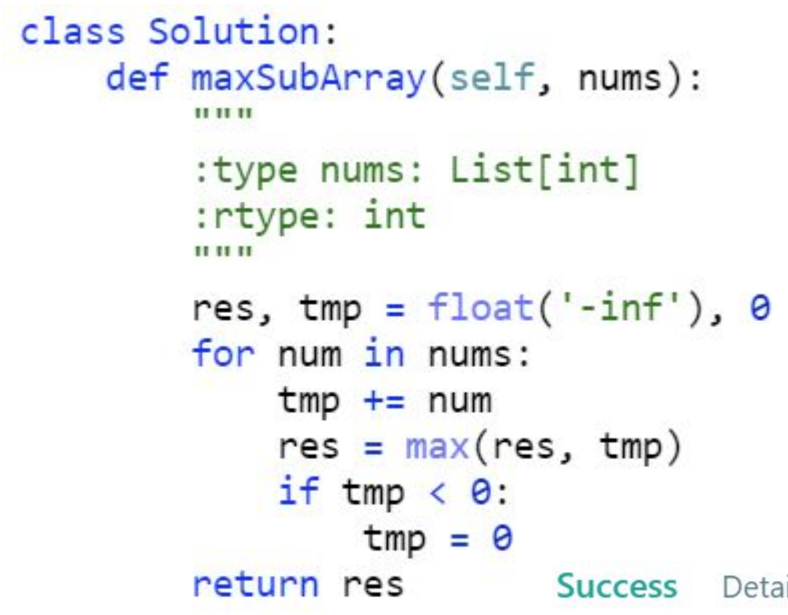
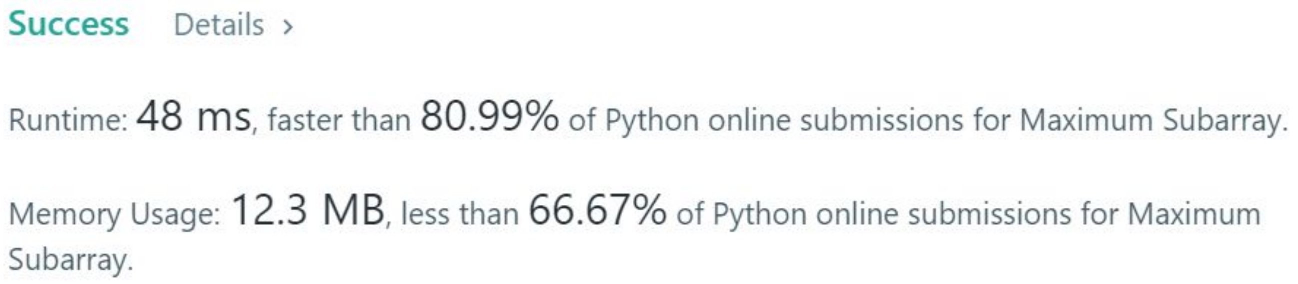
LC 53: Maximum Subarray



```python
class Solution:
    def maxSubArray(self, nums):
        """
        :type nums: List[int]
        :rtype: int
        """
        res, tmp = float('-inf'), 0
        for num in nums:
            tmp += num
            res = max(res, tmp)
            if tmp < 0:
                tmp = 0
        return res
```

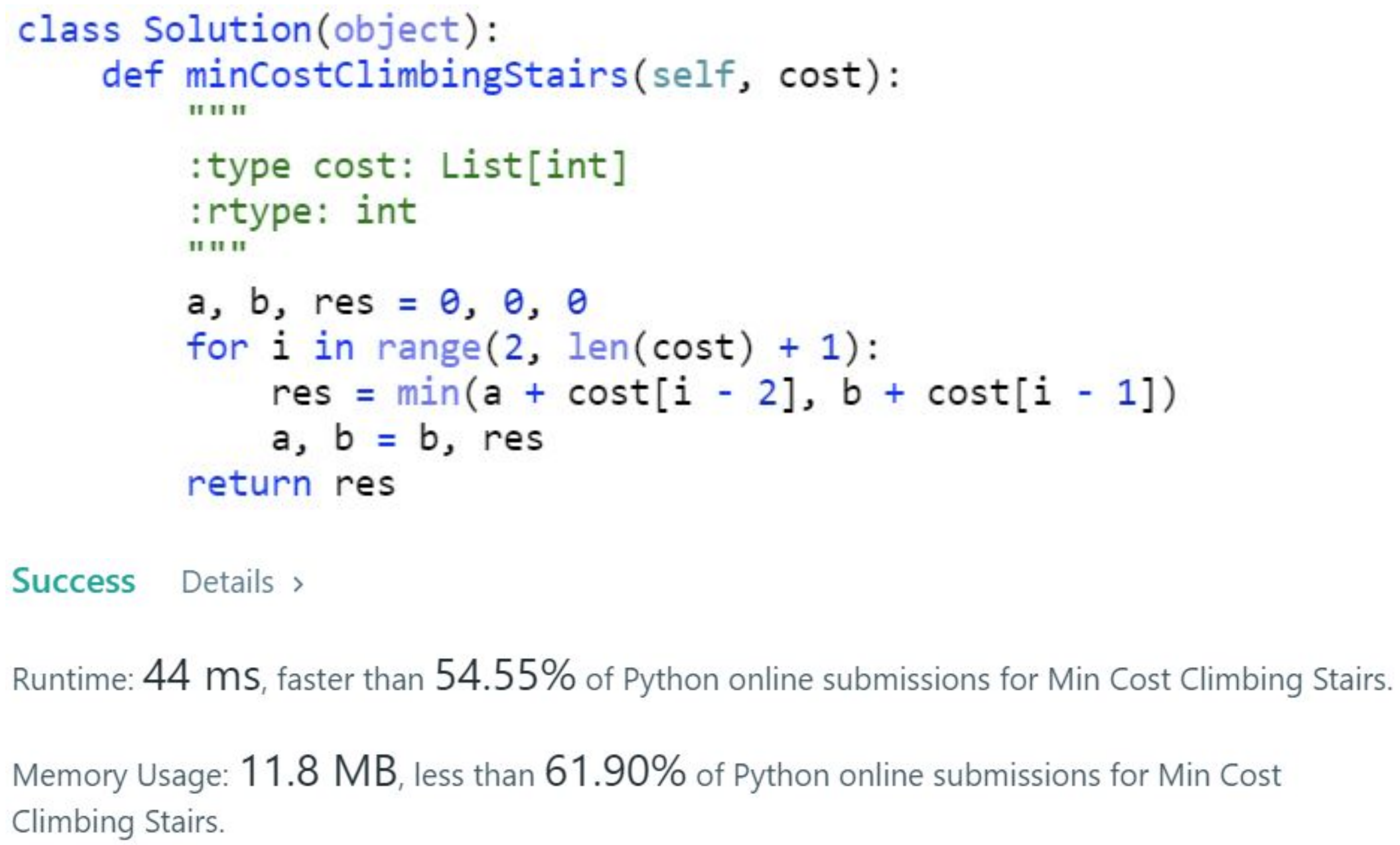


Success Details ›

Runtime: 48 ms, faster than 80.99% of Python online submissions for Maximum Subarray.

Memory Usage: 12.3 MB, less than 66.67% of Python online submissions for Maximum Subarray.

LC 746: Min Cost Climbing Stairs



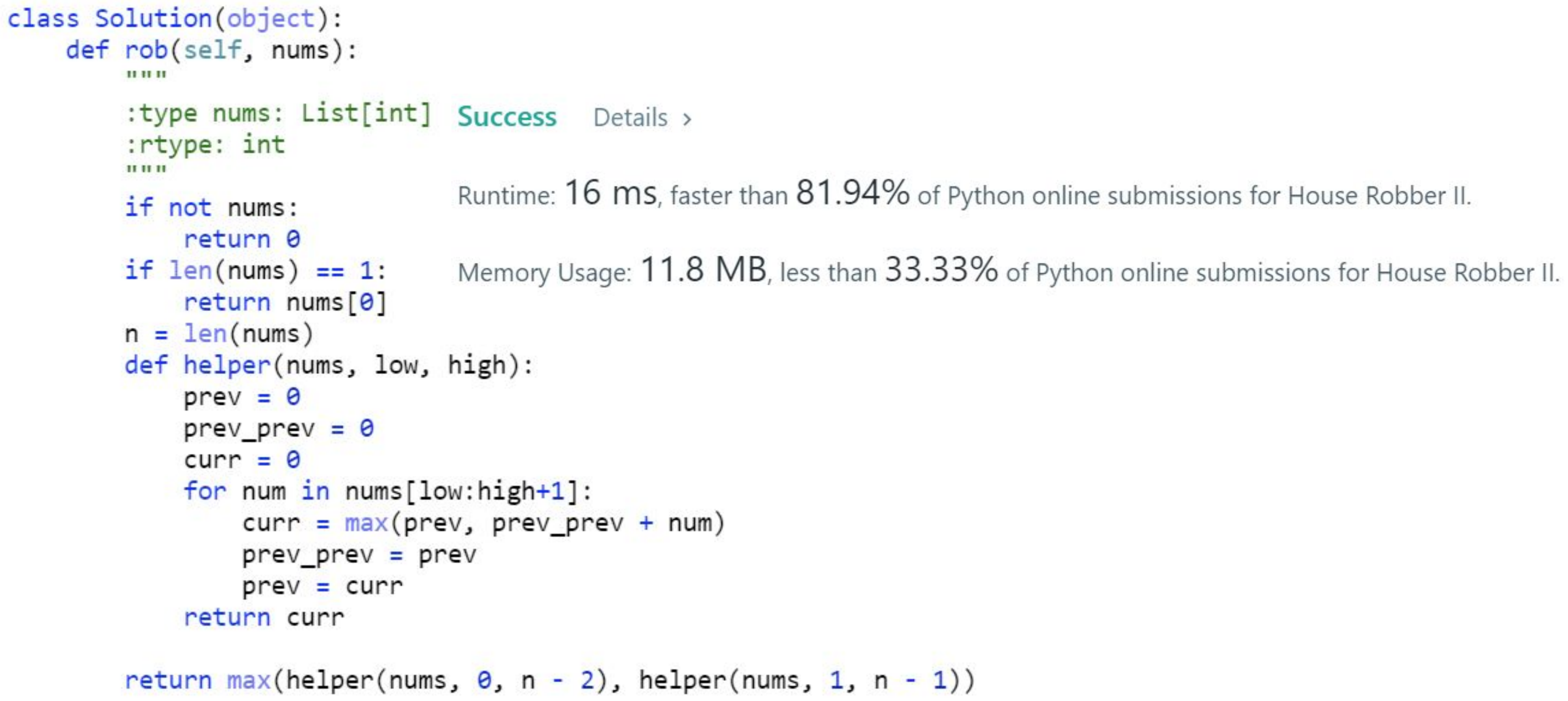
```python
class Solution(object):
    def minCostClimbingStairs(self, cost):
        """
        :type cost: List[int]
        :rtype: int
        """
        a, b, res = 0, 0, 0
        for i in range(2, len(cost) + 1):
            res = min(a + cost[i - 2], b + cost[i - 1])
            a, b = b, res
        return res
```

Success   Details ›

Runtime: 44 ms, faster than 54.55% of Python online submissions for Min Cost Climbing Stairs.

Memory Usage: 11.8 MB, less than 61.90% of Python online submissions for Min Cost Climbing Stairs.

LC 213: House Robber II



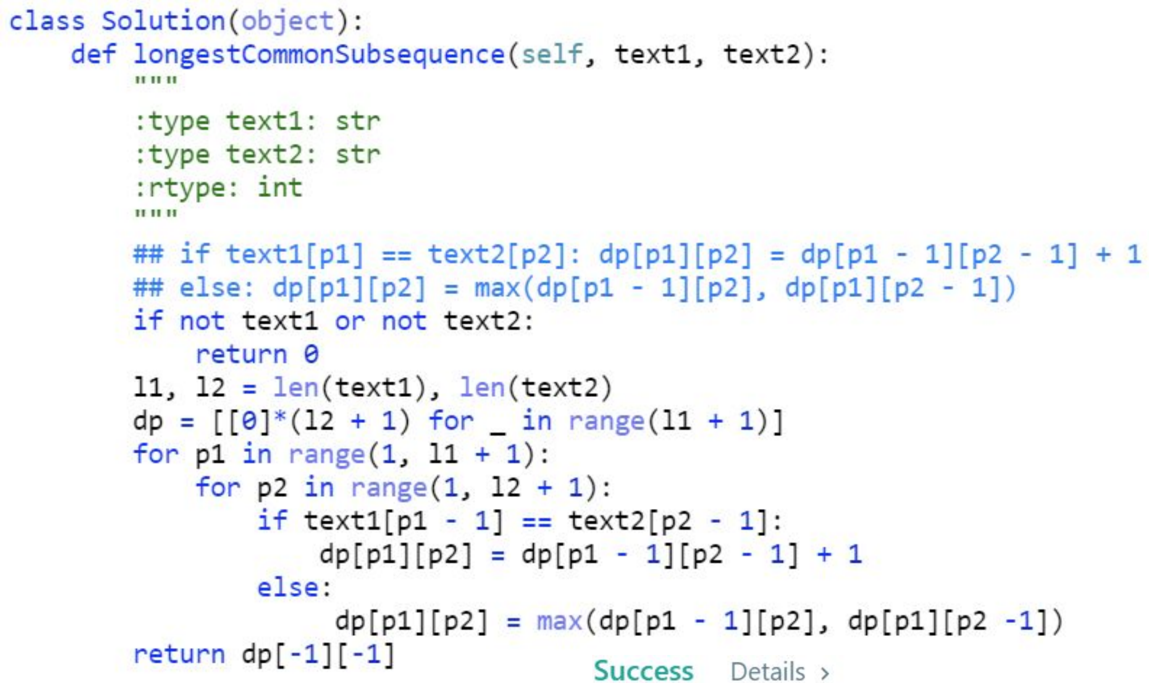
```python
class Solution(object):
    def rob(self, nums):
        """
        :type nums: List[int]
        :rtype: int
        """
        if not nums:
            return 0
        if len(nums) == 1:
            return nums[0]
        n = len(nums)
        def helper(nums, low, high):
            prev = 0
            prev_prev = 0
            curr = 0
            for num in nums[low:high+1]:
                curr = max(prev, prev_prev + num)
                prev_prev = prev
                prev = curr
            return curr

        return max(helper(nums, 0, n - 2), helper(nums, 1, n - 1))
```

Success Details ›

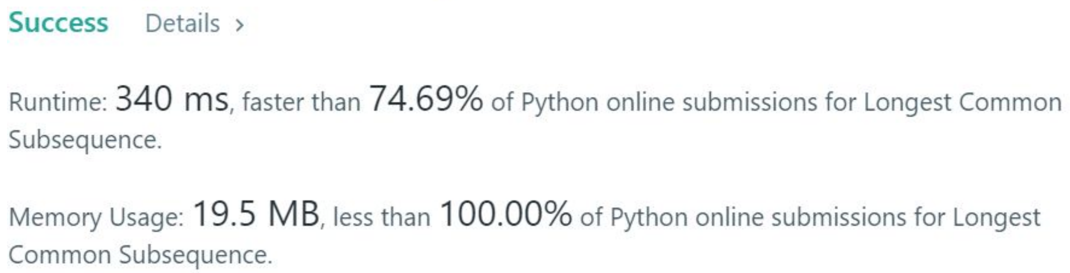
Runtime: 16 ms, faster than 81.94% of Python online submissions for House Robber II.

Memory Usage: 11.8 MB, less than 33.33% of Python online submissions for House Robber II.

LC 1143: Longest Common Subsequence



```python
class Solution(object):
    def longestCommonSubsequence(self, text1, text2):
        """
        :type text1: str
        :type text2: str
        :rtype: int
        """
        ## if text1[p1] == text2[p2]: dp[p1][p2] = dp[p1 - 1][p2 - 1] + 1
        ## else: dp[p1][p2] = max(dp[p1 - 1][p2], dp[p1][p2 - 1])
        if not text1 or not text2:
            return 0
        l1, l2 = len(text1), len(text2)
        dp = [[0]*(l2 + 1) for _ in range(l1 + 1)]
        for p1 in range(1, l1 + 1):
            for p2 in range(1, l2 + 1):
                if text1[p1 - 1] == text2[p2 - 1]:
                    dp[p1][p2] = dp[p1 - 1][p2 - 1] + 1
                else:
                     dp[p1][p2] = max(dp[p1 - 1][p2], dp[p1][p2 -1])
        return dp[-1][-1]
```

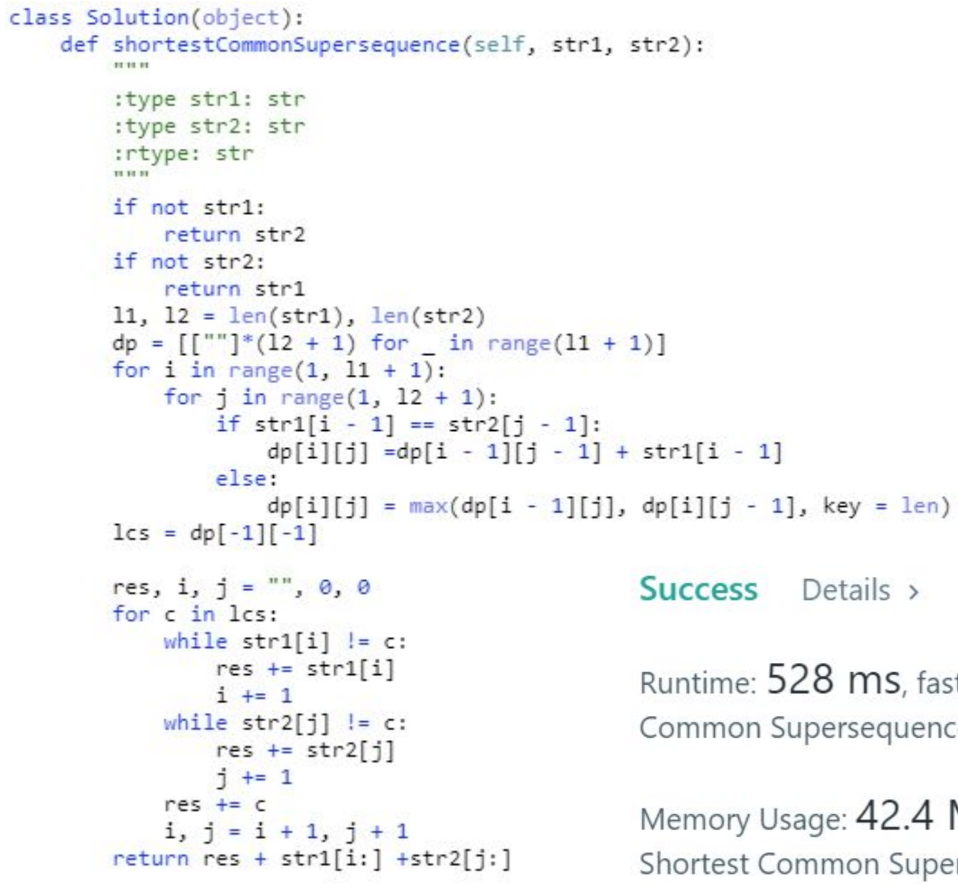


Success Details ›

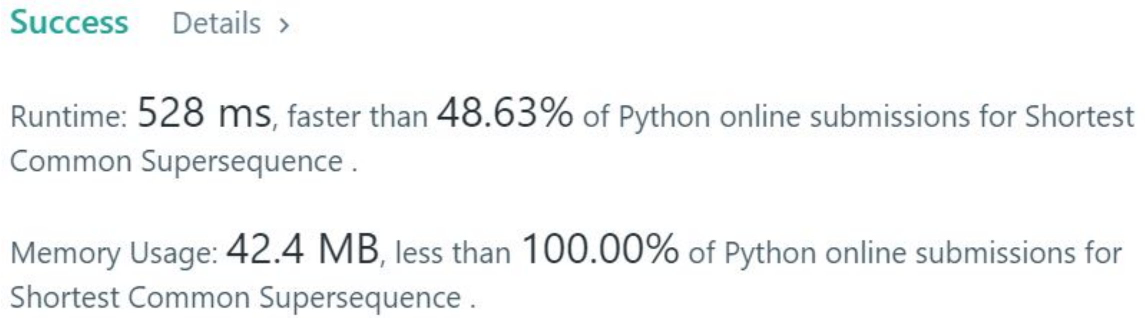
Runtime: 340 ms, faster than 74.69% of Python online submissions for Longest Common Subsequence.

Memory Usage: 19.5 MB, less than 100.00% of Python online submissions for Longest Common Subsequence.

LC 1092: Shortest Common Supersequence



```python
class Solution(object):
    def shortestCommonSupersequence(self, str1, str2):
        """
        :type str1: str
        :type str2: str
        :rtype: str
        """
        if not str1:
            return str2
        if not str2:
            return str1
        l1, l2 = len(str1), len(str2)
        dp = [[""]*(l2 + 1) for _ in range(l1 + 1)]
        for i in range(1, l1 + 1):
            for j in range(1, l2 + 1):
                if str1[i - 1] == str2[j - 1]:
                    dp[i][j] =dp[i - 1][j - 1] + str1[i - 1]
                else:
                    dp[i][j] = max(dp[i - 1][j], dp[i][j - 1], key = len)
        lcs = dp[-1][-1]

        res, i, j = "", 0, 0
        for c in lcs:
            while str1[i] != c:
                res += str1[i]
                i += 1
            while str2[j] != c:
                res += str2[j]
                j += 1
            res += c
            i, j = i + 1, j + 1
        return res + str1[i:] +str2[j:]
```



Success Details ›

Runtime: 528 ms, faster than 48.63% of Python online submissions for Shortest Common Supersequence .

Memory Usage: 42.4 MB, less than 100.00% of Python online submissions for Shortest Common Supersequence .