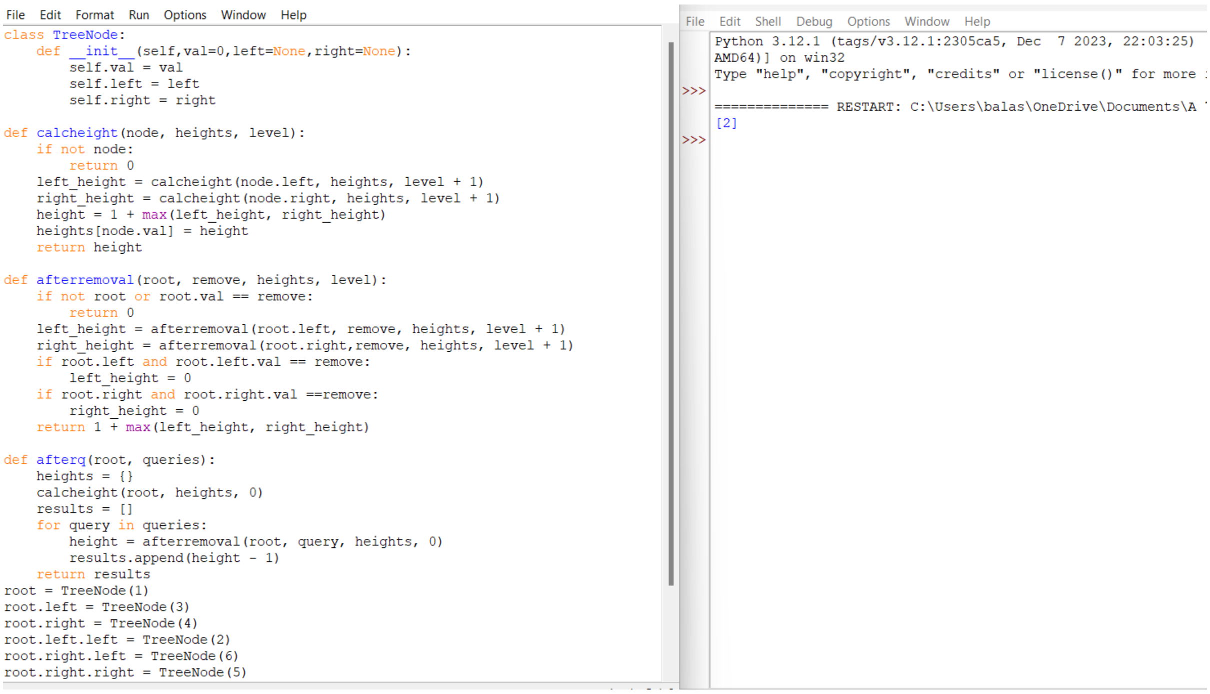
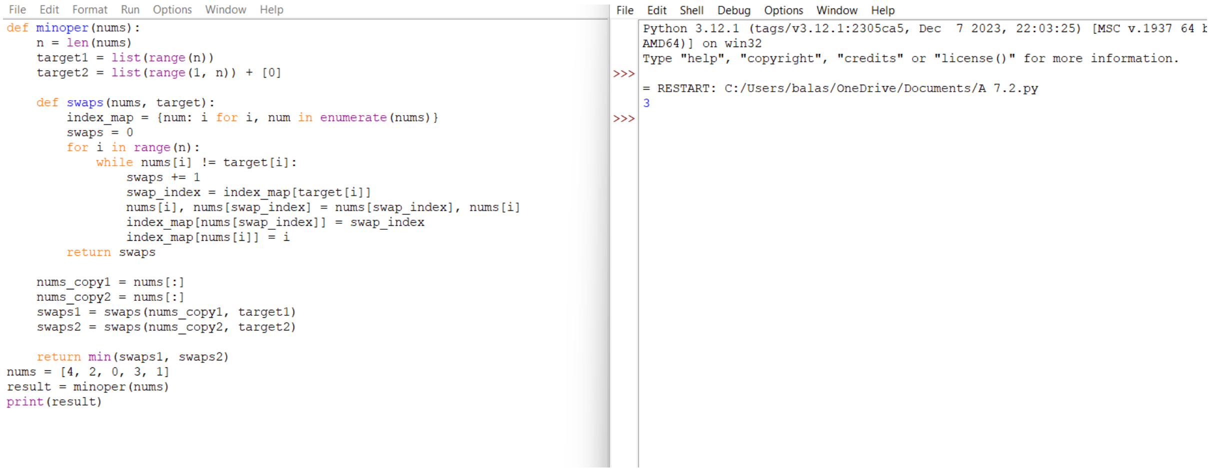
1. .Height of Binary Tree After Subtree Removal Queries You are given the root of a binary tree with n nodes. Each node is assigned a unique value from 1 to n. You are also given an array queries of size m.You have to perform m independent queries on the tree where in the ith query you do the following: ● Remove the subtree rooted at the node with the value queries[i] from the tree. It is guaranteed that queries[i] will not be equal to the value of the root. Return an array answer of size m where answer[i] is the height of the tree after performing the ith query.


File Edit Format Run Options Window Help
File Edit Shell Debug Options Window Help

```
class TreeNode:
    def __init__(self,val=0,left=None,right=None):
        self.val = val
        self.left = left
        self.right = right

def calcheight(node, heights, level):
    if not node:
        return 0
    left_height = calcheight(node.left, heights, level + 1)
    right_height = calcheight(node.right, heights, level + 1)
    height = 1 + max(left_height, right_height)
    heights[node.val] = height
    return height

def afterremoval(root, remove, heights, level):
    if not root or root.val == remove:
        return 0
    left_height = afterremoval(root.left, remove, heights, level + 1)
    right_height = afterremoval(root.right,remove, heights, level + 1)
    if root.left and root.left.val == remove:
        left_height = 0
    if root.right and root.right.val ==remove:
        right_height = 0
    return 1 + max(left_height, right_height)

def afterq(root, queries):
    heights = {}
    calcheight(root, heights, 0)
    results = []
    for query in queries:
        height = afterremoval(root, query, heights, 0)
        results.append(height - 1)
    return results
root = TreeNode(1)
root.left = TreeNode(3)
root.right = TreeNode(4)
root.left.left = TreeNode(2)
root.right.left = TreeNode(6)
root.right.right = TreeNode(5)
```

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25)
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more 
>>>
============== RESTART: C:\Users\balas\OneDrive\Documents\A 
[2]
>>>
```

2. You are given an integer array nums of size n containing each element from 0 to n - 1 (inclusive). Each of the elements from 1 to n - 1 represents an item, and the element 0 represents an empty space. In one operation, you can move any item to the empty space. nums is considered to be sorted if the numbers of all the items are in ascending order and the empty space is either at the beginning or at the end of the array

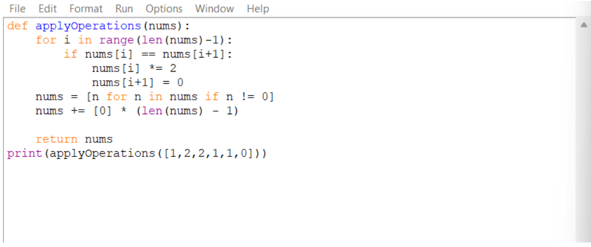

File Edit Format Run Options Window Help
File Edit Shell Debug Options Window Help

```
def minoper(nums):
    n = len(nums)
    target1 = list(range(n))
    target2 = list(range(1, n)) + [0]

    def swaps(nums, target):
        index_map = {num: i for i, num in enumerate(nums)}
        swaps = 0
        for i in range(n):
            while nums[i] != target[i]:
                swaps += 1
                swap_index = index_map[target[i]]
                nums[i], nums[swap_index] = nums[swap_index], nums[i]
                index_map[nums[swap_index]] = swap_index
                index_map[nums[i]] = i
        return swaps

    nums_copy1 = nums[:]
    nums_copy2 = nums[:]
    swaps1 = swaps(nums_copy1, target1)
    swaps2 = swaps(nums_copy2, target2)

    return min(swaps1, swaps2)
nums = [4, 2, 0, 3, 1]
result = minoper(nums)
print(result)
```

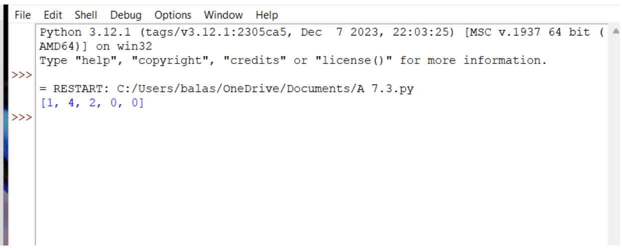
```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/balas/OneDrive/Documents/A 7.2.py
3
>>>
```

3. You are given a 0-indexed array nums of size n consisting of non-negative integers.You need to apply n - 1 operations to this array where, in the ith operation (0-indexed), you will apply the following on the ith element of nums: ● If nums[i] == nums[i + 1], then multiply nums[i] by 2 and set nums[i + 1] to 0. Otherwise, you skip this operation. After performing all the operations, shift all the

0's to the end of the array. ● For example, the array [1,0,2,0,0,1] after shifting all its 0's to the end, is [1,2,1,0,0,0]. Return the resulting array.

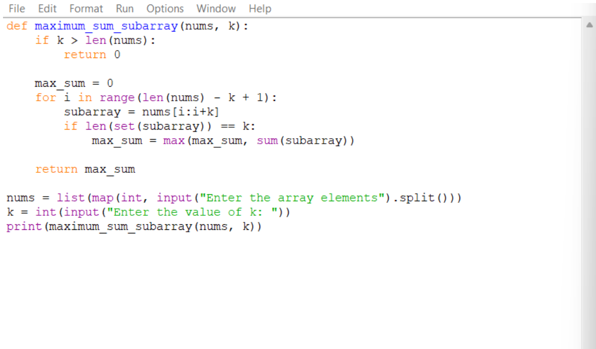


```
def applyOperations(nums):
    for i in range(len(nums)-1):
        if nums[i] == nums[i+1]:
            nums[i] *= 2
            nums[i+1] = 0
    nums = [n for n in nums if n != 0]
    nums += [0] * (len(nums) - 1)

    return nums
print(applyOperations([1,2,2,1,1,0]))
```

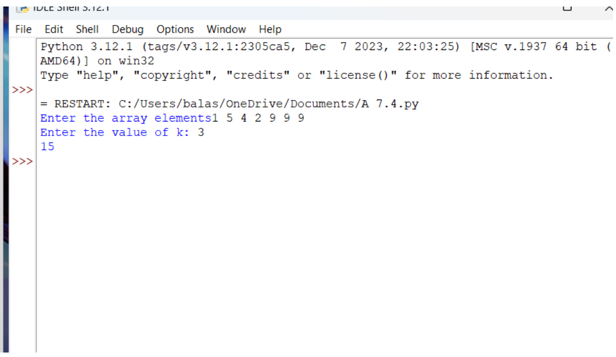


```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/balas/OneDrive/Documents/A 7.3.py
[1, 4, 2, 0, 0]
>>>
```

4. Maximum Sum of Distinct Subarrays With Length K You are given an integer array nums and an integer k. Find the maximum subarray sum of all the subarrays of nums that meet the following conditions: ● The length of the subarray is k, and ● All the elements of the subarray are distinct. Return the maximum subarray sum of all the subarrays that meet the conditions. If no subarray meets the conditions, return 0. A subarray is a contiguous non-empty sequence of elements within an array.

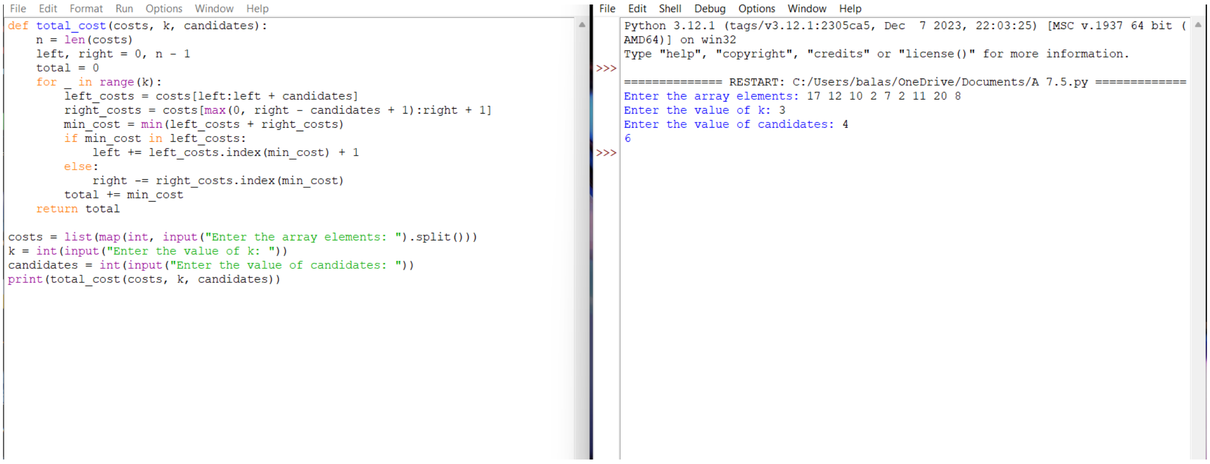


```
def maximum_sum_subarray(nums, k):
    if k > len(nums):
        return 0

    max_sum = 0
    for i in range(len(nums) - k + 1):
        subarray = nums[i:i+k]
        if len(set(subarray)) == k:
            max_sum = max(max_sum, sum(subarray))

    return max_sum

nums = list(map(int, input("Enter the array elements").split()))
k = int(input("Enter the value of k: "))
print(maximum_sum_subarray(nums, k))
```



```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/balas/OneDrive/Documents/A 7.4.py
Enter the array elements1 5 4 2 9 9 9
Enter the value of k: 3
15
>>>
```

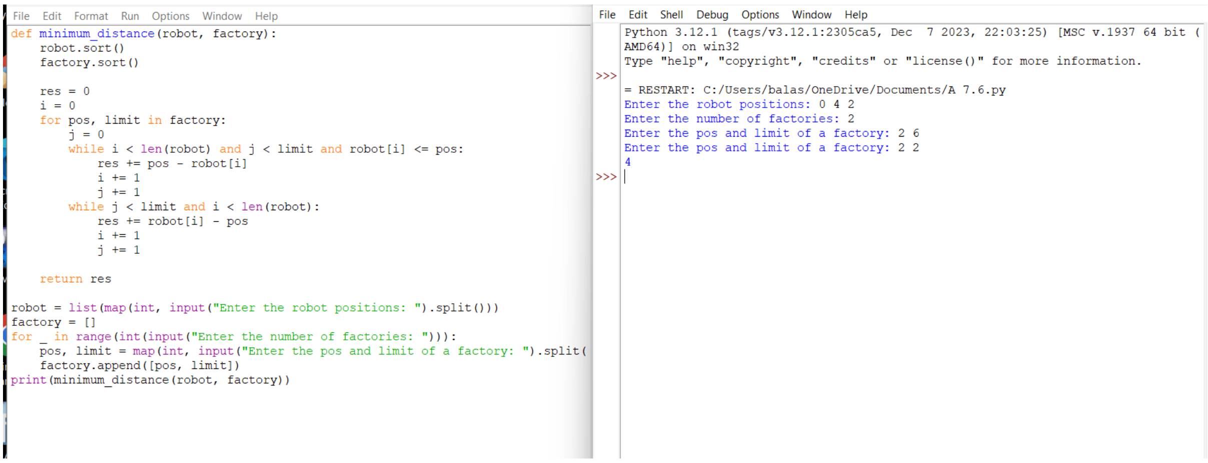
5. Total Cost to Hire K Workers You are given a 0-indexed integer array costs where costs[i] is the cost of hiring the ith worker.You are also given two integers k and candidates. We want to hire exactly k workers according to the following rules: ● You will run k sessions and hire exactly one worker in each session. ● In each hiring session, choose the worker with the lowest cost from either the first candidates workers or the last candidates workers. Break the tie by the smallest index. ○ For example, if costs = [3,2,7,7,1,2] and candidates = 2, then in the first hiring session, we will choose the 4th worker because they have the lowest cost [3,2,7,7,1,2]. ○ In the second hiring session, we will choose 1st worker because they have the same lowest cost as 4th worker but they have the smallest index [3,2,7,7,2]. Please note that the indexing may be changed in the process. ● If there are fewer than candidates workers remaining, choose the worker with the lowest cost among them. Break the tie by the smallest index. ● A worker can only be chosen once.



```
def total_cost(costs, k, candidates):
    n = len(costs)
    left, right = 0, n - 1
    total = 0
    for _ in range(k):
        left_costs = costs[left:left + candidates]
        right_costs = costs[max(0, right - candidates + 1):right + 1]
        min_cost = min(left_costs + right_costs)
        if min_cost in left_costs:
            left += left_costs.index(min_cost) + 1
        else:
            right -= right_costs.index(min_cost)
        total += min_cost
    return total

costs = list(map(int, input("Enter the array elements: ").split()))
k = int(input("Enter the value of k: "))
candidates = int(input("Enter the value of candidates: "))
print(total_cost(costs, k, candidates))
```

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============== RESTART: C:/Users/balas/OneDrive/Documents/A 7.5.py =============
Enter the array elements: 17 12 10 2 7 2 11 20 8
Enter the value of k: 3
Enter the value of candidates: 4
6
>>>
```

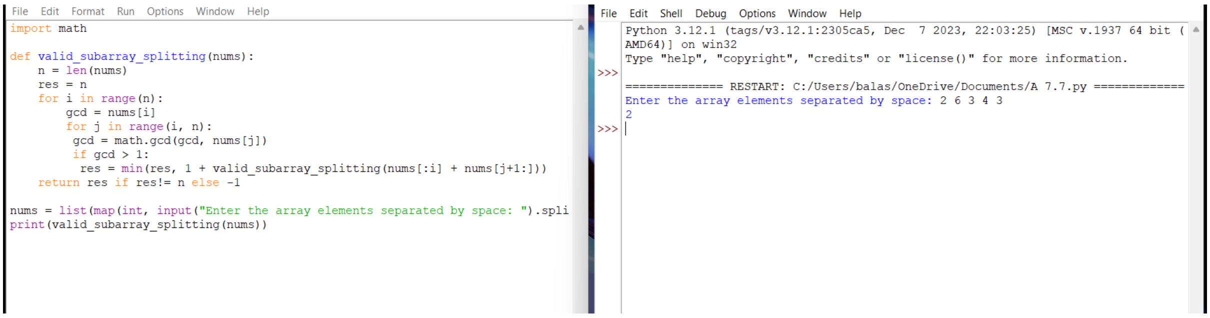
6. . Minimum Total Distance Traveled There are some robots and factories on the X-axis. You are given an integer array robot where robot[i] is the position of the ith robot.



```
def minimum_distance(robot, factory):
    robot.sort()
    factory.sort()

    res = 0
    i = 0
    for pos, limit in factory:
        j = 0
        while i < len(robot) and j < limit and robot[i] <= pos:
            res += pos - robot[i]
            i += 1
            j += 1
        while j < limit and i < len(robot):
            res += robot[i] - pos
            i += 1
            j += 1

    return res

robot = list(map(int, input("Enter the robot positions: ").split()))
factory = []
for _ in range(int(input("Enter the number of factories: "))):
    pos, limit = map(int, input("Enter the pos and limit of a factory: ").split(
    factory.append([pos, limit])
print(minimum_distance(robot, factory))
```

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/balas/OneDrive/Documents/A 7.6.py
Enter the robot positions: 0 4 2
Enter the number of factories: 2
Enter the pos and limit of a factory: 2 6
Enter the pos and limit of a factory: 2 2
4
>>> |
```

7. Minimum Subarrays in a Valid Split You are given an integer array nums.Splitting of an integer array nums into subarrays is valid if: ● the greatest common divisor of the first and last elements of each subarray is greater than 1

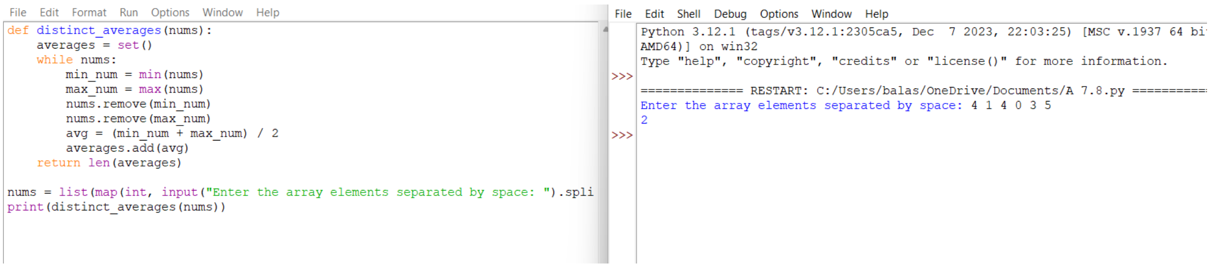

File Edit Format Run Options Window Help

```
import math

def valid_subarray_splitting(nums):
    n = len(nums)
    res = n
    for i in range(n):
        gcd = nums[i]
        for j in range(i, n):
         gcd = math.gcd(gcd, nums[j])
         if gcd > 1:
          res = min(res, 1 + valid_subarray_splitting(nums[:i] + nums[j+1:]))
    return res if res!= n else -1

nums = list(map(int, input("Enter the array elements separated by space: ").spli
print(valid_subarray_splitting(nums))
```

File Edit Shell Debug Options Window Help

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============== RESTART: C:/Users/balas/OneDrive/Documents/A 7.7.py =============
Enter the array elements separated by space: 2 6 3 4 3
2
>>> |
```

8. . Number of Distinct Averages

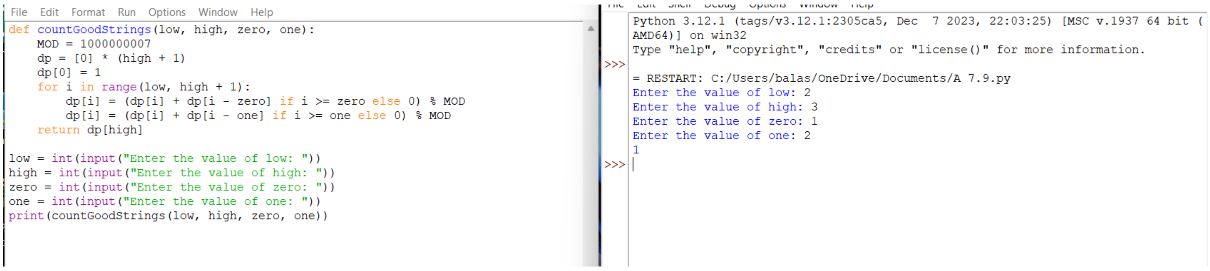

File Edit Format Run Options Window Help

```
def distinct_averages(nums):
    averages = set()
    while nums:
        min_num = min(nums)
        max_num = max(nums)
        nums.remove(min_num)
        nums.remove(max_num)
        avg = (min_num + max_num) / 2
        averages.add(avg)
    return len(averages)

nums = list(map(int, input("Enter the array elements separated by space: ").spli
print(distinct_averages(nums))
```

File Edit Shell Debug Options Window Help

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bi
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
============== RESTART: C:/Users/balas/OneDrive/Documents/A 7.8.py ==========
Enter the array elements separated by space: 4 1 4 0 3 5
2
>>>
```

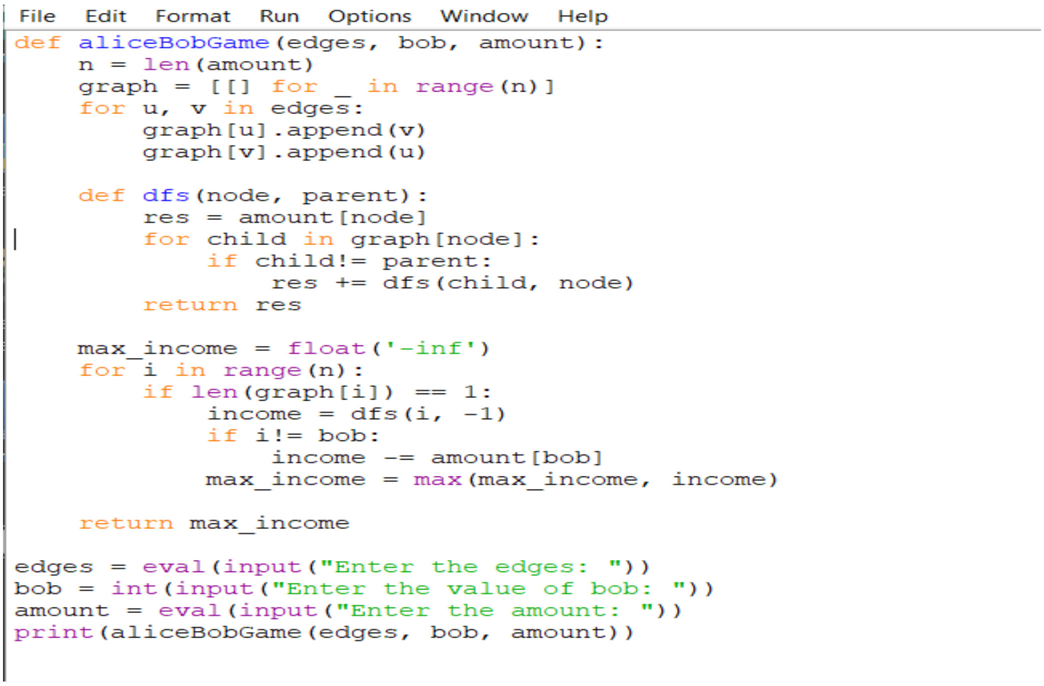
9. 9. Count Ways To Build Good Strings Given the integers zero, one, low, and high, we can construct a string by starting with an empty string, and then at each step perform either of the following: ● Append the character '0' zero times. ● Append the character '1' one times. This can be performed any number of times.A good string is a string constructed by the above process having a length between low and high (inclusive). Return the number of dif erent good strings that can be constructed satisfying these properties. Since the answer can be large, return it



```
def countGoodStrings(low, high, zero, one):
    MOD = 1000000007
    dp = [0] * (high + 1)
    dp[0] = 1
    for i in range(low, high + 1):
        dp[i] = (dp[i] + dp[i - zero] if i >= zero else 0) % MOD
        dp[i] = (dp[i] + dp[i - one] if i >= one else 0) % MOD
    return dp[high]

low = int(input("Enter the value of low: "))
high = int(input("Enter the value of high: "))
zero = int(input("Enter the value of zero: "))
one = int(input("Enter the value of one: "))
print(countGoodStrings(low, high, zero, one))
```

```
Python 3.12.1 (tags/v3.12.1:2305ca5, Dec  7 2023, 22:03:25) [MSC v.1937 64 bit (
AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/balas/OneDrive/Documents/A 7.9.py
Enter the value of low: 2
Enter the value of high: 3
Enter the value of zero: 1
Enter the value of one: 2
1
>>> |
```

10. Most Profitable Path in a Tree There is an undirected tree with n nodes labeled from 0 to n - 1, rooted at node 0. You are given a 2D integer array edges of length n - 1 where edges[i] = [ai, bi] indicates that there is an edge between nodes ai and bi in the tree. At every node i, there is a gate. You are also given an array of even integers amount, where amount[i] represents: ● the price needed to open the gate at node i, if amount[i] is negative, or, ● the cash reward obtained on opening the gate at node i, otherwise. The game goes on as follows: ● Initially, Alice is at node 0 and Bob is at node bob. ● At every second, Alice and Bob each move to an adjacent node. Alice moves towards some leaf node, while Bob moves towards node 0. ● For every node along their path, Alice and Bob



```
def aliceBobGame(edges, bob, amount):
    n = len(amount)
    graph = [[] for _ in range(n)]
    for u, v in edges:
        graph[u].append(v)
        graph[v].append(u)

    def dfs(node, parent):
        res = amount[node]
        for child in graph[node]:
            if child!= parent:
                res += dfs(child, node)
        return res

    max_income = float('-inf')
    for i in range(n):
        if len(graph[i]) == 1:
            income = dfs(i, -1)
            if i!= bob:
                income -= amount[bob]
            max_income = max(max_income, income)

    return max_income

edges = eval(input("Enter the edges: "))
bob = int(input("Enter the value of bob: "))
amount = eval(input("Enter the amount: "))
print(aliceBobGame(edges, bob, amount))
```