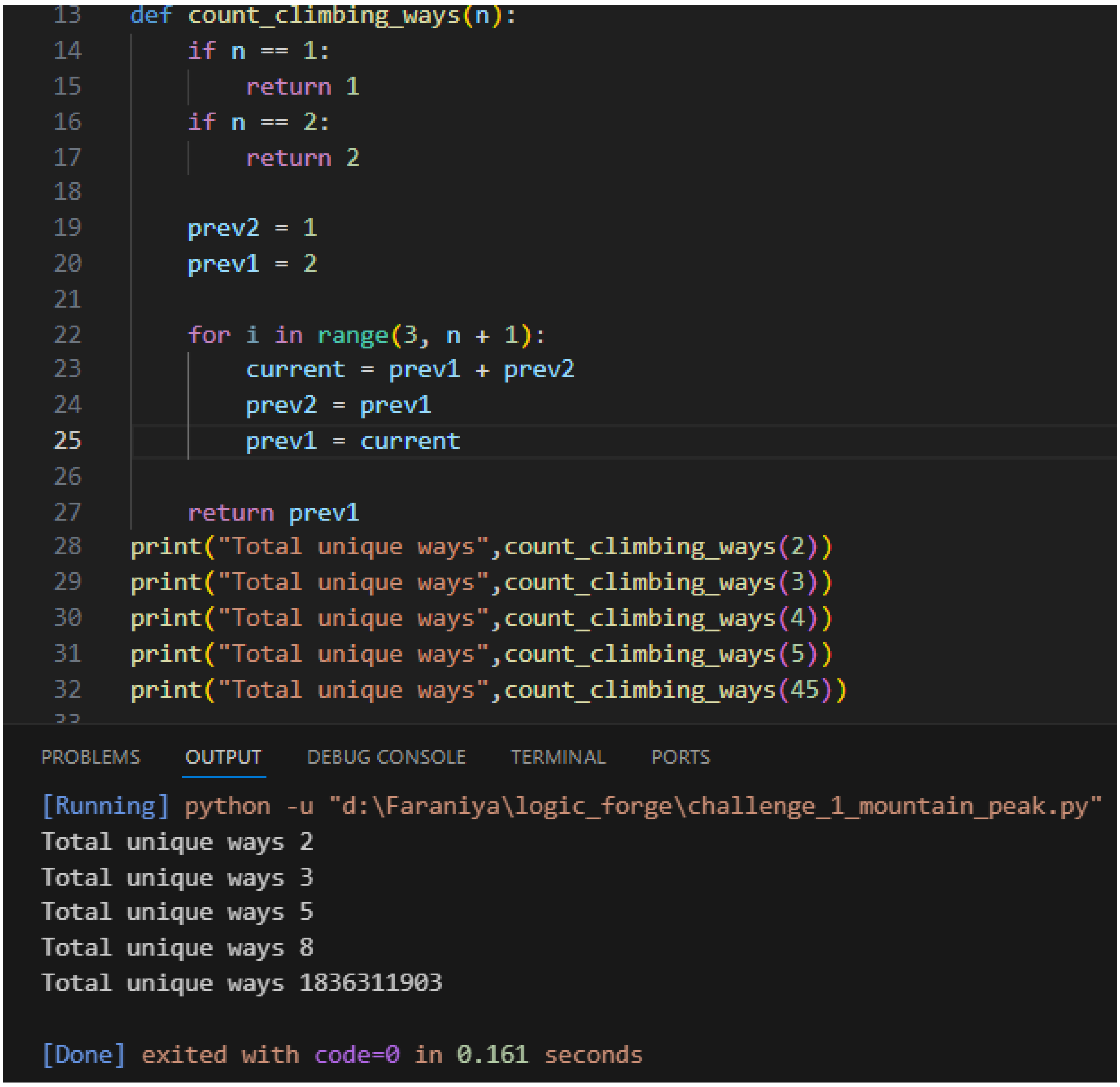
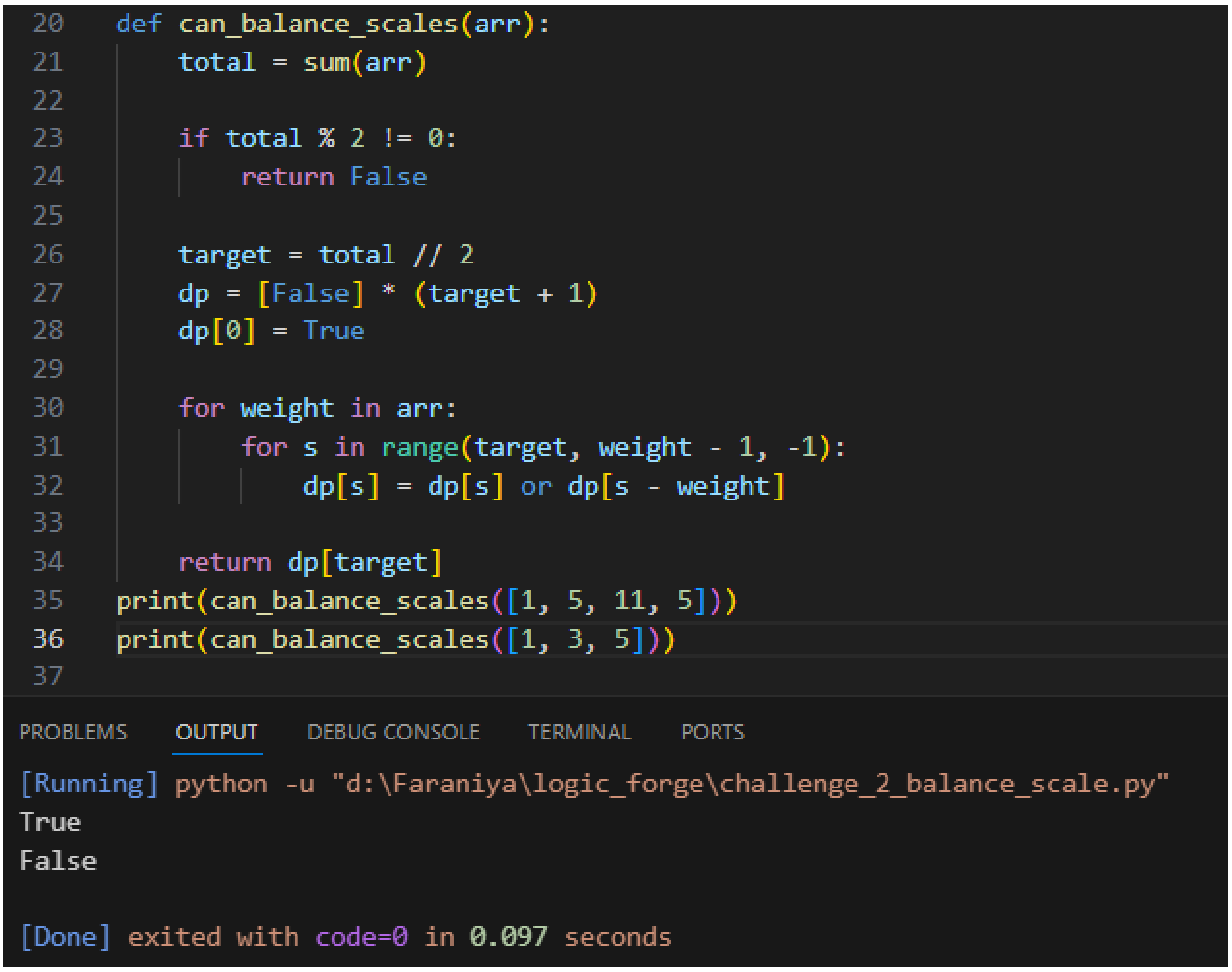
OUTPUT:01



```python
def count_climbing_ways(n):
    if n == 1:
        return 1
    if n == 2:
        return 2

    prev2 = 1
    prev1 = 2

    for i in range(3, n + 1):
        current = prev1 + prev2
        prev2 = prev1
        prev1 = current

    return prev1
print("Total unique ways",count_climbing_ways(2))
print("Total unique ways",count_climbing_ways(3))
print("Total unique ways",count_climbing_ways(4))
print("Total unique ways",count_climbing_ways(5))
print("Total unique ways",count_climbing_ways(45))
```

```
[Running] python -u "d:\Faraniya\logic_forge\challenge_1_mountain_peak.py"
Total unique ways 2
Total unique ways 3
Total unique ways 5
Total unique ways 8
Total unique ways 1836311903

[Done] exited with code=0 in 0.161 seconds
```

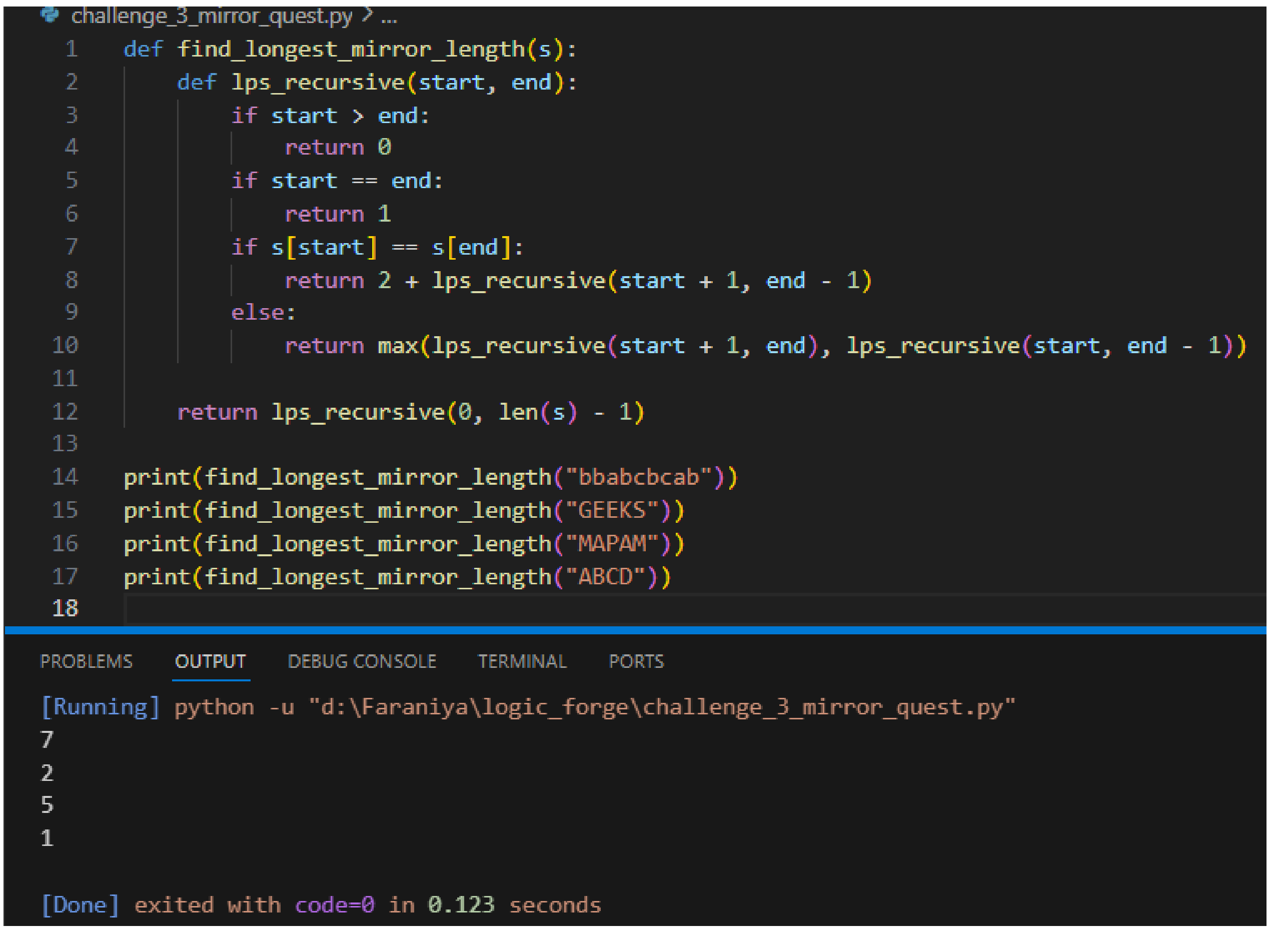
OUTPUT:02



```python
def can_balance_scales(arr):
    total = sum(arr)

    if total % 2 != 0:
        return False

    target = total // 2
    dp = [False] * (target + 1)
    dp[0] = True

    for weight in arr:
        for s in range(target, weight - 1, -1):
            dp[s] = dp[s] or dp[s - weight]

    return dp[target]
print(can_balance_scales([1, 5, 11, 5]))
print(can_balance_scales([1, 3, 5]))
```

```
[Running] python -u "d:\Faraniya\logic_forge\challenge_2_balance_scale.py"
True
False

[Done] exited with code=0 in 0.097 seconds
```

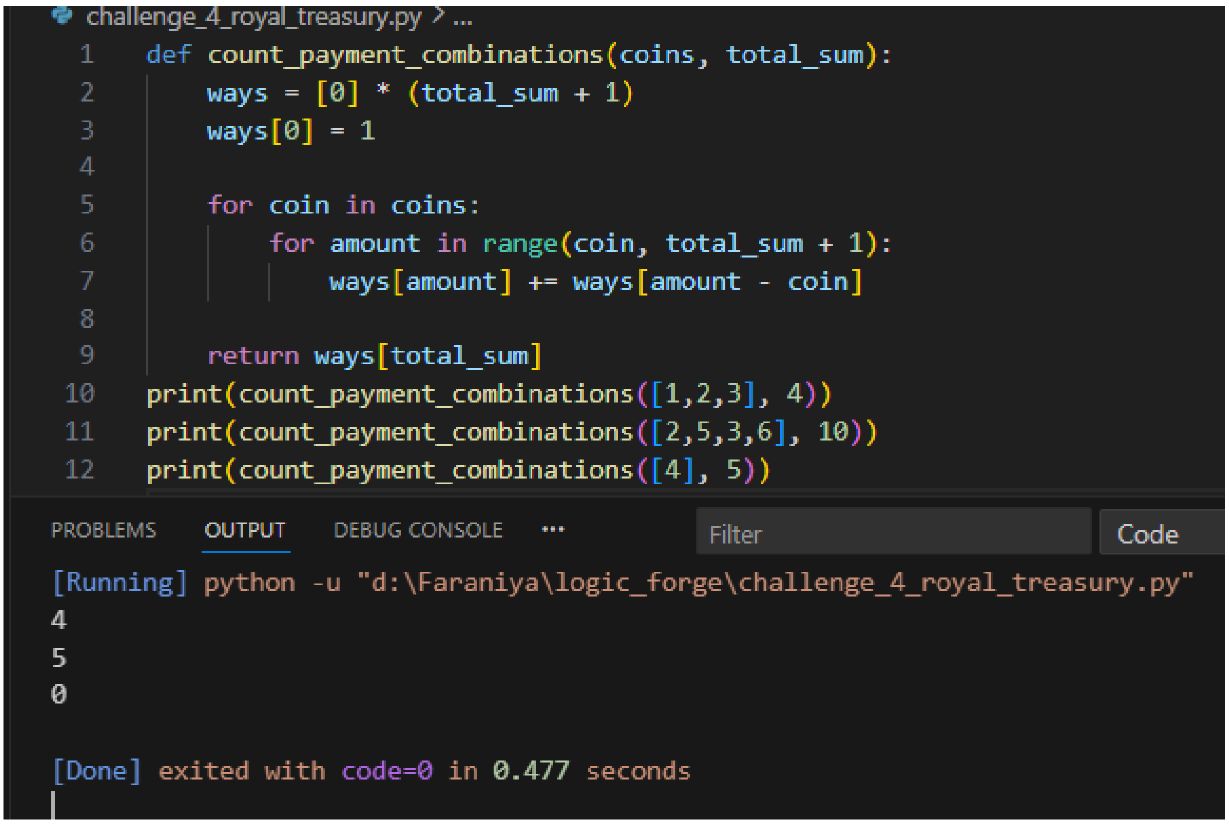
OUTPUT:03



```python
def find_longest_mirror_length(s):
    def lps_recursive(start, end):
        if start > end:
            return 0
        if start == end:
            return 1
        if s[start] == s[end]:
            return 2 + lps_recursive(start + 1, end - 1)
        else:
            return max(lps_recursive(start + 1, end), lps_recursive(start, end - 1))

    return lps_recursive(0, len(s) - 1)

print(find_longest_mirror_length("bbabcbcab"))
print(find_longest_mirror_length("GEEKS"))
print(find_longest_mirror_length("MAPAM"))
print(find_longest_mirror_length("ABCD"))
```

```
[Running] python -u "d:\Faraniya\logic_forge\challenge_3_mirror_quest.py"
7
2
5
1

[Done] exited with code=0 in 0.123 seconds
```

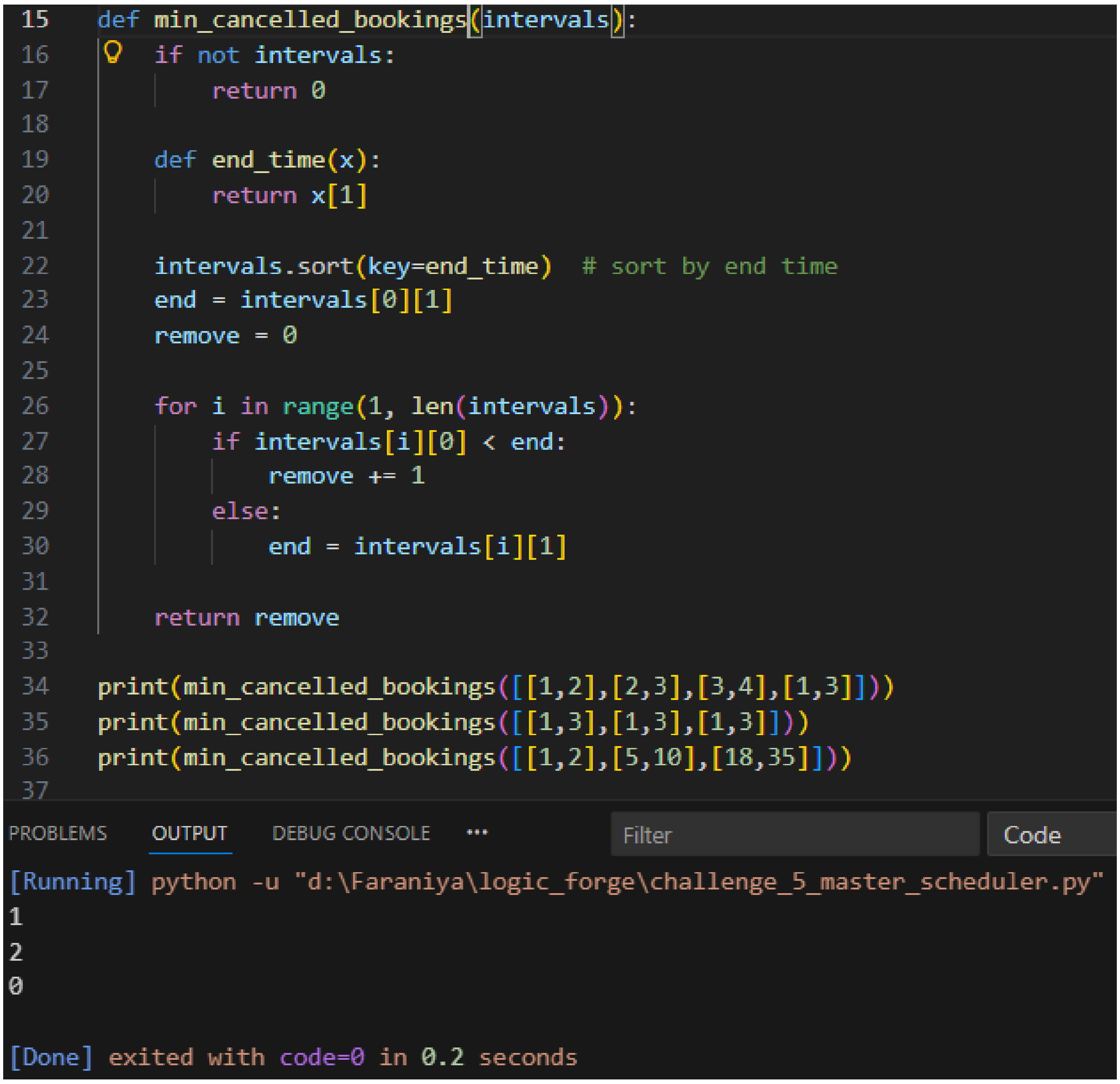
OUTPUT:04



```python
def count_payment_combinations(coins, total_sum):
    ways = [0] * (total_sum + 1)
    ways[0] = 1

    for coin in coins:
        for amount in range(coin, total_sum + 1):
            ways[amount] += ways[amount - coin]

    return ways[total_sum]
print(count_payment_combinations([1,2,3], 4))
print(count_payment_combinations([2,5,3,6], 10))
print(count_payment_combinations([4], 5))
```

```
[Running] python -u "d:\Faraniya\logic_forge\challenge_4_royal_treasury.py"
4
5
0

[Done] exited with code=0 in 0.477 seconds
```

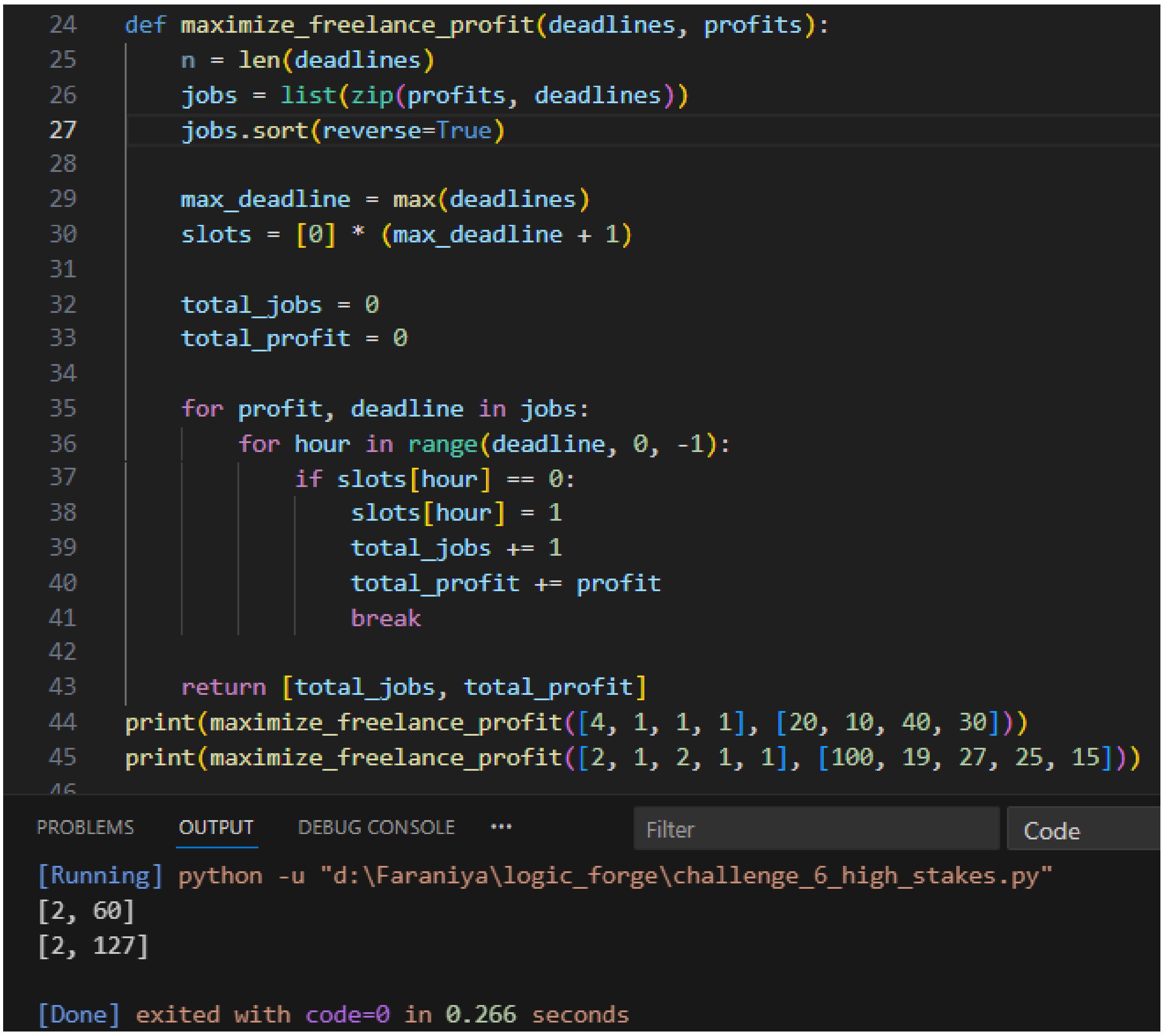
OUTPUT:05



```python
def min_cancelled_bookings(intervals):
    if not intervals:
        return 0

    def end_time(x):
        return x[1]

    intervals.sort(key=end_time)  # sort by end time
    end = intervals[0][1]
    remove = 0

    for i in range(1, len(intervals)):
        if intervals[i][0] < end:
            remove += 1
        else:
            end = intervals[i][1]

    return remove

print(min_cancelled_bookings([[1,2],[2,3],[3,4],[1,3]]))
print(min_cancelled_bookings([[1,3],[1,3],[1,3]]))
print(min_cancelled_bookings([[1,2],[5,10],[18,35]]))
```

```
[Running] python -u "d:\Faraniya\logic_forge\challenge_5_master_scheduler.py"
1
2
0

[Done] exited with code=0 in 0.2 seconds
```

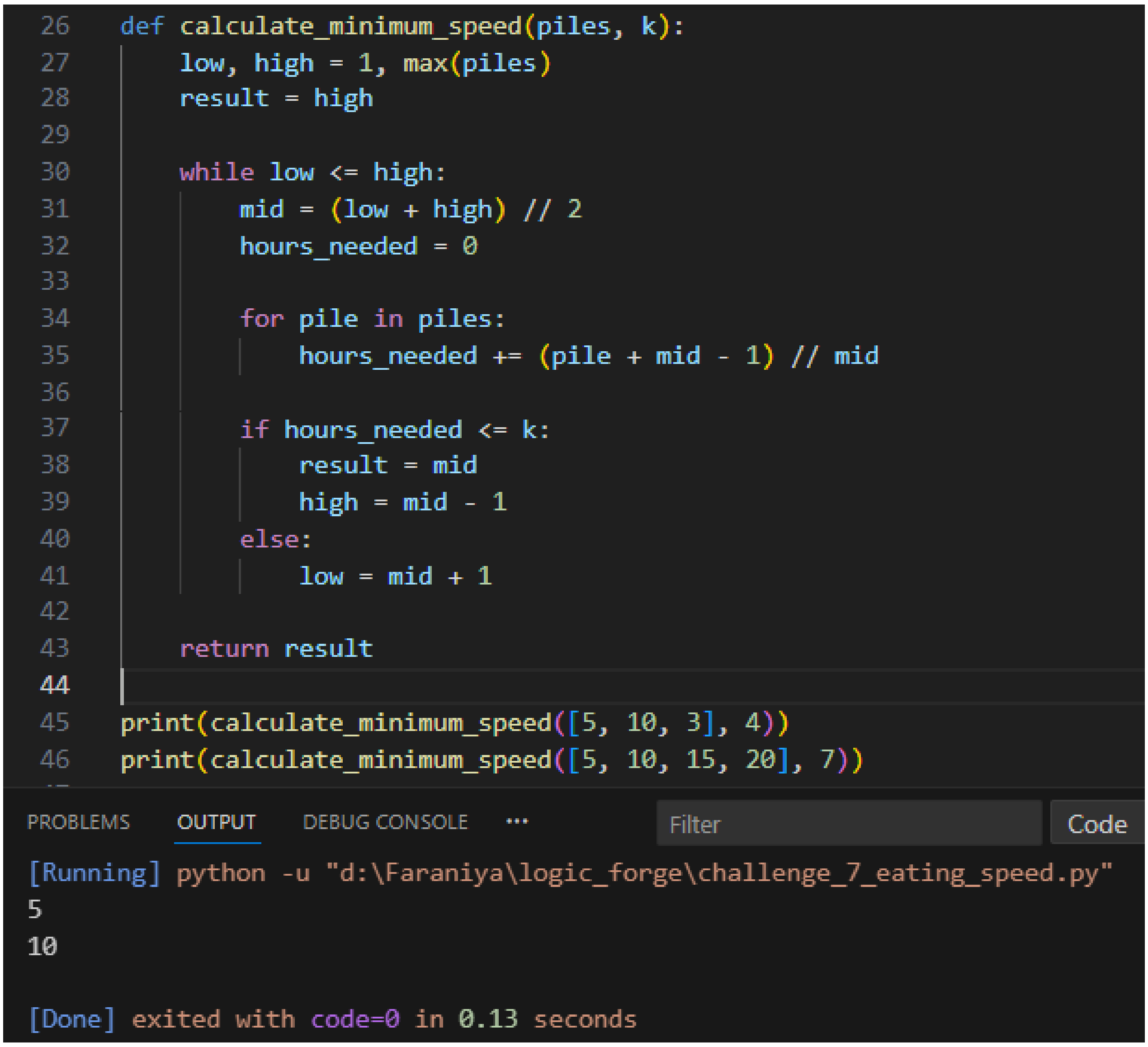
OUTPUT:06



```python
def maximize_freelance_profit(deadlines, profits):
    n = len(deadlines)
    jobs = list(zip(profits, deadlines))
    jobs.sort(reverse=True)

    max_deadline = max(deadlines)
    slots = [0] * (max_deadline + 1)

    total_jobs = 0
    total_profit = 0

    for profit, deadline in jobs:
        for hour in range(deadline, 0, -1):
            if slots[hour] == 0:
                slots[hour] = 1
                total_jobs += 1
                total_profit += profit
                break

    return [total_jobs, total_profit]
print(maximize_freelance_profit([4, 1, 1, 1], [20, 10, 40, 30]))
print(maximize_freelance_profit([2, 1, 2, 1, 1], [100, 19, 27, 25, 15]))
```

```
[Running] python -u "d:\Faraniya\logic_forge\challenge_6_high_stakes.py"
[2, 60]
[2, 127]

[Done] exited with code=0 in 0.266 seconds
```

OUTPUT:07



```python
def calculate_minimum_speed(piles, k):
    low, high = 1, max(piles)
    result = high

    while low <= high:
        mid = (low + high) // 2
        hours_needed = 0

        for pile in piles:
            hours_needed += (pile + mid - 1) // mid

        if hours_needed <= k:
            result = mid
            high = mid - 1
        else:
            low = mid + 1

    return result

print(calculate_minimum_speed([5, 10, 3], 4))
print(calculate_minimum_speed([5, 10, 15, 20], 7))
```

```
[Running] python -u "d:\Faraniya\logic_forge\challenge_7_eating_speed.py"
5
10

[Done] exited with code=0 in 0.13 seconds
```