NAME : Parth Pathak

A6-B3-47

PRACT 4- DAA

[11]
✓ 0s

```
def max_sum_subarray(arr):
    max_sum = current_sum = arr[0]
    start = end = temp_start = 0

    for i in range(1, 6):
        if arr[i] > current_sum + arr[i]:
            current_sum = arr[i]
            temp_start = i
        else:
            current_sum += arr[i]

        if current_sum > max_sum:
            max_sum = current_sum
            start = temp_start
            end = i

    return arr[start:end+1], max_sum
    resources = [2,-4,3,-1,5,-6]
subarray, total = max_sum_subarray(resources)
print("Max sum subarray:", subarray)
print("Sum:", total)
```

```
Max sum subarray: [3, -1, 5]
Sum: 7
```

[23]
0s

```
resources = [2, -4, 3, -1, 5, -6]
mid1 = len(resources) // 2
left = resources[:mid1]
right = resources[mid1:]
mid2 = len(left) // 2
left_of_left = left[:mid2]
right_of_left = left[mid2:]

print("Full array:", resources)
print("Left:", left)
print("Right:", right)
print("Left(left)", left_of_left)
print("Right(right)", right_of_left)
```

```
Full array: [2, -4, 3, -1, 5, -6]
Left: [2, -4, 3]
Right: [-1, 5, -6]
Left(left) [2]
Right(right) [-4, 3]
```

[48]
0s

```
final_sum = max_subarray(resources, 0, len(resources)-1)
print("Final max subarray sum:", final_sum)
```

```
Final max subarray sum: 8
```

TEST CASES:

1.

```python
resources = [2, 1, 3, 4]
constraint = 5

def best_subarray(arr, limit):
    best = []
    max_sum = 0

    for start in range(len(arr)):
        current = []
        total = 0
        for end in range(start, len(arr)):
            total += arr[end]
            current.append(arr[end])
            if total <= limit and total > max_sum:
                best = current[:]
                max_sum = total

    return best, max_sum

subarray, total = best_subarray(resources, constraint)
print("1: Best subarray:", subarray)
print("Sum:", total)
```

```
1: Best subarray: [1, 3]
Sum: 4
```

2.

```
[47]
✓ 0s
```

```python
resources = [2, 2, 2, 2]
constraint = 4

subarray, total = best_subarray(resources, constraint)
print("2: Exact match to constraint")
print("Best subarray:", subarray)
print("Sum:", total)
```

```
2: Exact match to constraint
Best subarray: [2, 2]
Sum: 4
```

3:

[38]
✓ 0s

```
resources = [1, 5, 2, 3]
constraint = 5

subarray, total = best_subarray(resources, constraint)
print("3:Single element equals constraint")
print("Best subarray:", subarray)
print("Sum:", total)
```

```
3:Single element equals constraint
Best subarray: [5]
Sum: 5
```

4:

[40]
✓ 0s

```
resources = [6, 7, 8]
constraint = 5

subarray, total = best_subarray(resources, constraint)
print("4: All elements larger than constraint")
if subarray:
    print("Best subarray:", subarray)
    print("Sum:", total)
else:
    print("No feasible subarray found.")
```

```
4: All elements larger than constraint
No feasible subarray found.
```

5:

[41]
✓ 0s

```
resources = [1, 2, 3, 2, 1]
constraint = 5

subarray, total = best_subarray(resources, constraint)
print("5: Multiple optimal subarrays")
print("Best subarray:", subarray)
print("Sum:", total)
```

```
5: Multiple optimal subarrays
Best subarray: [2, 3]
Sum: 5
```

6:

[33]
✓ 0s

```
resources = [1, 1, 1, 1, 1]
constraint = 4

def best_subarray(arr, constraint):
    max_sum = float('-inf')
    best = []
    for i in range(len(arr)):
        total = 0
        for j in range(i, len(arr)):
            total += arr[j]
            if total <= constraint and total > max_sum:
                max_sum = total
                best = arr[i:j+1]
    return best, max_sum

subarray, total = best_subarray(resources, constraint)
print("6: Large window valid")
print("Best subarray:", subarray)
print("Sum:", total)
```

```
6: Large window valid
Best subarray: [1, 1, 1, 1]
Sum: 4
```

7:

[53]
✓ 0s

```
def best_subarray(arr, k):
    left = total = max_sum = 0
    best = []

    for right in range(len(arr)):
        total += arr[right]
        while total > k:
            total -= arr[left]
            left += 1
        if total > max_sum:
            max_sum = total
            best = arr[left:right+1]
    return best, max_sum

resources = [4, 2, 3, 1]
constraint = 5

subarray, total = best_subarray(resources, constraint)
print("7: Best subarray:", subarray)
print("Sum:", total)
```

```
7: Best subarray: [2, 3]
Sum: 5
```

8:

[44]
✓ 0s

```
resources = []
constraint = 10

subarray, total = best_subarray(resources, constraint)
print("8: Empty array")
if subarray:
    print("Best subarray:", subarray)
    print("Sum:", total)
else:
    print("No feasible subarray found.")
```

```
8: Empty array
No feasible subarray found.
```

9:

[45]
✓ 0s

```
resources = [1, 2, 3]
constraint = 0

subarray, total = best_subarray(resources, constraint)
print("9: Constraint = 0")
if subarray:
    print("Best subarray:", subarray)
    print("Sum:", total)
else:
    print("No feasible subarray found.")
```

```
9: Constraint = 0
No feasible subarray found.
```

10: