DESIGN AND ANALYSIS OF ALGORITHM

PRACTICAL-7

NAME: VEDANT BHUTADA

ROLL: 69

BATCH: A4

```
def maxSubArraySum(arr, l, h):
    if (l > h):
        return -10000, []

    if (l == h):
        return arr[l], [arr[l]]

    m = (l + h) // 2

    left_max, left_subarray = maxSubArraySum(arr, l, m - 1)
    right_max, right_subarray = maxSubArraySum(arr, m + 1, h)
    cross_max, cross_subarray = maxCrossingSum(arr, l, m, h)

    if left_max >= right_max and left_max >= cross_max:
        return left_max, left_subarray
    elif right_max >= left_max and right_max >= cross_max:
        return right_max, right_subarray
    else:
        return cross_max, cross_subarray

def maxCrossingSum(arr, l, m, h):
    sm = 0
    left_sum = -10000
    max_left = m
    for i in range(m, l - 1, -1):
        sm = sm + arr[i]
        if sm > left_sum:
            left_sum = sm
            max_left = i

    sm = 0
    right_sum = -10000
    max_right = m
    for i in range(m, h + 1):
        sm = sm + arr[i]
        if sm > right_sum:
            right_sum = sm
            max_right = i

    return left_sum + right_sum - arr[m], arr[max_left : max_right + 1]

#arr = [5, -3, 9, 12, -8, 7, 11, -9, 1, -2, 4, 6]
#arr=[-2,-5,6,-2,-3,1,5,-6]
arr=[-2,1,-3,4,-1,2,1,-5,4]
n = len(arr)

max_sum, max_subarray = maxSubArraySum(arr, 0, n - 1)
print("Maximum contiguous sum is", max_sum)
print("Subarray with maximum sum is", max_subarray)
```

```
Maximum contiguous sum is 6
Subarray with maximum sum is [4, -1, 2, 1]
```