**Name: Shreya Agrawal**

**Class: A5\_B3**

**Roll no.: 38**

**DAA PRACTICAL 4**

***Code:***

def maximum\_sum\_subarray(arr, constraint):

if not arr:

print("No feasible subarray")

return

def max\_cross\_sum(arr, mid):

left\_sum = -999999

s = 0

left\_index = mid

for i in range(mid, -1, -1):

s += arr[i]

if s > left\_sum:

left\_sum = s

left\_index = i

right\_sum = -999999

s = 0

right\_index = mid + 1

for i in range(mid + 1, len(arr)):

s += arr[i]

if s > right\_sum:

right\_sum = s

right\_index = i

cross\_sum = left\_sum + right\_sum

print("Left Sum:", left\_sum, "Right Sum:", right\_sum, "Cross Sum:", cross\_sum)

return cross\_sum, arr[left\_index:right\_index + 1]

def divide(arr):

if len(arr) == 1:

if arr[0] <= constraint:

return arr[0], arr

else:

return -999999, []

mid = len(arr) // 2

left\_sum, left\_subarray = divide(arr[:mid])

right\_sum, right\_subarray = divide(arr[mid:])

cross\_sum, cross\_subarray = max\_cross\_sum(arr, mid - 1)

max\_sum = left\_sum

max\_subarray = left\_subarray

if right\_sum > max\_sum:

max\_sum, max\_subarray = right\_sum, right\_subarray

if cross\_sum > max\_sum:

max\_sum, max\_subarray = cross\_sum, cross\_subarray

if max\_sum > constraint:

return -999999, []

return max\_sum, max\_subarray

max\_sum, max\_subarray = divide(arr)

if max\_sum <= 0:

print("No feasible subarray")

else:

print("Maximum Sum:", max\_sum)

print("Subarray:", max\_subarray)

print("Test 1:"); maximum\_sum\_subarray([2, 1, 3, 4], 5)

print("Test 2:"); maximum\_sum\_subarray([2, 2, 2, 2], 4)

print("Test 3:"); maximum\_sum\_subarray([1, 5, 2, 3], 5)

print("Test 4:"); maximum\_sum\_subarray([6, 7, 8], 5)

print("Test 5:"); maximum\_sum\_subarray([1, 2, 3, 2, 1], 5)

print("Test 6:"); maximum\_sum\_subarray([1, 1, 1, 1, 1], 4)

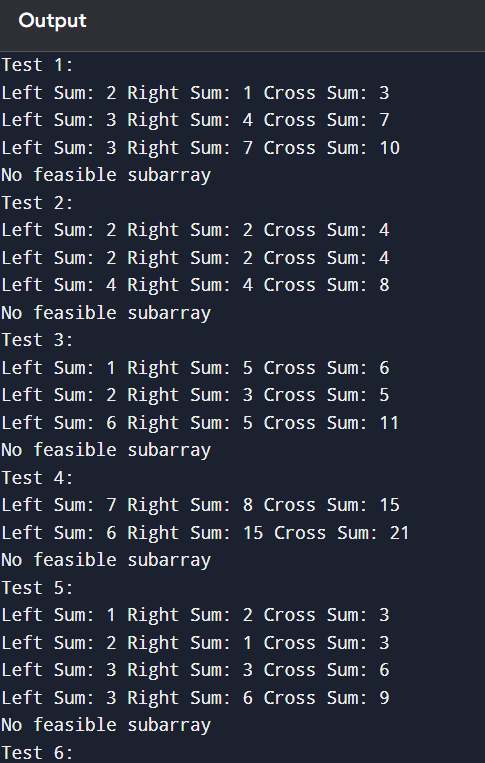
print("Test 7:"); maximum\_sum\_subarray([4, 2, 3, 1], 5)

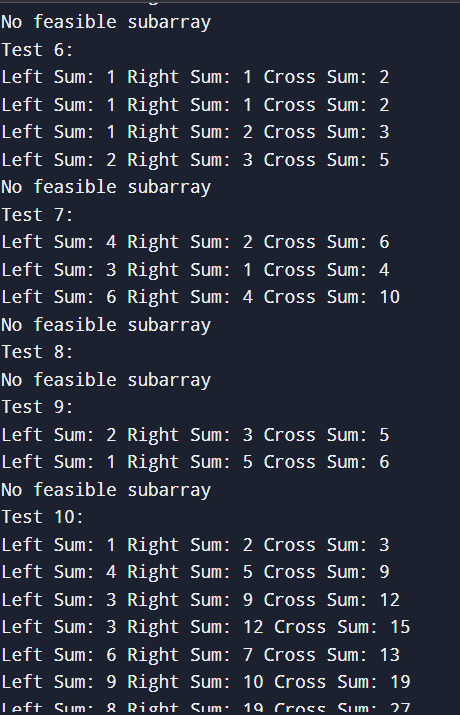
print("Test 8:"); maximum\_sum\_subarray([], 10)

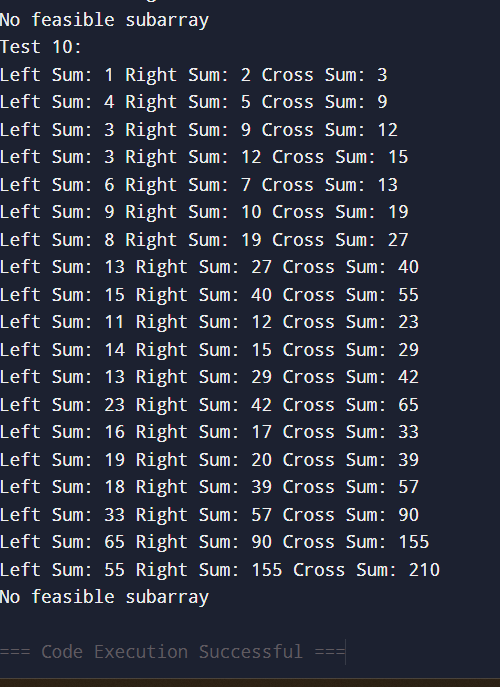
print("Test 9:"); maximum\_sum\_subarray([1, 2, 3], 0)

print("Test 10:"); maximum\_sum\_subarray(list(range(1, 21)), 50)

**Output:**

****

****

****

**LEETCODE:**

int maxSubArray(int\* nums, int numsSize) {

int current\_sum = nums[0];

int max\_sum = nums[0];

for (int i = 1; i < numsSize; i++) {

// Either extend current subarray or start new subarray

if (current\_sum + nums[i] > nums[i])

current\_sum = current\_sum + nums[i];

else

current\_sum = nums[i];

if (current\_sum > max\_sum)

max\_sum = current\_sum;

}

return max\_sum;

}

