CS 310 'Algorithms' Fall 2019 **Quiz-3A** [Total marks=8] [Time: 20 mins]

Name: _____ Roll No: _____

Q1. Following is a description of the 'Maximum Subarray Sum' problem. You are given an array `A[0..n-1]` of `n` integers. These integers may be positive or negative. Let `sum(i,j)` denote the sum of integers `A[i … j]`, where `i≤j`.

(i.e. $sum(i,j) = \sum_{k=i}^{j} A[k]$)

You have to find indices max_i and max_j such that:

`maxSum = sum(max_i,max_j) = max(sum(i,j) | 0≤i≤j<n)`

Following is an **incomplete** divide and conquer pseudocode for finding maximum subarray. **You have to fill in the missing statements in the code. You are not allowed to write more than one statement in each blank.** **[8 marks]**

```
FIND-MAXIMUM-SUBARRAY(A, low, high)
1  if high == low
2      return (low, high, A[low])          // base case: only one element
3  else mid = ⌊(low + high)/2⌋
4      (left-low, left-high, left-sum) =
           FIND-MAXIMUM-SUBARRAY(A, low, mid)
5      (right-low, right-high, right-sum) =
           FIND-MAXIMUM-SUBARRAY(A, mid + 1, high)
6      (cross-low, cross-high, cross-sum) =
           FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
7      if left-sum ≥ right-sum and left-sum ≥ cross-sum
8          return (left-low, left-high, left-sum)
9      elseif right-sum ≥ left-sum and right-sum ≥ cross-sum
10         return (right-low, right-high, right-sum)
11     else return (cross-low, cross-high, cross-sum)
```

```
FIND-MAX-CROSSING-SUBARRAY(A, low, mid, high)
1   left-sum = −∞
2   sum = 0
3   for i = mid downto low
4       sum = sum + A[i]
5       if sum > left-sum
6           left-sum = sum
7           max-left = i
8   right-sum = −∞
9   sum = 0
10  for j = mid + 1 to high
11      sum = sum + A[j]
12      if sum > right-sum
13          right-sum = sum
14          max-right = j
15  return (max-left, max-right, left-sum + right-sum)
```

What is the recurrence relation for Find-Maximum-Subarray?

T(1) = O(1) and T(n) = 2T(n/2) +O(n) if n>1