O(n) Dynamic Programming Algorithm (Kadane's Algorithm)

```
1.  MaxSubArraySum(A,n)
2.  {   globalSum = A[1]
3.      MaxSum[1] = A[1]
4.      for (i = 2 to n)
5.          if (MaxSum[i-1] + A[i] > A[i] )
6.              MaxSum[i] = MaxSum[i-1] + A[i]
7.          else
8.              MaxSum[i] = A[i]
9.          If (globalSum < MaxSum[i])
10.             globalSum = MaxSum[i]
11.             globalEnd = i
12.     return globalSum
13. }
```

Task 1:

This algorithm keeps track of end of Max sub array in line 11. Modify this algorithm to keep track of start of Max sub array

Task 2

- Dry run brute force $O(n^2)$ algorithm on following array and show all working. Show all values of MaxSum[i] array. MaxSum[i] array stores maximum sum out of all subarrays ending at index i.

i	1	2	3	4	5	6	7	8	9
A[i]	2	-4	3	4	-3	5	-5	6	-1

Task 3

- Dry run Kadane's algorithm on following array and show all working. Show all values of MaxSum[i] array.

i	1	2	3	4	5	6	7	8	9
A[i]	2	-4	3	4	-3	5	-5	6	-1

Task 4

- Can you write the dynamic programing solution of this problem that takes O(1) memory (without array of MaxSum) and O(n) time?
- If yes, write the pseudocode.