Venkata Krishna
Chaitanya

G01336659

Assignment - 6

The maximum - subarray problem :-

To Do :-

Finding a sequence of days over which the net change from the first day to the last is maximum. Find the non-empty, contiguous subarray of A whose values have the largest sum.

Input :- An array A[1,2,...n] of numbers

Output :- Indices 'i' & 'j' such that A[i,..j] has the greatest sum of any nonempty, contiguous subarray of 'A', along with the sum of the values in A[i,...j].

I have initially tried designing a pseudo code with the 'for' loops to iterate through the array and finding the sum and considering the suitable array. But, I had to use three loops (for-loops) for that which would be of $O(n^3)$ complexity.

Now, in this approach :-

-> I try to find if the length of the array is '0' or not
-> If not '0', I try to find the max_value in the array after initializing them.

Venkata [illegible]
Chaitanya

G01336659

Assignment - 6

The maximum - subarray problem :-

To Do :-

Finding a sequence of days over which the net change from the first day to the last is maximum. Find the non-empty, contiguous subarray of A whose values have the largest sum.

Input :- An array A[1,2,...n] of numbers

Output :- Indices 'i' & 'j' such that A[i,...j] has the greatest sum of any nonempty, contiguous subarray of 'A', along with the sum of the values in A[i,...j].

I have initially tried designing a pseudo code with the 'for' loops to iterate through the arrays and finding the sum and considering the suitable array. But, I had to use three loops (for-loops) for that which would be of $O(n^3)$ complexity.

Now, in this approach :-

-> I try to find if the length of the array is '0' or not
-> If not '0', I try to find the max_value in the array after initializing them.

n → length of Array

```
def MaxSubArray(Array, n):
    if (n == 0):
        return 0
    else
        max_value = inf
        maxinstance = 0
        ind = 1    start = ~~stop~~ = ind = 1
                          stop
        end = 1
        start = 1
    for value in range(len(Array)):
        maxinstance = A[value]
        if (max_value < maxinstance):
            New_max_value = max_value
                index = index
                end = stop

for i in [1,.....n]:
    for j in [i,...n]:
        for k in [i,..j]:
            A[val] = A[val] + A[val+1]
```

```
def MaxSubArray (Array, n):
    if (n == 0):
        return 0
    else:
        max_value = inf
        maxinstance = 0
        start = stop = ind = 1
    for value in range(0, len(Array)):
        maxinstance = A[value]
        if (max_value < maxinstance):
            max_value = max_value
            inde= ind
            stop = stop
        if (maxinstance < 0):
            Do maxinstance = maxinstance
                           until
                            maxinstance = 0.
            Update start =.
    return maxvalue, index, stop.
```

We are technically using only one 'for' loop.

So, it would be iterating through Array of 'n' values

So, it would be of O(n) complexity.