Question 7

Part A:

```
def Maximum(A, right):
    if right == 0:
        return A[0]
    else:
        max_of_other_elements = Maximum(A, right-1)
        if A[right] > max_of_other_elements:
            return A[right]
        else:
            return max_of_other_elements


A = [5, 13, 9, 10]
print(Maximum(A, len(A)-1))
```

Part B:

A = [17, 62, 49, 73, 26, 51]:

A [17, 62, 49, 73, 26, 51] right = 1 max = 17
A [17, 62, 49, 73, 26, 51] right = 2 max = 62
A [17, 62, 49, 73, 26, 51] right = 3 max = 62
A [17, 62, 49, 73, 26, 51] right = 4 max = 73
A [17, 62, 49, 73, 26, 51] right = 5 max = 73
So, the max element of array A[0] to A[5] = 73

Part C:

```
def Maximum(A, right):
    if right == 0: ---------------------------------------------------- constant
        return A[0] ---------------------------------------------------- constant
    else:
        max_of_other_elements = Maximum(A, right-1) ------ n <= will be called n times
        if A[right] > max_of_other_elements: ---------------------- constant
            return A[right] ------------------------------------------- constant
        else:
            return max_of_other_elements -------------------------- constant


A = [5, 13, 9, 10]
print(Maximum(A, len(A)-1))
```

As per line five we have a T(n) of T(n-1) for a recurrence relation

This function has a big-oh of n of O(n)