**Design and Analysis of Algorithms Assignment**

**James Mutemi**

**sct221-0092/2020**

**Q1.**

Implementation using python:

def extract\_sentence(sentence, start, end):

if start > end:

return ""

elif start == end:

return sentence[start]

else:

return sentence[start] + " " + extract\_sentence(sentence, start + 1, end)

The recurrence relation for the algorithm

T(n) = T(n-1) + c

Time complexity using tracing tree method

T(n) = T(n-1) + 1

= [T(n-2) + 1] + 1

= [T(n-3) + 1] + 1+ 1

= ...

= T(1) + (n-1)1

T(n) = O(n)

**Q2**.

Implementation using python:

def circular\_shift\_array(arr, k):

n = len(arr)

if n == 0:

return arr

k = k % n

result = [0] \* n

for i in range(k):

result[i] = arr[n - k + i]

for i in range(n - k):

result[k + i] = arr[i]

return result

The recurrence relation for this algorithm is:

T(n) = O(n)

Time complexity

T(n) = O(1) + O(n) + O(k) + O(n – k)

T(n) = O(1) + O(n) + O(1) + O(n)

T(n) = O(n)