Name : Venkatesh G D
Div : D

USN : 2GI19CS175
Date :

DAA OBA-1

1.)

(i) $x(n) = x(n-1)+5$, for $n>1$, $x(1)=0$

$\rightarrow$ $x(n) = x(n-1)+5$

$= [x(n-2)+5]+5 = x(n-2)+5\cdot 2$

$= [x(n-3)+5]+5\cdot 2 = x(n-3)+5\cdot 3$

$= \ldots$

$= x(n-i)+5\cdot i$

$= \ldots$

$= x(1)+5\cdot(n-1) = 5(n-1)$

$x(n) = \Theta(n)$

(ii) $x(n) = x(n/2)+n$ for $n>1$, $x(n)=1$

$\rightarrow$

$x(2^k) = x(2^{k-1}) + 2^k$

$= [x(2^{k-2}) + 2^{k-1}] + 2^k = x(2^{k-2}) + 2^{k-1} + 2^k$

$= [x(2^{k-3}) + (2^{k-2})] + 2^{k-1} + 2^k = x(2^{k-3}) + 2^{k-2} + 2^{k-1} + 2^k$

$= \ldots$

$= x(2^{k-i}) + 2^{k-i+1} + 2^{k-i+2} + \ldots + 2^k$

$= \ldots$

$= x(2^{k-k}) + 2^1 + 2^2 + \ldots + 2^k = 1 + 2 + 2^2 + 2^3 + \ldots + 2^k$

$= 2^{k+1} - 1 = 2\cdot 2^k - 1$

$= 2n - 1$

$x(n) = \Theta(n)$

2.) "Q U I C K S O R T"

Applying selection sort algorithm.

Writing the ASCII values we get

Q	U	I	C	K	S	O	R	T
81	85	73	67	75	83	79	82	84

The algorithm for selection sort is

```
for i ← 0 to n-2 do
    min ← i
    for j ← i+1 to n-1 do
        if A[j] < A[min]
            min ← j
    swap A[i] & A[min]
```

By tracing selection sort algorithm, we get

Q	U	I	C	K	S	O	R	T
(81) |	85	73	(67)	75	83	79	82	84
67 |	(85)	(73)	81	75	83	79	82	84
67	73 |	(85)	81	(75)	83	79	82	84
67	73	75 |	(81)	85	83	(79)	82	84
67	73	75	79 |	(85)	83	(81)	82	84
67	73	75	79	81 |	(83)	85	82	84
67	73	75	79	81	82 |	(85)	(83)	84
67	73	75	79	81	82	83 |	(85)	(84)
67	73	75	79	81	82	83	84 |	85
67	73	75	79	81	82	83	84	85

After sorting we get: C I K O Q R S T U

2.) Performance:

* Worst case performance : $O(n^2)$
  If we want to sort in ascending order & array is in descending order, worst case occurs
* Best case performance : $O(n^2)$
  It occurs when array is already sorted
* Average case performance : $\Theta(n^2)$
  It occurs when elements of array are in jumbled order.

3.)

*) Maximum Element in array

MaxElement (A[0 ... n-1])
Input: An array A[0...n-1] of real numbers
Output: The value of largest element in A
Algorithm:

```
max ← A[0]
for i ← 1 to n-1 do
    if A[i] > max
        max ← A[i]
return max
```

*) Minimum element in array

MinElement (A[0 ... n-1])
Input: An array A[0 ... n-1] of real numbers
Output: The value of smallest element in A
Algorithm:

```
min ← A[0]
```

3.)

```
for i ← 1 to n-1 do
    if A[i] < min
        min ← A[i]
return min
```

* General plan for non-recursive algorithm to find order of growth of an element

→ Max Element

Input size : n

Basic operation : Comparison in for loop

if A[i] > max

$C(n) = 1+1+\dots n-1$

$= \sum_{i=1}^{n} 1 \quad = (n-1)-1+1$

$= n-1$

$= \in \Theta(n)$

→ Min Element

Input size : n

Basic Operation : comparison in for loop

if A[i] < min

$C(n) = 1+1+\dots n-1$

$= \sum_{i=1}^{n-1} \quad = (n-1)-1+1$

$= (n-1)$

$\in \Theta(n)$