Date :

DAA Assignment

M. Bilal Gondal

FA20-BCS-048

Question 1 :

a)

This algorithm has loop that runs from $i=1$ to $i=n-1$ so;

$\because$ base-condition $= 1$ to $n-1$

$T(n) = O(n)$ tim

Inside this loop :

MinMax () Functions that, also executes in $O(n)$

time so; $T(n) = O(n) \cdot (n)$

$= O\ n^2$

b) $T(n)$

As base condition are :

minval = A[0]

maxval = A[0]

inner loop runs from $i = 1$ to $i = n-1$ so;

$T(n) = O(n)$

Question 2 :

(a)

$T(n) = T(n-1) + T(n)$ -> general case $T(0) = 0$

$T(1) = 1$ -> base case

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

$T(2) = T(1) + 1 = 2$

$T(3) = T(2) + 2 = 3$

$T(4) = T(3) + 2 = 4$

Date :

$T(n) = 1+2+3 \ldots + (n-1) + n$

$T(n) = \Theta(n \cdot n$

$T_n = O(n)^2$

b) result ← 0

while b > 0

result ← result + a

b ← b - 1

End while

return result

(c) $T(b) = b$

$T(b) = O(b)$

(d) Recursive algo

$T(n) = O(n^2)$

$T(n) = O(b)$

(e) Algo - recursive Multiplication (a, b)

if b = 0 then

return 0

else then:

return a + recursive Multiplication (a, b-1)

$T(b) = O(b)$

Question: 3

$T(n) = T(n-1) + T(n) \Rightarrow$ general case : $T(n) = n$

$T(1) = 1$

$T(2) = T(1) + 1 = 2$ $T(4) = T(3) + 1 = 4$

$T(3) = T(2) + 1 = 3$

CamScanner

$$T(n) = T(n-1) + 1 \quad \because n > 1$$
$$T(2) = T(1) + 1 = 2.$$
$$T(3) = T(2) + 1 = 3$$
$$T(4) = T(3) + 1 = 4$$
$$\vdots$$
$$T(n) = 1 + 2 + 3 + \cdots + (n-1) + n$$

So;

$$T(n) = O(n) \cdot n$$
$$T(n) = O(n^2) \Rightarrow \text{recursive algorithm}$$

(b)

iterative version of algorithm:

```
Algorithm iterative-multiplication (a, b)
    result ← 0
while b > 0
        result ← result + a
        b ← b - 1
End while
    return result.
```

(c) the iterative algo consist of basic operation which is executed once for each iteration of loop so:

$$T(b) = b$$
$$T(b) = O(b)$$

$T(2) = T(1) + 1 = 2$
$T(3) = T(2) + 1 = 3.$
$T(4) = T(3) + 1 = 4$

$\vdots$

$T(n) = 1 + 2 + 3 + \cdots + n-1$

So

$$T(n) = On * O(1) = O(n) \cdot 1 = O(n).$$

the number of recursive calls also

$$T(n) = O(n)$$

So

$$\text{Overall } T(n) = O(n^2)$$

Question # 4:-

(a) Binary Insertion Sort (A, n).

• given algo contains loop that runs $i = 1$ to $i = n-1$ so.

$$T(n) = O(n).$$

* inside this loop, there is a call of binary search ( ) which

$$T(n) = O(\log_2(n))$$

This is are another loop that runs from $j = i-1$ to j eins which

$$T(n) = O(1).$$

Therefore overall:-

$$T(n) = O(n) \cdot O(\log_2(n)) * O_1 = O(n\log)$$

Since i is a function of n

$i = n-1$ so;

$T(n) = O(n^2 \log_2 n)$

let $n = 2^k$

$\log_2 n = \log_2 2^k$.

Xing by n^2 on b/S

$n^2 \log_2 n = n^2 \log_2 2^k$

$n^2 \log_2 n = (2^k)^2 \log_2 2^k$

$n^2 \log_2 n = 2^{k^2} \log_2 2^k$

$\because \log_2 2 = 1 \quad = k(1) = k$.

for $n^2 = k\, 2^{k^2}$

As $k(2^{k^2}) > 2^{k^2} = n^2$

$\therefore n^2 (\log n) > n^2$.

So

$n^2 \log n = n^2$ hold true

Question # 5

basic operation : $i = n$ $\quad i < n$.

$T(n) = n$.

$T(1) = 1$ base - case

$T(n) = T(n-1) + T(n) \Rightarrow$ general case

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

$T(2) = T(1) + 1 = 1 + 1 = 2$

$T(3) = T(2) + 1 = 3$

$T(n) = 1 + 2 + 3 + \ldots\ldots + (n-1) + n$

$T(n) = O(n^2)$

(d) Recursive algo has:

$$T(n) = O(n^2)$$

iterative algo has

$$T(n) = O(b)$$

iterative algo. is more effecient or faster, since "b" is typically much smaller than n

(e) Yes, it is possible to have recursive algo based on above definition

```
Algo_recursive Multiplication (a,b)
    if b=0 then
        return 0,
    else then;
        return a + recursive Multiplication
                (a ; b-1).
```

This algo runs until b=0 & reduces value of b each time so; $T(b) = O(b)$, final result is sum of all added values of a which is equal to $a*b$ according to the given definition.

Question #3:

$i+1$

basic operation:

$i=n$ $\therefore i<n$

basic operation :-

$$T(n) = T(n-1) + T(b) \Rightarrow \text{general case}$$

$$T(1) = 1 \Rightarrow \text{base-case} \quad \therefore T(n) = n$$

Question # 6 :-

(a) first algo has a recursive structure :

base - Case : $K = 0$; $K = n$
basic condition : rethe recursive call

=) addition of 2 algo.

As $n > 0$ or $n = 0$.

$T(n) = O(2^n)$ for $n > 0$.

or

$T(n) = O(1)$ for $n = 0$.

Overall time Complexity

$T(n) = O(2^n - 1) = O(2^n)$.

(b) 2nd algo has 2 nested loop :

- outer loop $i = 0$ to $i = x$
- inner loop $j = 0$ to $j =$

basic operation :

$j = 0$ || $j = 1$

$C[i][j]$ computes for general case,

inner loop iteration based on in (i, j) function.
i must be from 0 to n ; K is fund
no. of inner loop iteration based on n so

$T(n) = O(n^2) = O(n^2)$