DAA

Assignment – 1 – Answers

Q1. Write pseudocode of the following:

i. Factorial using loop

Ans:

$FACTORIAL(n)$
$result \leftarrow 1$
$For\ i \leftarrow 1\ to\ n$
do
    $result \leftarrow result \times i$
$end\ for$
$PRINT\ result$

ii. Factorial using recursion

Ans:

$FACTORIAL(n)$
$if\ n = 0\ then$
do
    $return\ 1;$
$else$
    $return\ n \times FACTORIAL(n-1);$
$end\ if$

iii. Pythagorean Theorem $c = \sqrt{a^2 + b^2}$

Ans:

$$PYTHAGOREAN\_THEOREM(a, b)$$
$$c \leftarrow sqrt\ (a\hat{}2 + b\hat{}2)$$
$$PRINT\ c;$$

iv. Fibonacci sequence

Ans:

$$FIBONACCI(n)$$
$$if\ n \leq 1\ then$$
$$do$$
$$\quad return\ n;$$
$$else$$
$$\quad return\ FIBONACCI(n-1)\ +\ FIBONACCI(n-2);$$
$$end\ if$$

Q2. Calculate the worst-case in terms of Theta (running time and memory access) on the pseudocodes given in question 1.

i. Factorial using for loop

$$FACTORIAL(n)$$
$$result \leftarrow 1$$
$$For\ i \leftarrow 1\ to\ n \qquad \boldsymbol{n\ times}$$
$$do$$
$$\quad result \leftarrow result \times i$$
$$end\ for$$
$$PRINT\ result$$

The for loop runs “n times” and each run involves a constant amount of work.

Hence time complexity is $\boldsymbol{\theta(n)}$

ii. Factorial using recursion

Ans:

$FACTORIAL(n)$
$if\ n = 0\ then$
do
$\quad return\ 1;$
$else$
$\quad return\ n \times FACTORIAL(n-1);$ $\quad$ $\boldsymbol{n\ times}$
$end\ if$

This recursive function, calls itself 'n' times, each time reducing 'n' by 1 until 'n' reaches zero.

Therefore, its time complexity is $\boldsymbol{\theta(n)}$

iii. Pythagorean Theorem $c = \sqrt{a^2 + b^2}$

Ans:

$PYTHAGOREAN\_THEOREM(a, b)$
$c \leftarrow sqrt\ (a\hat{}2 + b\hat{}2)$
$PRINT\ c;$

The above function does not involve any loops or recursion. It’s a simple mathematical formula.

Therefore, its time complexity is $\boldsymbol{\theta(1).}$

iv. Fibonacci sequence

Ans:

$FIBONACCI(n)$

$if\ n \leq 1\ then$

do

$\quad return\ n;$

$else$

$\quad return\ FIBONACCI(n-1)\ +\ FIBONACCI(n-2);$ $\boldsymbol{recursion\ 2\ dimentional}$

$end\ if$

The recursive Fibonacci algorithm you provided has an exponential time complexity, specifically $\boldsymbol{\theta(2\char`^n)}$. This is because for each call to FIBONACCI(n), it makes two additional recursive calls FIBONACCI(n-1) and FIBONACCI(n-2).

Q3 (a)

Follow the same steps which he did in the class for Min and Max algorithm (First Class Activity).

Q3 (b) Calculate the worst-case time (running time and memory access) of question 3.a.

```
    MAXIMA(int n, Point P[1 … n])
1.  for i ← 1 to n                                                      n times
2.  do maximal ← true
3.      for j ← 1 to n                                                  n times
4.      do
5.      if (i ≠ j) and (P[i].x ≤ P[j].x) and (P[i].y ≤ P[j].y)          4 access
6.          then maximal ← false
7.          break
8.      end if
9.      end for
10. if maximal
11. then output P[i].x, P[i].y                                          2 access
12. end if
13. end for
```

In terms of $T(n)$

1. Calculate inner loop i.e. on line no. 3, Lets say $T(I)$

$$I = \sum_{j=1}^{n} 4 = 4\sum_{j=1}^{n} j \ = 4n$$

For outer for loop lets say $T(M)$

$$M = \sum_{i=1}^{n} (2 + T(I)) = \sum_{i=1}^{n} (2 + 4n) = 2\sum_{i=1}^{n} + 4n\sum_{i=1}^{n} = 2\sum_{i=1}^{n} + 4n^2$$

$$= 2n + \ 4n^2 => \ 4n^2 + 2n$$

In terms of n Hence the largest n would be the worst-time case i.e.

$$\boldsymbol{\theta(n^2)}$$