Computer
ECE 001

Benha University

Computer Systems Engineering
Electrical Engineering Department

Faculty of Engineering
(at Shoubra)

Sheet 2 - Sol

I

- **5.22**
  The produced list of numbers: `1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89`

```
F0 ← 0;
F1 ← 1;            Initialization       Test
while (F1 < 100) do {
    print F1;
    Temp ← F0;                          Loop Body
    F0 ← F1;
    F1 ← F0 + Temp;                     Modification
}
```

- **5.25**

Searching for the value J: H, L, J	Searching for the value Z: H, L, N, O
A, B, C, D, E, F, G, **H**, I, J, K, L, M, N, O	A, B, C, D, E, F, G, **H**, I, J, K, L, M, N, O
A, B, C, D, E, F, G, H, I, J, K, **L**, M, N, O	A, B, C, D, E, F, G, H, I, J, K, **L**, M, N, O
A, B, C, D, E, F, G, H, I, **J**, K, L, M, N, O	A, B, C, D, E, F, G, H, I, J, K, L, M, **N**, O
	A, B, C, D, E, F, G, H, I, J, K, L, M, N, **O**

- **5.53**
  No. The algorithm will not terminate when `X = 0`.

```
Product ← 0;
Count ← 0;
while (Count < X) do {
    Product ← Product + Y,;
    Count ← Count + 1;
}
```

- **5.54**
  No. The algorithm will not compute the correct answer when `X = Y`.

```
Difference ← X - Y;
if (Difference = 0)
then {
    print "X equals Y";
}
else {
    if (Difference > 0)
    then {
        print "X is bigger than Y";
    }
    else {
        print "Y is bigger than X";
    }
}
```

- **5.57**
  The *loop invariant*[1] is:

```
    J ≤ Y and Z = X - J
```

The *stop condition* is:

```
    J ≥ Y
```

Upon loop termination, the *loop invariant* will be combined with the *stop condition* to give:

```
    (J ≥ Y) and (J ≤ Y and Z = X - J)
    J = Y and Z = X - J
    Z = X - Y
```

[1] An invariant of a loop is an assertion (claim) that is true before (and after) each iteration of that loop

II Answer the following questions:

1.

a)	b)
$\Theta(n)$	$\Theta(\log n)$

2.

The precondition is:

`L` is arranged in ascending order

The *loop invariant* is:

`V` is greater than any item in `L` preceding `T`.

The *stop condition* is:

`V ≤ T` **`or`** `T` **`is last`**`(L)`

The *termination argument* is:

`L` contain only a finite number of entries and every loop iteration `T` advances to the next item; therefore, `T` will eventually be **`last`**`(L)`,which satisfies the *stop condition*.

3.

Yes, both of them calculate the factorial of a given number `N`.

a)	b)
Recursive definition of factorial: $n! = n \times (n-1)!, 1! = 1$	Iterative definition of factorial: $n! = 1 \times 2 \times 3 \times \cdots \times n$

4.

$\Theta(1)$	Calculating $(-1)^n$
$\Theta(\log n)$	Binary Search
$\Theta(n)$	Sequential Search
$\Theta(n \log n)$	Quick Sort
$\Theta(n^2)$	Insertion Sort