V. Trinath 192311115 CSA0671 Assignment-1

1. Find the efficiency and order of notation for recursive algorithm - factorial of given no.

General Plan:-

1. Integer n
2. multiplication.
3. n times.
4. $f(n) = f(n-1) * n.$
   $M(n) = M(n-1) + 1$
   To compute $f(n-1)$ one more multiplication $f(n-1)$ by 1

$0! = 1$

$M(0) = 0$ → Initial Condition

5) Solving.

Pseudo code

Algorithm: fact(n).

// Problem description: Computes fact(n).

// input : Any integer n.

// output : factorial of n.

```
if (n==0)
    return 1;
else
    return fact(n-1)* n;
```

Subtraction Methods

1. forward substitution
2. Backward substitution.

forward substitution :-

$m(n) = m(n-1) + 1 \rightarrow (1)$

$m(0) = 0$

$n = 1$

$m(1) = m(1-1) + 1$

$m(1) = M(0)+1.$
$m(1) = 0+1.$
$m(1) = 1.$

$n = 2$

$m(2) = m(2-1)+1.$
$m(2) = m(1)+1$
$m(2) = 1+1$
$m(2) = 2$

$n = 3.$

$m(3) = m(3-1)+1.$
$m(3) = m(2)+1.$
$m(3) = 2+1.$
$m(3) = 3$

⋮

$n = i$
$M(i) = M(i-1)+1$
$m(n) = M(n-1)+1.$

Backward substitution.

$M(n) = M(n-1)+1 \rightarrow (1)$

$M(0) = 0.$

$M = n-1$

$M(n-1) = M(n-1-1)+1$
$M(n-1) = M(n-2)+1 \rightarrow (2).$

sub eq (2) in eq (1)
$M(n) = M(n-2)+1+1$
$M(n) = m(n-2)+2 \rightarrow (3).$

$M = n-2$

$M(n-2) = M(n-2-1)+1$
$M(n-2) = M(n-3)+1 \rightarrow (4).$

sub eq (4) in eq (3).

$M(n) = M(n-3-1)+1.$

$M(n) = M(n-4)+1 \rightarrow (5).$

$\vdots$

i^{th} recursive call

$M(n) = M(n-i)+i$

$n = i$

$M(i) = M(i-i)+i$

$M(i) = M(0)+i$

$M(i) = i$

Time Complexity $\rightarrow T(n) \in O(n).$

Find the efficiency and order of notation for the non-recursive Algorithm. find the maximum value in a list.

General plan

1. input.
2. Basic operation.
3. No. of times.
4. summation.
5. Solving summation.

Pseudo code.

```
Algorithm max-element (n[0,1,2,.....n-1])
// Problem description
// input: given array.
// output: maximum element in the array.
  max-value ← A[0]
  for i ← 1 to n-1 do
  {
  if (A[i] > max-value)
     max-value ← A[i]
  }
return max-value.
```

Iteration 1 :-

[5, 8, 4, 7, 9]

max value = 5

i = 1

if A[i] > 5

if 8 > 5 satisfies.

Iteration 2 :-

max-value = 8

i = 2

if A[2] > 8

if 4 > 8 not satisfies.

max-value = 8.

return 8

Iteration 3

max-value = 8

i = 3

if A[3] > 8

if 7 > 8 not satisfies.

max-value = 8

return 8

Iteration 4 :-

max-value = 8

i = 4

if A[4] > 8.

if 9 > 8 satisfies

max value = 9

return 9.

Time Complexity :- $C(n) = \sum_{i=1}^{n-1} 1$

formula :- $\sum_{i=k}^{n} 1 = n-k+1.$

$C(n) = (n-1) - \cancel{1} + \cancel{1}$

$= n-1.$

$T(n) \in O(n).$

3. Explain the steps to solve the tower of Hanoi Problem. And also Estimate the order of notation for n disk

General plan :-

1. n disk
2. Move
3. n times.
4. setup reccurance relation.
   - → Reccurance Equation.
   - → initial condition
5. solving.

Pseudo Code :-

Algorithm ToH (n, A, B, C)
//problem description Move disk from A to B
// input : Any integer n.
// output : tower of honai
if (n==1) then.
{
write ("Disk moved from A to B")
return
}
// move top (n-1) disk from A to B using C. TOH (n-1, A, B, C)
// move remaining Disk
TOH (n-1, B, C, A)
}

Recurrance relation :-
if $n > 1$

$m(n) = m(n-1) + 1 \; m(n-1) \Rightarrow 2m(n-1)+1.$

To move (n-1) disk from A to B

to move largest disk from A to C

to move disk from B to C.

$n = 1$

$m(1) = 1$

initial Condition.

forward substition :-

$m(n) = 2m(n-1)+1.$

$m(1) = 1$

$m(2) = 3$

$m(3) = 7$

$n = i$

$M(i) = 2M(n-i) + i$

$M(n) = 2M(n-1) + 1.$

Backward substitution :-

$M(n) = 2M(n-1) + 1 \rightarrow (1)$

$M(1) = 1$

$n = n-1$

$M(n-1) = 2M(n-1-1) + 1$

$M(n-1) = 2M(n-2) + 1 \rightarrow (2)$

sub (2) in (1).

$M(n) = 2\{2M(n-2) + 1\} + 1.$

$M(n) = 4M(n-2) + 2 + 1.$

$M(n) = 4M(n-2) + 3 \rightarrow (3)$

$n = n-2$

$M(n-2) = 2M(n-2-1) + 1.$

$M(n-2) = 2M(n-3) + 1 \rightarrow (4)$

Eq (4) in Eq (3).

$M(n) = 4[2M(n-3) + 1] + 3$

$M(n) = 8M(n-3) + 4 + 3$

$M(n) = 2^3 M(n-3) + 2^2 + 2 + 1$

$M(i) = 2^i M(n-3) + 2^{i-1} \ldots + 2 \ldots + 2 + 1.$

$$\boxed{x^{i-1} + x^{i-2} \ldots\ldots + 2 + 1 = \frac{1-x^i}{1-x}}$$

$M(n) = 2^i M(n-i) + \frac{1-2^i}{1-2}$

$\frac{1-2^i}{-1} = -(1-2^i) = 2^i - 1$

$M(n) = 2^i M(n-i) + 2^i - 1$

sub $i = n-1$

$M(n) = 2^{n-1} M(n-(n-1)) + 2^{n-1} - 1$

$= 2^{n-1} M(\cancel{n} - \cancel{n} + 1) + 2^{n-1} - 1$

$= 2^{n-1} M(1) + 2^{n-1} - 1$

$= 2^{n-1} + 2^{n-1} - 1$

$= 2 \cdot 2^{n-1} - 1$

$= \cancel{2} \cdot 2^n - \cancel{2^1} - 1$

$= 2^n - 1$