☆ Recursion in Programming

↓

function calling itself

↓

Merge Sort.

Recursion → great ideas used it is solved tons of problems.

Pseudo Code.

```
f(n)
  if n=0 or n=1  → base case → stops recursion
    return 1
  else
    return n*f(n-1) → recursion case.
```

1) $f(n) = \begin{cases} n * f(n-1) & \text{else} \\ 1 * \ n=0 \text{ or } 1. \end{cases}$ recursive

2) $f(n) = n * (n-1) * (n-2) * \ldots 1$ } iterative

Ex:- f(5)

↓

5 * f(4)

↓

4 * f(3)

↓

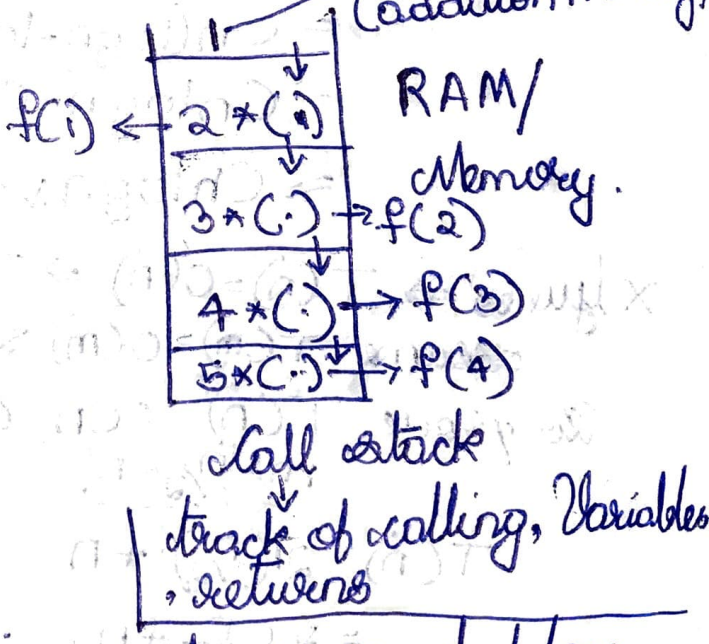
3 * f(2)

↓

2 * f(1)

↓

1 → stop recursion and comes out of loop.



$T(n) = C + T(n-1) \rightarrow O(n)$

Time Complexity

Space Complexity → O(n) ∵ because of call-stack

✱ Recursion VS iteration

```
f(n)
  if n=0 or n=1
      return 1
  else
      return n*f(n-1)
```

$T(n) = C + T(n-1) = O(n)$
$S(n) = O(n)$
(call stack)

```
f(n)
  p=1
  for i to n.
      p = p*i
  return p.
```

$T(n) = O(n)$
$S(n) = O(1)$
(No call stack)

Major Different (V.V.Imp).

✱ Tail - Recursion / Tail - call Optimization

```
fTR(n, a)        → accumulator
  if n=0 or n=1
      return a.
  else.
      return fTR(n-1, n*a)
```

Ex:-

fTR(5, 1) ← n!
↓
fTR(4, 5) → f(TR)(3, 20).
↓
fTR(1, 120) ← fTR(2, 60)

```
f(n).
  if(n=0) or n=1
      return 1
  else.
      return n*f(n-1)
```

⇒ non-Tail ~~Rec~~ recursion
or
Head recursion

⇒ after the recursion returns, operation to be performed

⇒ after the recursion returns, no operation need to be done

Tail Recursion:

Code → Compiler (C, C++, Java, Python)
↓
executable.

fTR → modern Compiler → iterative (space-complex) O(1)

Q) Consider the following recursive C function that takes two arguments.

```
unsigned int foo(unsigned int n, unsigned int r){
    if (n>0) return ((n%r) + foo(n/r, r));
    else return 0;
}
```

What is the return value of the function foo. when it is called as foo(345,10)?

a) 345
b) 12.
c) 5
d) 3.

foo(345, 10).
12 ← 5 + foo(34, 10).
5 ← 4 + foo(3, 10).
4 ← 3 + foo(0, 10).
3 ← 0

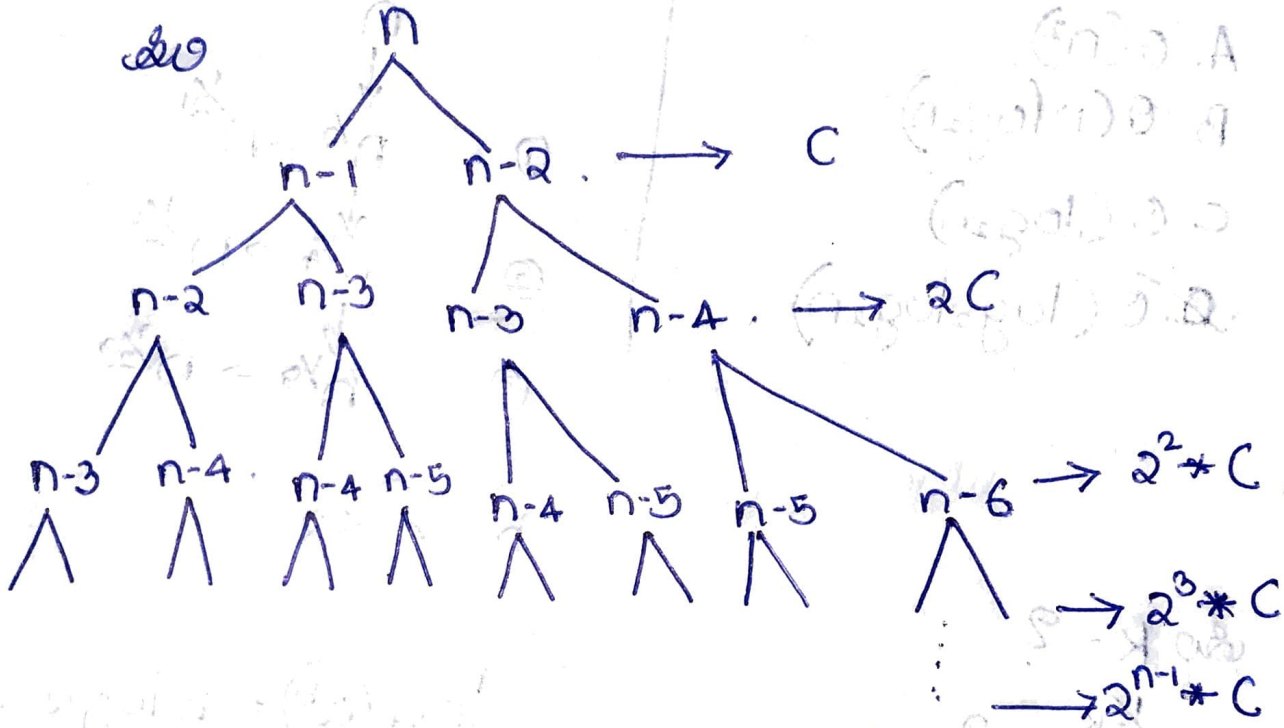
(2) fib(n)

```
if n=0
    return 0
if n=1
    return 1
else.
    return fib(n-1) + fib(n-2).
```

$$fib(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ fib(n-1) + fib(n-2) \end{cases}$$

What is the time complexity?

$$T(n) = T(n-1) + T(n-2) + C$$



$$T(n) \leq C + 2C + 2^2C + 2^3C + \ldots 2^{n-1}C.$$

$$T(n) \leq C(1 + 2 + 2^2 + 2^3 + \ldots 2^{n-1})$$

$$\leq C\{2^n - 1\}$$

$$T(n) \leq 2^n.$$

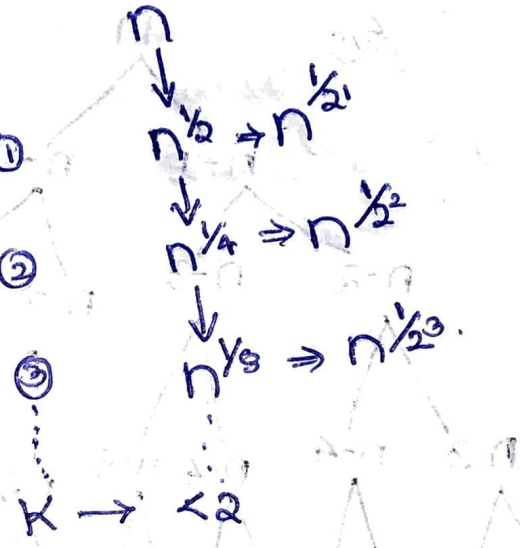
$$\underline{T(n) = O(2^n)}$$

② What is the time complexity of the following recursive function?

```
int Dosomething (int n) {
    if (n<=2)
        return 1;
    else
        return (Dosomething (floor(sqrt(n))) + n)
}
```

A. $\theta(n^2)$.
B. $\theta(n \log_2 n)$
C. $\theta(\log_2 n)$
D. $\theta(\log_2 \log_2 n)$

$T(n) = T(\sqrt{n}) + C$.



$\log(a^b) = b \log a$.

$\therefore K = ?$

$n^{1/2^K} = 2$.

$\log_2 n^{1/2^K} = \log_2 2$.

$\Rightarrow \frac{1}{2^K} \log_2 n = 1$

$\Rightarrow \log\left(\frac{1}{2^K}\right) + \log(\log_2 n) = 0$

$\Rightarrow \log 2^{-K} + \log(\log_2 n) = 0$.

$\Rightarrow -K = -\log(\log_2 n)$

$\Rightarrow K = \log_2(\log_2 n)$

$\therefore \log_2(\log_2 n)$

(2) Consider the following C program fragment in which i, j and n are integer variables.

```
for(i=n, j=0; i>0; i/=2, j+=i);
```

Let val(j) denote the value stored in the variable j after termination of the for loop. Which one of the following is true?

$i=n$	$i=n/2$	$i=n/4$
$j=0$	$j=0+n/2$	$j=\frac{n}{2}+\frac{n}{4}$ .....

$i=1$

$j=0+\frac{n}{2}+\frac{n}{4}+\frac{n}{8}+\ldots+1$

A) $val(j)=\theta(\log n)$

B) $val(j)=\theta(\sqrt{n})$

✓ C) $Val(j)=\theta(n)$

D) $Val(j)=\theta(n\log n)$

$$Val(j)=n\left[\frac{1}{2}+\frac{1}{4}+\frac{1}{8}+\ldots\frac{1}{n}\right]$$

G.S $\Rightarrow 1+r+r^2+r^3+\ldots\infty=\frac{1}{1-r}$. $|r|<1$

so $n\left[1/2+1/4+\frac{1}{8}+\ldots.\right]\Rightarrow$ finite number.

★ Shortcut.

$$T(n)=aT(n-b)+O(n^k)$$

$a>0;\ b\geq 1,\ k\geq 0$

Case 1:- if $a>1$ then $T(n)=O(n^k a^{n/b})$.

Case 2:- if $a=1$ then $T(n)=O(n^{k+1})$.

Case 3:- if $a<1$ then $T(n)=O(n^k)$.

Q). The time complexity of the following C function is (assume $n>0$)

```
int recursive (int n){
    if (n==1)
        return (1);
    else
        return (recursive(n-1)+ recursive(n-1));
}
```

A) $O(n)$
B) $O(n\log n)$
C) $O(n^2)$
D) $O(2^n)$.

$T(n) = T(n-1) + T(n-1) + C.$
$\Rightarrow 2T(n-1) + C.$
$\Rightarrow T(n) = aT(n-b) + n^k C.$
$a=2 \quad b=1 \quad k=0.$

$a>2:$
$\theta(n^k a^{n/b})$
$\Rightarrow \theta(n^0 2^{n/1}) \Rightarrow O(2^n).$

Q) $T(n) = T(n/3) + \theta(n).$
$a=1, \; b=3, \; k=1, \; p=0.$
$a \; b^k$
$1 < 3^1$
$p \geq 0 \rightarrow T(n) = \theta(n^k \log^p n).$
$= \theta(n^1).$
$\Rightarrow \underline{\theta(n)}.$

Q) $T(n) = \sqrt{2}\, T(n/2) + \sqrt{n}$.

$a = \sqrt{2}$, $b = 2$, $k = 1/2$, $p = 0$.

$a \quad b^k \qquad p > -1$

$\sqrt{2} = \sqrt{2}$.

$\Rightarrow \Theta(n^{\log_b a} \log^{p+1} n)$.

$\rightarrow \Theta(n^{1/2} \log n)$

$\Rightarrow \Theta(\sqrt{n} \log n)$.

Q) $T(n) = 2T(n/2) + n$. Which of these options is false.

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

b) $O(n \log n)$.

c) $\Theta(n^2)$.

d) $\Omega(n^2)$.

$a = 2 \quad b = 2 \quad k = 1 \quad p = 0$.

$a \quad b$.

$2 = 2$.

$p > -1 \Rightarrow \Theta(n^{\log_b a} \log^{p+1} n)$

$\Rightarrow \Theta(n \log n)$ or

$O(n \log n)$ or

$\Omega(n \log n)$

So w.k.t.

$n^2 > n \log n$.

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

$T(n) \neq \Omega(n^2)$.

Q) $T(1) = 1$

$T(n+1) = T(n) + \lfloor \sqrt{n+1} \rfloor \quad \forall\, n \geq 1$

The value of $T(m^2)$ for $m \geq 1$ is

a) $m/6\,(6m - 39) + 4$.

b) $m/6\,(4m^2 - 3m + 5)$

c) $m/2\,(3m^{2.5} - 11m + 20) - 5$.

d) $m/6\,(5m^3 - 34m^2 + 137m - 104) + 56$.

$n+1$
↓
$n \rightarrow \lfloor\sqrt{n+1}\rfloor$
↓
$n-1 \rightarrow \lfloor\sqrt{n}\rfloor$
↓
$n-2 \rightarrow \lfloor\sqrt{n-1}\rfloor$
⋮
$1 \rightarrow \lfloor\sqrt{2}\rfloor$
↓
$T(1) = 1$

$T(n+1) = \lfloor\sqrt{n+1}\rfloor + \lfloor\sqrt{n}\rfloor + \lfloor\sqrt{n-1}\rfloor + \lfloor\sqrt{n-2}\rfloor + \cdots + \lfloor\sqrt{2}\rfloor + 1$

$T(m^2) = \lfloor\sqrt{m^2}\rfloor + \lfloor\sqrt{m^2-1}\rfloor + \cdots + \lfloor\sqrt{2}\rfloor + 1$

1) $T(1) = 1$ $\quad m=1;\ m^2=1$

a) 1 $\quad$ b) 1 $\quad$ c) 5 $\quad$ d) 5/3.

2) $m=2;\ m^2=4$

$T(m^2) = \lfloor\sqrt{4}\rfloor + \lfloor\sqrt{3}\rfloor + \lfloor\sqrt{2}\rfloor + 1$

$\Rightarrow 2 + 1+1+1 \Rightarrow 5.$

a) 5 $\quad$ b) 5.

3) $m=3 \quad m^2=9$

$T(m^2) = \lfloor\sqrt{9}\rfloor + \lfloor\sqrt{8}\rfloor + \lfloor\sqrt{7}\rfloor + \lfloor\sqrt{6}\rfloor + \lfloor\sqrt{5}\rfloor + \lfloor\sqrt{4}\rfloor + \lfloor\sqrt{3}\rfloor + \lfloor\sqrt{2}\rfloor + 1$

$\Rightarrow 3 + (2+2+2+2+2) + 1+1+1$

$\Rightarrow 16.$

4) $m=4 \quad m^2=16.$

a) 16 $\quad$ b) 16.

$T(m^2) = \lfloor\sqrt{16}\rfloor + \lfloor\sqrt{15}\rfloor + \lfloor\sqrt{14}\rfloor + \lfloor\sqrt{13}\rfloor + \lfloor\sqrt{12}\rfloor + \lfloor\sqrt{11}\rfloor + \lfloor\sqrt{10}\rfloor + \lfloor\sqrt{9}\rfloor + \lfloor\sqrt{8}\rfloor + \lfloor\sqrt{7}\rfloor + \lfloor\sqrt{6}\rfloor + \lfloor\sqrt{5}\rfloor + \lfloor\sqrt{4}\rfloor + \lfloor\sqrt{3}\rfloor + \lfloor\sqrt{2}\rfloor + 1 \Rightarrow \underline{38}.$

a) 34.

b) 38.