Recursion- factorial of a number using Recursion.

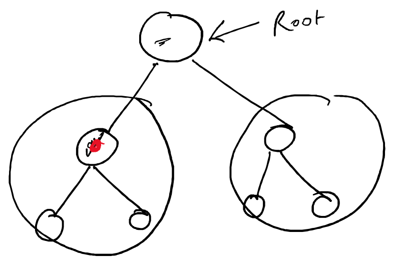
fac(n) = n x fac(n-1).

Recursion Code

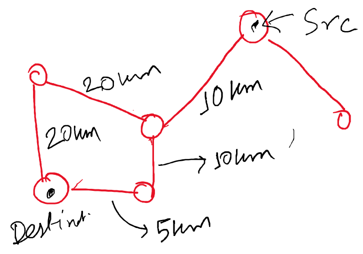
```
int fact(int n){
        if (n==0) return 1;
        return n x fac(n-1);
}
```

$$\begin{tikzcd}
5! = 5 \times 4! \arrow[dr] & & & & \\
& 4\times3! \arrow[dr] & & & \\
& & 3\times2! \arrow[dr] & & \\
& & & 2\times1! \arrow[dr] & \\
& & & & 1\times0!
\end{tikzcd}$$

BST



left-nodes < Root
right-nodes > Root



Programming —

Variables —

Data Types — → Primitive → char, boolean, int, float
→ User-defined → structure, class

Data Structure — → Linear: □□□□□□□□□□□ → stack, queue, LL → □→□→□→ null, array □□□□□ 0 1 2 3 4
→ Non-linear: → Trees, Graph

Design && Analysis of Algorithm —

Algorithm — Finite set of unambiguous instructions to complete a task.

```
while (1){
    System.out.println("Hello");
}
```

True, print("Hello");
True, print(Hello)

```
while (1){
    print ("Hello");
    break;
}
```

True, print(Hello) > 2

```
True,    print(
True,    print(Hello)
   .
   .
   .
```

$\left.\begin{array}{ll}\text{True,} & \text{print(Hello)} \\ & \text{break}\end{array}\right\rangle 2$

Analysis of algorithm — (1) Check if task is performed correctly or not.

(2) Efficiency — Time & Space