Recursion

Definition:

Recursion is a programming technique where a function calls itself directly or indirectly to solve a problem by breaking it down into smaller subproblems.

Key Points:

- **Base Case:** A condition to stop the recursion and prevent infinite calls.
- **Recursive Case:** The part where the function calls itself with a smaller or simpler input.
- Used to simplify complex problems by dividing them into smaller, identical problems.
- Each recursive call creates a new function call frame on the call stack.

Common Uses:

- Calculating factorials
- Generating Fibonacci numbers
- Tree traversal (preorder, inorder, postorder)

- Graph traversal (DFS)
- Solving puzzles like Tower of Hanoi
- Divide and conquer algorithms (Merge Sort, Quick Sort)

Example – Factorial:

cpp

CopyEdit

```
int factorial(int n) {
    if (n == 0)  // Base case
        return 1;
    else
        return n * factorial(n - 1);  // Recursive call
}
```

Advantages:

- Simplifies code for problems that naturally fit recursive solutions
- Easy to implement divide and conquer algorithms

Disadvantages:

- Uses more memory due to call stack overhead
- Risk of stack overflow if base case is missing or recursion is too deep
- Sometimes less efficient than iterative solutions
-