Recursion & Backtracking - Level 1

Page No.:
Date : \ \

* 1. Introduction to Recursion - How Recursion Works | Print Decreasing using Recursion *

- Recursion is a function calling itself to solve ~~similar~~ smaller versions of a problem.

- Every recursion function has -
  1] Base Case - Stopping condition (to avoid infinite calls)
  2] Recursive Case - The Fun^n calls itself with simpler input.

- Recursion → breaks → Big tasks → Smaller tasks.

- Python code for Decreasing using Recursion -

```
def print_decreasing(n):
    if n==0:
        return
    print(n)
    print_decreasing(n-1)  # Recursive call
```

* Working - (for n=5):

1] Print 5 → call print_decreasing(4)
⋮ ⋮ ⋮
5] Print 1 → call print_decreasing(0)
6] Base case reached → stop.

Recursion = Do a smaller part → let recursion handle the rest → stop at base case.

Teacher's Signature:.....

Page No.:
Date : \ \

* 2. Recursion Motion - Basics of Recursion | Print Increasing using Recursion *

* Recursion = Function calling itself to solve smaller subproblems.

- Base case stops recursion → prevents infinite calls.
- Recursive case continues recursion → reduces problem size.
- Fun^n follows LIFO.
- Printing before recursive call = work happens while going in (top-down).
- Printing after recursive call = work happens while coming back (bottom up).
- Python Example - Print increasing numbers (1→n)

```
def printIncreasing(n):
    if n == 0:
        return
    printIncreasing(n-1)
    print(n)
```

Print before = decreasing, print after = increasing.

Teacher's Signature:.....

Page No.:
Date : \ \

* 3. Basics of Recursion | Print Decreasing Increasing Numbers using Recursion *

Print ↓ First → Decreasing Sequence

Print ↑ after → Increasing Sequence

Ex. - (n=5):

Decreasing → 5 4 3 2 1

Increasing → 1 2 3 4 5

- Order control = Placement of print stmt in fun^n.

Tip - Base case stops, print placement decides order.

* 4. Recursion Basics | Print Decreasing Increasing Solutions Using Recursion *

Combining both → one print before recursion (decreasing), one print after recursion (Increasing)

Ex-

```
def printDI(n):
    if n == 0:
        return
# decreasing part
print(n, end = " ")
# Recursive call
printDI(n-1)
# increasing part
print(n, end = " ")
```

printDI(5)

Page No.:
Date : \ \

* 5. Factorial - Question | Recursion | Data Structures & Algorithms in JAVA * Python *

- Factorial defn - $n! = n \times (n-1) \times (n-2) \ldots \times 1$
  (Special case: $0! = 1$)

- Recursive factorial formula → $n! = n \times (n-1)!$

Ex:

```
def fact (n):
    if n == 0 or n == 1:
        return 1
    return n * fact (n-1)
```

* 6. Recursion Basics | Factorial of a no. using Recursion in Python *

- When base case is reached, recursion stops & values get returned back step by step.

Ex.

```
def Factorial (n):
    if n == 0:
    return 1
    return n * factorial (n-1)
```

Call factorial (n)
• n = 4, not 0, so → 4 * factorial(3)
Returning back:
Factorial (0) returns 1
1 = 1 * 1 = 1

Teacher's Signature:.....