

Learning Consolidation Implement Recursion





In this sprint, you have learnt:

- Explain Recursive Methods
- Explore the Base Case in Recursion
- Calculate Factorial using Recursion

Recursive Methods

- Recursion is the process a method goes through when one of the steps of the method involves invoking the method itself.
- A method that goes through recursion is said to be 'recursive'.
- Recursion is achieved in java programming by using methods called recursive methods that calls itself during its execution.
- The process may repeat several times, outputting the result at the end of each iteration until a **base case is reached**.

Base Case

- The recursive method can become entangled in an infinite loop, since it keeps calling itself repeatedly.
- In any recursive method there is a base case which is the termination condition for the recursion.
- Base case is a condition specified in the recursive method using a simple if statement, the method terminates once the condition is evaluated to true.
- A base case represents the end of recursion.

Recursively Calculating Factorial of a Number

- A factorial function is defined as:
 $n! = 1 \times 2 \times 3 \times 4 \times \dots \times n$
- For example, to calculate the factorial of 3 by using recursion,

- First define 3! in terms of 2!:• $3! = (3 \times 2!)$
- Then, define 2! in terms of 1!:• $3! = (3 \times (2 \times 1!))$
- Finally define 1! in terms of 0!: $0! = 1$ • $3! = (3 \times (2 \times (1 \times 0!)))$
- Therefore, the expression becomes:
 $3! = (3 \times (2 \times (1 \times 1)))$
 $3! = (3 \times (2 \times 1))$
 $3! = (3 \times 2)$
 $3! = 6$

Note that here until 0! is reached, or the result becomes 1, recursion happens. Hence number **$n == 1$** is the **base case**

Recursive Method for Finding the Factorial of a Number

```
int factorial(int num)
{
    if (num==1) {
        return 1;
    }
    return num * factorial(num-1);
}
```

Base Case

Method calling itself

