Recursion Assignment

- Objective:
  - This assignment aims to strengthen your understanding of recursion in Java. You will implement various recursive algorithms, analyze their performance, and write well-structured recursive functions.
- Instructions:
  - Implement the problems in Java.
  - Use only recursion (loops are not allowed).
  - Ensure your code is properly commented and follows good coding practices.
  - Submit a well-formatted report containing:
    - Problem statement
    - Code implementation
    - Explanation of your approach
    - Sample input/output

Problem Statements

- Part A: Basic Recursion
  - Factorial Calculation
    - Write a recursive function to calculate the factorial of a number.
  - Example:

```
Input: 5
Output: 120
```

  - Fibonacci Series (1 Mark)
    - Implement a recursive function to compute the n-th Fibonacci number.
  - Example:

```
Input: 6
Output: 8
```

  - Sum of Digits (1 Mark)
    - Write a recursive function to compute the sum of digits of a number.
  - Example:

```
Input: 987
Output: 24
```

  - Reverse a String
    - Implement a recursive function to reverse a string.
  - Example:

```
Input: "hello"
Output: "olleh"
```

  - Power Function
    - Implement a recursive function to compute x^n.
  - Example:

```
Input: (2, 5)
Output: 32
```

- Part B: Intermediate Recursion
  - GCD using Recursion
    - Find the Greatest Common Divisor (GCD) of two numbers using recursion.
  - Example:

```
Input: (48, 18)
Output: 6
```

  - Check Palindrome
    - Implement a recursive function to check if a string is a palindrome.
  - Example:

```
Input: "racecar"
Output: true
```

  - Binary Search
    - Implement Binary Search using recursion on a sorted array.
  - Example:

```
Input: ([1, 3, 5, 7, 9], 5)
Output: 2 (index)
```

  - Tower of Hanoi
    - Implement the Tower of Hanoi problem for n disks.
  - Example:

```
Input: n = 3
Output:
Move disk 1 from A to C
Move disk 2 from A to B
Move disk 1 from C to B
Move disk 3 from A to C
```

  - Generate All Subsequences of a String
    - Print all possible subsequences of a given string using recursion.
  - Example:

```
Input: "abc"
Output: {"", "a", "b", "c", "ab", "ac", "bc", "abc"}
```

- Part C: Advanced Recursion
  - String Permutations
    - Generate all permutations of a given string.
  - Example:

```
Input: "abc"
Output: {"abc", "acb", "bac", "bca", "cab", "cba"}
```

  - Combination Sum Problem
    - Given an array of numbers and a target sum, find all unique combinations that sum up to the target.
  - Example:

```
Input: [2, 3, 6, 7], target = 7
Output: [[7], [2, 2, 3]]
```

  - N-Queens Problem (2 Marks)
    - Place N queens on an N×N chessboard so that no two queens attack each other.
  - Example:

```
Input: N = 4
Output:
[0, 1, 0, 0]
[0, 0, 0, 1]
[1, 0, 0, 0]
[0, 0, 1, 0]
```

  - Rat in a Maze Problem
    - Given an N×N matrix, where 0 represents a wall and 1 represents a possible path, find a way from the top-left to the bottom-right.

Submission Guidelines

- Deadline: 27 March 2025 before 10 PM
- Submit at: Our Lab WhatsApp Group
- Submit a PDF document with problem name, its code, output and explanation
- Plagiarism is strictly prohibited.