Day 19:

Assignment 1: Tower of Hanoi Solver

**Create a program that solves the Tower of Hanoi puzzle for n disks.
The solution should use recursion to move disks between three
pegs (source, auxiliary, and destination) according to the game's rules.
The program should print out each move required to solve the puzzle.**

A)
Introduction:
Java program that solves the Tower of Hanoi puzzle using recursion. The program prints each move required to solve the puzzle.

Java code:

```
Package Day18;
public class TowerOfHanoi {

  // Function to print the move
  public static void moveDisk(int n, char fromPeg, char toPeg) {
    System.out.println("Move disk " + n + " from " + fromPeg + " to " + toPeg);
  }

  // Recursive function to solve Tower of Hanoi puzzle
  public static void solveTowerOfHanoi(int n, char sourcePeg, char destinationPeg, char auxiliaryPeg) {
    if (n == 1) {
      // Base case: only one disk to move
      moveDisk(n, sourcePeg, destinationPeg);
      return;
    }

    // Move n-1 disks from source to auxiliary, using destination as a temporary holding area
    solveTowerOfHanoi(n - 1, sourcePeg, auxiliaryPeg, destinationPeg);

    // Move the nth disk from source to destination
    moveDisk(n, sourcePeg, destinationPeg);

    // Move the n-1 disks from auxiliary to destination, using source as a temporary holding area
    solveTowerOfHanoi(n - 1, auxiliaryPeg, destinationPeg, sourcePeg);
  }

  public static void main(String[] args) {
    int numberOfDisks = 3; // Change this value to solve for a different number of disks
    solveTowerOfHanoi(numberOfDisks, 'A', 'C', 'B');
  }
}
```

Explanation:

1. moveDisk function*:This function prints the move of a disk from one peg to another.*

2. solveTowerOfHanoi function*: This is the recursive function that solves the Tower of Hanoi puzzle. It takes the number of disks n, the source peg, the destination peg, and the auxiliary peg as arguments.*

3. Base case*: When n is 1, the function simply moves the disk from the source peg to the destination peg.*

4. Recursive case:
- The function first moves n-1 disks from the source peg to the auxiliary peg using the destination peg as a temporary holding area.
- Then, it moves the nth disk from the source peg to the destination peg.
- Finally, it moves the n-1 disks from the auxiliary peg to the destination peg using the source peg as a temporary holding area.

*5***. main method***: The main method initializes the number of disks and calls the solveTowerOfHanoi method to start the process.*

You can change the value of numberOfDisks to solve the Tower of Hanoi puzzle for a different number of disks.