Control Flow - Introduction to For Loop

Generally we will do like this ….

```
public class ForStatement {

    public static void main(String[] args) {
        System.out.println("10,000 at 2% interest = " + calculateInterest(10000, 2));
        System.out.println("10,000 at 3% interest = " + calculateInterest(10000, 3));
        System.out.println("10,000 at 4% interest = " + calculateInterest(10000, 4));
        System.out.println("10,000 at 5% interest = " + calculateInterest(10000, 5));
    }

    public static double calculateInterest(double amount, double interestRate) {
        return amount * (interestRate / 100);
    }
}
```

Instead of calling the functions everytime we can use FOR loop

For Loop:

A for loop in programming is a control flow statement that allows code to be executed repeatedly for a set number of times or through a collection of elements (like arrays, lists, or ranges). It's one of the most commonly used loops, especially when you know in advance how many times the loop should run.

Basic Structure

In many programming languages, the for loop has this general syntax:

rust

Copy code

```
// General Syntax (in pseudocode)
```

```
for initialization; condition; increment/decrement {
   // Code to be executed repeatedly
}
```

Here’s a breakdown of each part:

1. **Initialization**: This step is executed only once at the beginning of the loop. It typically sets a loop control variable to an initial value. For example, int i = 0; initializes i to 0.
2. **Condition**: The loop continues to run as long as this condition is true. The condition is checked before each iteration of the loop. If the condition becomes false, the loop terminates.
3. **Increment/Decrement**: This step is executed after each iteration of the loop body. It updates the loop control variable, usually increasing or decreasing it by a specific amount (e.g., i++ to increment i by 1 after each loop cycle).
4. **Code block**: This is the part of the loop that gets executed repeatedly as long as the condition remains true.

Example in Java

```
for (int i = 0; i < 5; i++) {
   System.out.println("i is: " + i);
}
```

- **Initialization**: int i = 0; sets the loop control variable i to 0.
- **Condition**: i < 5 ensures that the loop will run as long as i is less than 5.
- **Increment**: i++ increases the value of i by 1 after each iteration.
- **Loop body**: System.out.println("i is: " + i); prints the current value of i.

The output will be:

```
i is: 0
i is: 1
i is: 2
i is: 3
i is: 4
```

How It Works

1. **Step 1 (Initialization)**: int i = 0 sets i to 0.
2. **Step 2 (Condition)**: i < 5 checks if i is less than 5. Since 0 is less than 5, the loop proceeds.
3. **Step 3 (Code execution)**: The loop body (System.out.println(...)) is executed, printing i is: 0.
4. **Step 4 (Increment)**: i++ increments i by 1, so now i is 1.

5. **Step 5 (Condition)**: The condition i < 5 is checked again. Since 1 is less than 5, the loop continues, printing i is: 1.
6. This process repeats until i equals 5. When i is 5, the condition i < 5 becomes false, and the loop ends.

So , now the above example which was discussed earlier can be written as:

```
public class ForStatement {

   public static void main(String[] args) {
      for (int i = 2; i <= 5; i++) {
         System.out.println("10,000 at " + i + "% interest = " + calculateInterest(10000, i));
      }
   }

   public static double calculateInterest(double amount, double interestRate) {
      return amount * (interestRate / 100);
   }
}
```