Loops in Java

Introduction

In a loop, a part of a program is repeated over and over until a specific goal is reached. Loops will continue to execute statements, called the body of the loop, as long as the condition holds true. In Java, there are three types of loops:

- **while loops**
- **for loops**
- **do loops**

### Important Note:
When you declare a variable inside the loop body, the variable is created for each iteration and removed after the end of each iteration.

#### Operators in Loops

<table>
<thead>
<tr>
<th>Operator</th>
<th>Usage</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>++</td>
<td>x++</td>
<td>x = x+1</td>
</tr>
<tr>
<td>--</td>
<td>x--</td>
<td>x = x-1</td>
</tr>
<tr>
<td>+=</td>
<td>x+=y</td>
<td>x = x+y</td>
</tr>
<tr>
<td>-=</td>
<td>x-=y</td>
<td>x = x-y</td>
</tr>
</tbody>
</table>

### Common Errors

- **Infinite loop**: This occurs when the condition is never met, and the loop will continue to run forever. This can occur if you forget to update the variable used in the condition or if you increment instead of decrement the variable (or vice versa).

- **Off-by-one error**: It is easy to be off by one iteration of the loop. This can be avoided by manually tracing the variables as the loop runs.

### Practice

1. What does the following loop print?
   ```java
   int n = 1;
   while (n < 100)
   {
     n = 2*n;
     System.out.print(n + " ");
   }
   ```

2. What type of error do you see in the following code?
   ```java
   int n = 1;
   while (n != 50)
   {
     System.out.println(n);
     n = n+10;
   }
   ```

### Syntax for while loops:

```java
while (condition) {
  statements
}
```

#### Generally, the variable for the condition is declared outside of the loop and updated in the loop.

### Syntax for for loops:

```java
for (initialization; condition; update) {
  statements
}
```

#### In for loops, the variable for the condition is initialized in the for statement.