Introduction to For Loops

With While Loops Review!

Simple Loops have an anatomy that does not vary. The structure is:

- Loop Variable Declaration: A variable is declared that will be used as part of the Loop Condition.
- Loop Variable Initialization: The Loop Variable is set to an initial value that makes the Loop Condition `true`.
- Loop Condition: This will be a Boolean Expression (see definition below!). The loop continues as long as the Boolean Expression evaluates as `true`.
- Loop Variable Modification: The Boolean Expression defined in the Loop Condition must at some point evaluate as `false` for the loop to end. This occurs when the value of the Loop Variable changes so that on the next iteration of the loop, the Loop's Conditional Boolean Expression evaluates as `false`.

> In the special form of loop called the "infinite loop" the condition never becomes false. Such a loop never ends! When this happens in a program it usually means there is a design flaw! There will never be such a loop intentionally used on the AP Test!

What is a Boolean Expression?

A Boolean Expression is a mathematical expression that evaluates as either `true` or `false`.

For example, the expression `5 < 6` is clearly `true` while the formula `5 > 6` is `false`.

In computer programming, a variable replaces some part of the formula, so:

```
int a = 5;
if (a < 6) {
      System.out.println("A is less than 6");
} else {
      System.out.println("A is greater than or equal to 6");
}
```

In the example above we would see "`A is less than 6`" in the console. But changing the first line to read: `int a = 6;` would result in "`A is greater than or equal to 6`" in the console. In this example, the Boolean Expression is `(a < 6)` because it can be said to be (or evaluate to) either `true` or `false`.

Another way to look at it is that when a = 5, the expression `(a < 6)` is `true` and when a = 6, the expression `(a < 6)` is `false`.

In a loop, like an "if" statement, there is a conditional expression. The loop will execute over and over until the conditional expression is `false`.

Each of the Loop Structures can be seen in the simple loop below:

```
01 int x; // Loop Variable Declaration
02 x = 0; // Loop Variable Initialization
03 while (x < 5) { // Loop Condition: Boolean Expression is (x < 5)
04       x = x + 1; // Loop Variable Modifier
05 }
```

The loop variable is x, which is initialized with the value 0. The loop then evaluates the conditional expression `(x < 5)` that is `true` because 0 is less than 5.

Each time the loop executes, it is called an “Iteration” of the loop. During the Iteration of this loop, the Loop Variable `x` is modified by adding 1 to its current value. When the first iteration of the loop is complete, the variable `x` has the value 1.

Before the contents of the loop are executed again, the conditional expression is evaluated with the current value of `x`. Since `x` now has the value 1, the expression `(x < 5)` is still `true` so the loop iterates again. The value of `x` increases by 1 again and again during each iteration of the loop until it eventually has the value 5.

When `x` has the value 5 the expression `(x < 5)` is `false` so the loop does not iterate again!

A special form of loop can be used to simplify this process into fewer lines of code. Such a loop is called a "For Loop." It has the same basic structure, but is written in a much more compressed form:

```
01 for (int x = 0; x < 5; x = x + 1) {
02
03 }
```

This “For Loop” is exactly the same as the previous example of the “While Loop.” Each of the structural elements is represented. The Loop Variable Declaration is `int x`, the Loop Variable Initialization is `x = 0`, the Loop Conditional Expression is `x < 5`, and the Loop Variable Modification is `x = x + 1`.

The number of times the loop iterates can be summarized by determining how many times the loop code will be executed. In the loop above, line 2 of the code (which currently does nothing) will be executed 5 times so this loop is said to iterate 5 times.

WHAT IS AN "ITERATION VARIABLE?" – SINCE THE LOOP VARIABLE IS USUALLY THE SOLE DETERMINING FACTOR IN HOW MANY TIMES A LOOP ITERATES, IT IS OFTEN REFERED TO AS THE "ITERATION VARIABLE."

Below are several examples of basic while and for loops that result in identical console output:

While Loop	For Loop	Console Output

`int y=5;`
`while (y<10) {`
`  System.out.println("y = "+y);`
`  y=y+1;`
`}`
	`for (int y=5; y<10; y=y+1) {`
	`  System.out.println("y = "+y);`
	`}`
		`y = 5`
		`y = 6`
		`y = 7`
		`y = 8`
		`y = 9`

`int y=5;`
`while (y>0) {`
`  System.out.println("y = "+y);`
`  y=y-1;`
`}`
	`for (int y=5; y>0; y=y-1) {`
	`  System.out.println("y = "+y);`
	`}`
		`y = 5`
		`y = 4`
		`y = 3`
		`y = 2`
		`y = 1`

`int y=1;`
`while (y<=10) {`
`  System.out.println("y = "+y);`
`  y+=2;`
`}`
	`for (int y=1; y<=10; y+=2) {`
	`  System.out.println("y = "+y);`
	`}`
		`y = 1`
		`y = 3`
		`y = 5`
		`y = 7`
		`y = 9`

`int y=1;`
`while (y<=5) {`
`  System.out.println("y = "+y);`
`  y++;`
`}`
	`for (int y=1; y<=5; y++) {`
	`  System.out.println("y = "+y);`
	`}`
		`y = 1`
		`y = 2`
		`y = 3`
		`y = 4`
		`y = 5`