Chapter 4: Loops

Loops and Motivating Examples

- A loop is used when an action is to be repeated multiple times
- Examples
  - Printing "Hello" 50 times
  - Prompting the user to enter a positive number (and repeating the prompt if the number entered is not valid)
  - Adding all of the numbers between 1 and 100

while loop

```
// do something (and repeat) while a condition is true
while (condition is true) {
        // statements to repeat go here
}
```

Note: there are *no* semi-colons after the condition

```
// Example: print Hello 50 times
int count = 1;
while (count <=50) {
        System.out.println("Hello #" + count);
        count++;
}
```

do..while loop

```
// do something at least once, then repeat while condition is true
do {
        // statements to repeat go here
} while (condition is true);
```

Note: there is a semi-colon after the condition

```
// Example: prompt user to enter a positive #, ensuring valid input
do {
        System.out.println("Enter a positive number");
        num = scnr.nextInt();
} while (num <= 0);
```

for loops (typically for counting)

```
// start with initial expression
// repeat while conditionExpression is true, and
// execute the updateExpression when the end of the loop body is reached
for (initialExpression; conditionExpression; updateExpression) {
        // statements to repeat go here
}

// Print "hello" 50 times
for (int i = 1; i <= 50; i++) {
        System.out.println("Hello #" + count);
}
```

Additional looping concepts

- A *nested loop* is when you have one loop inside another. Every time the outer loop repeats, the inner loop starts from scratch and is executed the appropriate number of times
- A *break* statement exits the current loop
- A *continue* statement causes the program to jump to the loop condition check