Loops.

Loops implement repetitive computation, a.k.a. iteration.

Java loop statements:

1. Start with the `while`-loop. Syntax

   ```java
   while(condition)
   Statement;   //body of loop
   ```

   or

   ```java
   while(condition)
   {
   //body of loop
   First_Statement;
   ...
   Last_Statement;
   }
   ```

Several logical loop organizations

1. counting loops – know exactly how many times to repeat a set of actions
   - usually done with a use of a counter variable
   - counter is initialized before the loop starts executing
   - counter is updated after each iteration of the loop

Example: print numbers from 0 to m.

```java
public class SimpleCountingLoop {
    public static void main(String[] args) {
        int i = 0; // the counter variable - to keep track of
        // number of times gone through the loop
        int m = 5;
        while (i < m){
            System.out.println("i is " + i + " now.");
            i++;
        }
        System.out.println("Done. i is " + i);
    }
}
```

Question: What will happen if m = -5? If the i++; in the loop body is omitted?

Programming and Debugging Pitfalls: infinite loops, off by one
Practice problems: for all the problems, assume Scanner kb = new Scanner (System.in);
1. Write a code segment that allows the user to enter 10 numbers and computes and prints out their sum, average, product.

2. Write a code segment that asks the user how many numbers there are in a list, then allows the user to enter the numbers and computes and prints out the smallest and the largest number.

3. Write a code segment that prints out all characters of a string one character per line.
2. Condition controlled loops – repeat while a certain condition is true  
   - Sentinel-controlled loops – stop repeating when a certain sentinel value is encountered

**Examples:**
1. Let user input numbers until user enters 0.  
   0 is the sentinel in this case.

```java
Scanner kbrd = new Scanner (System.in);
aNum = kbrd.nextInt();
while ( aNum != 0 ) {
    System.out.print(aNum);
    aNum = kbrd.nextInt();
}
```

**Practice problems:**
1. Write a code segment that continues to read a number from its user, until an even number is entered.

2. Write a code segment that continues to read user input until the user enters a number between 0 and 9.
boolean flag controlled loops – stop when a boolean variable that reflects a certain state is false

Example: describe what happens, when the following code segment is executed.

```java
Scanner kb = new Scanner(System.in);
System.out.println("Please enter a number:");
int prevNum = kb.nextInt();
int currNum;
boolean nomatch = true; // boolean flag
while (nomatch) // same as nomatch == true
{
    System.out.println("Please enter next number:");
    currNum = kb.nextInt();
    if (currNum == prevNum)
        nomatch = false;
    prevNum = currNum;
}
```

Practice problem: How to make the above loop terminate in case no match occurred after 100 numbers were entered?
2. **do-while loop**: similar to while-loop, but condition is checked at the end of each iteration. The body of the loop is going to be executed at least once.

```
    do
      Statement;  //body of loop
    while(condition);

    do
      {
        //body of loop
        First_Statement;
        ...
        Last_Statement;
      } while(condition);
```

**Practice problem:**

1. Write a code segment that continues to read a number from the user, until the user enters an *even* number. Use a do-while loop.
3. for-loops – especially useful for counting loops. Syntax:

```java
for (init-statements; loop-condition; update-statements)
{
    //body of loop
    First_Statement;
    ...
    Last_Statement;
}
```

**Example:** count down from 9 to 0

```java
for( int count = 9; count >= 0; count-- )
{
    System.out.print("T = " + count);
    System.out.println(" and counting");
}
```

System.out.println("Blast off!");

**Practice problems.**

1. Write a code segment that prints out a reverse of the string entered by the user, e.g. given “Walter” it would print “retlaW”.

2. Write a code segment that reads in a string and a character from the user, and prints out the number of occurrences of the character inside that string.
4. Nested loops:

When one loop is placed within the body of another, the entire construct is called “nested loops”.

Example:

```java
for (int m = 1; m <= 10; m++)
{
    count = 1; // initialize the counter variable
    while (count <= m) {
        System.out.print(count);
        count = count + 1; // same as count++
    }
    System.out.println("\n ***");
}
```

Practice problems
1. Print out a triangular pattern based on value stored in variable `rows`. The pattern below shows what’s printed for value of rows equal to 7.

```
1
22
333
4444
55555
666666
7777777
```

2. What output is produced by the following code segment:

```java
int k, s;
for (int j = 1; j <= 10; j++) {
    if (j % 3 == 0) {
        for (k = j, s = 0; k>=0; k--){
            s += k;
        }
        System.out.println("j is " + j+ " k is " +k+ " s is " + s);
    }
}
```