Control flow: Repetition

- Sequence of statements that are actually executed in a program
- Conditional and Repetition statements: enable us to alter control flow
Example

- **Investment problem:** You put $10,000 into a bank account that earns 5% interest per year.

<table>
<thead>
<tr>
<th>year</th>
<th>interest</th>
<th>balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td>$10,000.00</td>
</tr>
<tr>
<td>1</td>
<td>$500.00</td>
<td>$10,500.00</td>
</tr>
<tr>
<td>2</td>
<td>$525.00</td>
<td>$11,025.00</td>
</tr>
<tr>
<td>3</td>
<td>$551.25</td>
<td>$11,576.25</td>
</tr>
<tr>
<td>4</td>
<td>$578.81</td>
<td>$12,155.06</td>
</tr>
</tbody>
</table>

- ... How many years does it take for the account balance to be double the original?

This example is adapted from Cay Horstmann’s *Big Java, Early Objects*, 5th edition.
The while Statement

```java
while (condition) {
    statement;
}
```

• If the condition is true, the statement is executed
• Then the condition is evaluated again, and if it is still true, the statement is executed again
• The statement is executed repeatedly until the condition becomes false

Example

• Print the even numbers from 2 to 100

Algorithm:

• initialize a counter to 2
• while counter <= 100
  – print counter
  – add 2 to counter

Java? → use while statement
Try this:

• Write a loop to print the powers of 2 less than or equal to 2048, i.e., 2, 4, 16, ..., 1024, 2048.
The \texttt{while} statement in action

\begin{verbatim}
int count = 1;
while (count <= 3) {
    System.out.println (count);
    count++;
}
\end{verbatim}

Output:
1
2
3

The \texttt{while} statement \textit{“unraveled”}

\begin{verbatim}
int count = 1;
TEST:(count <= 3) \Rightarrow true
{ System.out.println (count);
  count++;
}
TEST:(count <= 3) \Rightarrow true
{ System.out.println (count);
  count++;
}
TEST:(count <= 3) \Rightarrow true
{ System.out.println (count);
  count++;
}
TEST:(count <= 3) \Rightarrow false
\end{verbatim}

\textbf{EXIT LOOP}
Example

- Table of squares and cubes:

<table>
<thead>
<tr>
<th>N</th>
<th>N²</th>
<th>N³</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>64</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>125</td>
</tr>
</tbody>
</table>
Example

- **Table of powers:** Compute the powers of 2 and the powers of 3 and print a table like this:

<table>
<thead>
<tr>
<th>N</th>
<th>(2^N)</th>
<th>(3^N)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>81</td>
</tr>
</tbody>
</table>
What’s wrong with this code?

```java
int count = 1;
while (count <= 10)
    System.out.println (count);
count++;
```

What’s wrong with this code?

```java
int count = 1;
while (count <= 10)
{
    System.out.println (count);
    count++;
}
```
If the condition of a `while` loop is false initially, the statement is never executed

```
System.out.println("input a number >5");
int num = scan.nextInt();

// input validation
while (num <= 5)
{
    System.out.println ("type a number >5");
    num = scan.nextInt();
}
```

- Therefore, the body of a `while` loop will execute zero or more times

Example: Input validation

```
System.out.println("Enter lifestyle code");
System.out.println ("0=bad; 1=ok; 2=super fit");

int lifestyle = scan.nextInt();

while (lifestyle < 0 || lifestyle > 2)
{
    System.out.println ("Please try again");
    System.out.println ("0=bad; 1=ok; 2=super fit");
    num = scan.nextInt();
}
```
What if we want to do a calculation over and over again?

**Example:** Calculating GPA for many students (how many? when do you stop?)

**Possible scenarios:**
- Keep accepting new inputs (for each student) and calculating and printing corresponding GPA until user quits program (infinite loop).
- Same, but ask each time whether to keep going.
- Same, but quit if the user inputs -1 for the credits (signals end)
- Calculate GPA for exactly 20 students
Nested loops

*Example: Investment problem repetition*

The repeated action (calculating the number of years it take for investment to double) involves repetition.

General pattern for algorithms: A **nested loop**

```
while (condition for repeating action)
initialize variables
while (condition for reaching goal)
calculations
print results
```

Quick Check

How many times will the string "Here" be printed?

```java
count1 = 1;
while (count1 <= 10)
{
    count2 = 1;
    while (count2 < 20)
    {
        System.out.println("Here");
        count2++;
    }
    count1++;
}
```