Loops
• Two types of Loops in Java
  – while loop
  – for loop
• Both provide ways to define repetitive behavior
  – while something is true
  – for a given number of times

While Statement
• Repeat some action as long as a boolean expression is true
  while( <boolean expression> )
  <statement>
• Repeatedly execute <statement> until the boolean expression is false.
• Note that the <statement> may never be executed
• What causes the while loop to exit?

Examples
// Valentine.java - a simple while loop
class Valentine {
    public static void main( String[] args ) {
        int howMuch = 0;
        while( howMuch++ < 5 )
            System.out.println("I love you.");
    }
} 
I love you.
I love you.
I love you.
I love you.
I love you.

While Statement

General Form of While

<initialization> // Prepare for while loop
while( <boolean expression> ) {
    <statements> // The stuff you want to repeat
    <preparation> // Prepare for next iteration
}

What makes the loop terminate?

Examples
// Valentine.java - a simple while loop
class Valentine {
    public static void main( String[] args ) {
        int howMuch = 0;
        while( howMuch < 5 ) {
            System.out.println("I love you.");
            howMuch++;
        }
    }
}
Examples

// initialization before first loop iteration
System.out.println("Type some numbers, the last one being 0.");
number = Console.in.readDouble();
while (number != 0) {
    runningTotal = RunningTotal + number;
    count++;
    // prepare for next iteration
    number = Console.in.readDouble();
}
System.out.print("The average is: ");
System.out.println( runningTotal / count );

User Input Validation

• Remember this code from Calculate.java:

    // Get diameter from user
    System.out.println("Enter the diameter as a positive number");
    diameter = Console.in.readDouble();

    // Calculate area and circumference
    radius = diameter / 2;
    area = pi * (radius * radius);
    circumference = 2 * pi * radius;

• Let’s add a while statement to make sure that the user actually enters a positive number

User Input Validation

• Think about the four general parts of a while loop:
  – Preparation before the loop
  – The boolean expression
  – The statements that you want to repeat
  – Preparation for the next iteration

Be Careful!

• When does this while loop stop?

    int i = 3;
    while ( i != 100 ) {
        System.out.println(i);
        i = i + 10;
    }

• How do you stop an executing Java program?

While and Empty Statement

• Reason for an empty statement
• Wait until something happens

    while ( isMorning() ) {
    System.out.println("It's lunchtime!");

For Statement

• The for statement has this form:

    for ( <initialize>; <boolean>; <update> )
        <statement>

    <initialize> is evaluated only once
    – before the first iteration

    <boolean> is evaluated before each iteration
    – if <boolean> is true, then <statement> is executed

    <update> is evaluated at the end of each iteration
For Statement

For Statement: Example

• Display Square Roots

```java
class SquareRoot {
    public static void main(String[] args) {
        int i;
        double squareRoot;
        for (i=1; i <= 10; i++) {
            squareRoot = Math.sqrt(i);
            System.out.println("sqrt of "+ i + " is "+ squareRoot);
        }
    }
}
```

• Display Square Roots 2

```java
class SquareRoot {
    public static void main(String[] args) {
        int i;
        double squareRoot;
        for (i=1; i <= 10; i += 2) {
            squareRoot = Math.sqrt(i);
            System.out.println("sqrt of "+ i + " is "+ squareRoot);
        }
    }
}
```

• Count down

```java
class BlastOff {
    public static void main(String[] args) {
        for (int i=10; i > 0; i--) {
            System.out.println(i);
        }
        System.out.println("Blast Off!");
    }
}
```

• Stars in a line

```java
class Stars {
    public static void main(String[] args) {
        for (int i=1; i <= 5; i++) {
            for (int j=1; j <= i; j++) {
                System.out.print("*");
            }
            System.out.println("");
        }
    }
}
```

Converting For to While

• This for loop:

```java
for (int i = 1; i < 10; i++)
    System.out.println(i);
```

• does the same thing as this while loop:

```java
int i = 1;
while (i < 10) {
    System.out.println(i);
    i++;
}
```
Converting *For* to *While*

- Convert this to use a *while* statement:
  ```java
  for (int i=10; i > 0; i--) {
      System.out.println(i);
  }
  ```

---

### Break and Continue

- *break* and *continue* interrupt the normal flow of control
- *break* causes a loop to exit immediately
- *continue* causes the next iteration of a loop to begin immediately

**Break Example**

```java
while (true) {
    System.out.print("Enter a positive integer: ");
    n = Console.in.readInt();
    if (n <= 0)
        break;        // exit loop when user enters 0
    System.out.print("square root of ");
    System.out.println(" + " + Math.sqrt(n) );
}
// break causes Java to jump to here
```

**Continue Example**

```java
while (true) {
    System.out.print("Enter a positive integer ");
    System.out.print("or 0 to exit: ");
    n = Console.in.readInt();
    if (n <= 0)
        break;        // exit loop when user enters 0
    continue;       // invalid - try again
    System.out.print("square root of ");
    System.out.println(" + " + Math.sqrt(n) );
}
// break causes Java to jump to here
```

---

### Switch statement

- Choose between a number of options
- Select one branch based on the value of an *integer-valued* expression
  - Expression can be of type *byte, short, char, int,* or *long*
General Form of Switch

- `switch` (<integer-expression>) {
  - case constant-value1:
    - <statements>
    - break;
  - case constant-value2:
    - <statements>
    - break;
  - ... more cases here ...
  - case constant-value-n:
    - <statements>
    - break;
}

Switch Example

- Print out the day of the week

```java
switch (dayOfWeek) {
  case 1:
    System.out.println("Sunday");
    break;
  case 2:
    System.out.println("Monday");
    break;
  // More cases here ...
  case 7:
    System.out.println("Saturday");
    break;
}
```

Switch: Default case

- Default case gets invoked if nothing else matches
- If there is no matching expression AND no default, then switch statement does nothing.

```java
switch (dayOfWeek) {
  case 1:
    System.out.println("Sunday");
    break;
  // More cases here ...
  default:
    System.out.println("Error! Not a day!");
}
```

Switch and Break

- Break causes the switch to exit
- If you leave it out, execution 'falls through' to the next case
  - This is a frequent source of errors.

```java
switch (dayOfWeek) {
  case 1:
    System.out.println("Sunday");
    break;
  case 2:
    System.out.println("Monday");
    break;
  // More cases here ...
}
```

Switch and Break

- Sometimes, this is what you want

```java
switch (dayOfWeek) {
  case 1:
  case 7:
    System.out.println("It's the weekend!");
    break;
  case 2:
  case 3:
  case 4:
  case 5:
  case 6:
    System.out.println("Go to school!");
}
```

Switch Processing

1. Evaluate the `switch` expression
2. Go to the `case` label that has a constant value that matches the expression.
   - A. If there is no match, go to the `default` case
   - B. If there is no `default` case, terminate `switch`
3. Execute statements until end of `switch` is reached or a `break` statement is encountered
Quiz 2 Friday

• Quiz will cover chapter 3 topics
  – Look at the review questions – if you can answer those, you should do well on the quiz
  – Example topics that I may use:
    • Converting for to while, or vice versa
    • Dangling else
    • Common programming errors with if
    • Evaluating boolean expressions

Quiz 2 Friday

• You should also be able to determine the output of short programs that contain if, if-else, while, for, or switch statements.