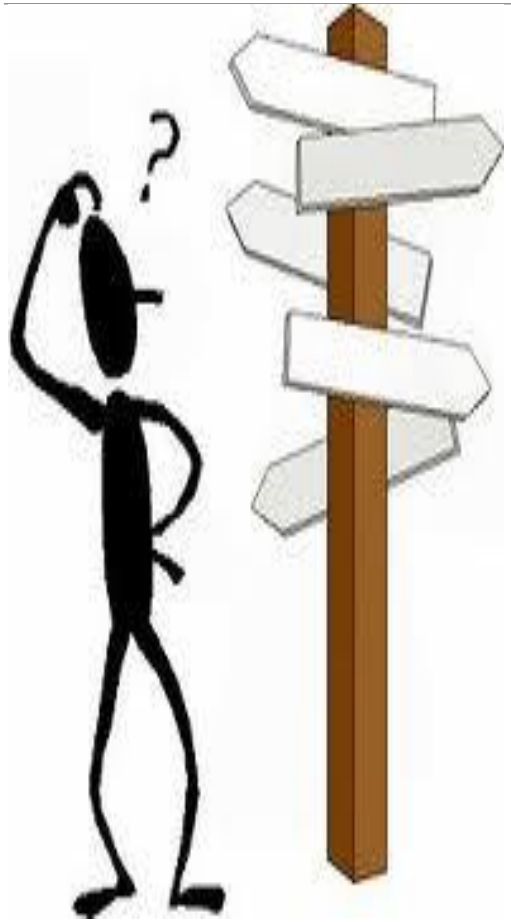
Java Class 13



Conditional Statements

A ***conditional statement*** lets us choose which statement will be executed next

Conditional statements give us the power to make basic decisions

The Java conditional statements are the:

- `if and if-else` statement
- `switch` statement (needed for Project 2)

Boolean Expressions

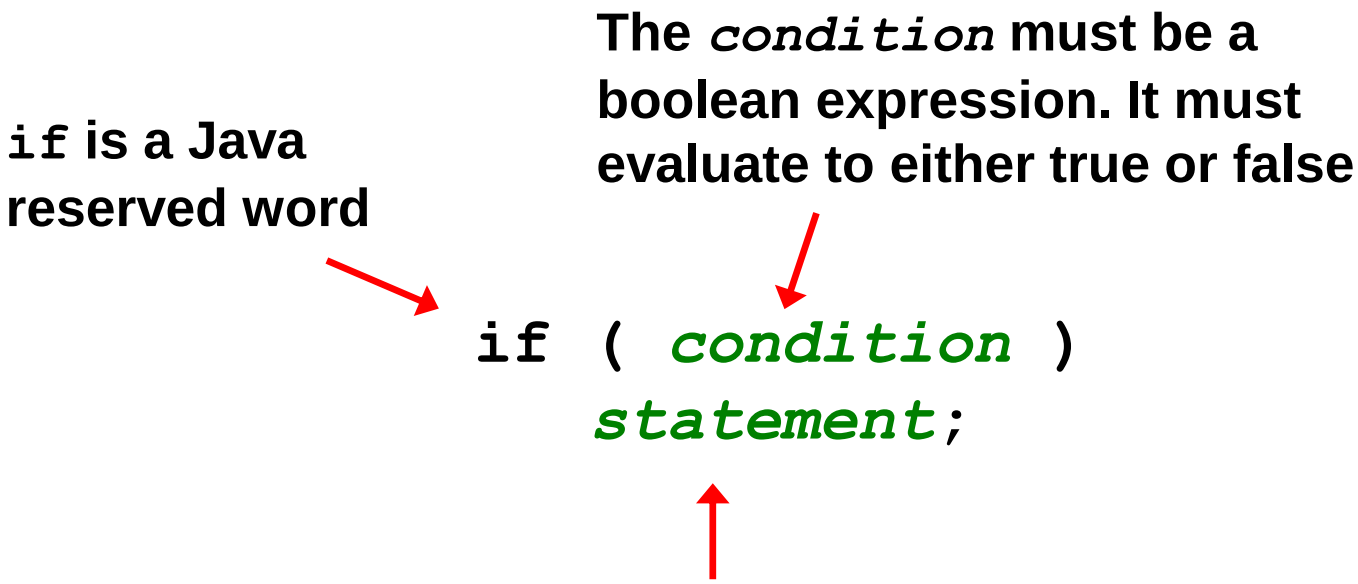
A condition often uses one of Java's ***equality operators*** or ***relational operators***, which all return boolean results:

==	equal to
!=	not equal to
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to

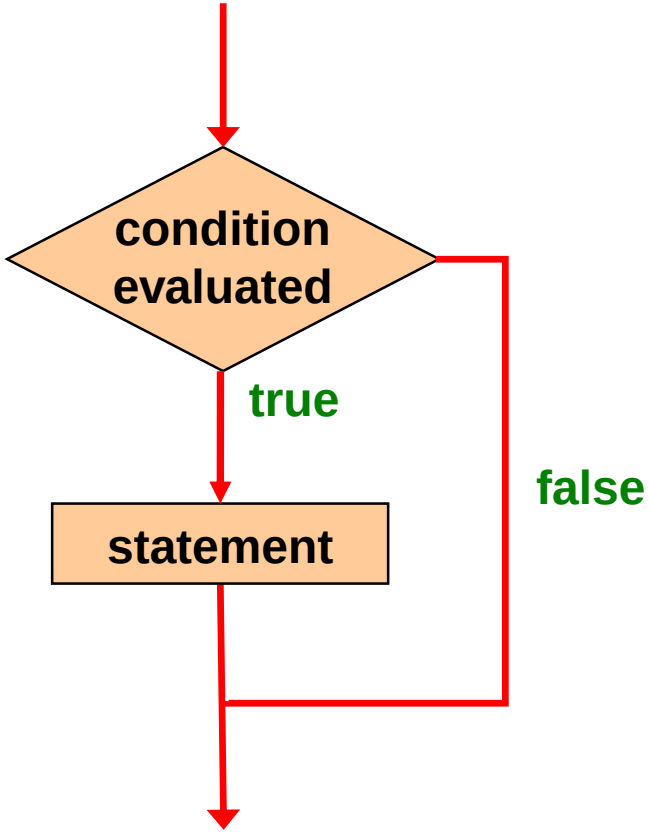
Note the difference between the equality operator (==) and the assignment operator (=)

The if Statement

The *if statement* has the following syntax:



Logic of an if Statement



Boolean Expressions

An `if` statement with its boolean condition:

```
if (sum > MAX)
    delta = sum - MAX;
```

First, the condition is evaluated; the value of `sum` is greater than the value of `MAX`, or it is not

If the condition is true, the assignment statement is executed; if it isn't, it is skipped

Logical Operators

Boolean expressions can also use the following ***logical operators***:

!	Logical NOT
&&	Logical AND
||	Logical OR

They all take boolean operands and produce boolean results

Logical NOT is a unary operator (it operates on one operand)

Logical AND and logical OR are binary operators (each operates on two operands)

Logical NOT

The *logical NOT* operation is also called *logical negation* or *logical complement*

If some boolean condition `a` is true, then `!a` is false; if `a` is false, then `!a` is true

Logical expressions can be shown using a ***truth table***:

`a`	`!a`
true	false
false	true

Logical AND and Logical OR

A truth table shows all possible true-false combinations of the terms

Since `&&` and `||` each have two operands, there are four possible combinations of conditions `a` and `b`

a	b	a && b	a || b
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

Logical Operators

Expressions that use logical operators can form complex conditions

```
if (total < MAX+5 && !found)
   System.out.println ("Processing…");
```

All logical operators have lower precedence than the relational operators

The ! operator has higher precedence than && and ||

Boolean Expressions

Specific expressions can be evaluated using truth tables

`total < MAX + 5`	`found`	`!found`	`total < MAX + 5 && !found`
false	false	true	false
false	true	false	false
true	false	true	true
true	true	false	false

Quick Check

Given the following declarations, what is the value of each of the listed boolean conditions?

```
int value1 = 5, value2 = 10;
boolean done = true;
```

`value1 <= value2`	true
`(value1 + 5) >= value2`	true
`value1 < value2 / 2`	false
`value2 != value1`	true
`!(value1 == value2)`	true
`(value1 > value2) || done`	true
`(value1 < value2) && !done`	false
`done || !done`	true
`((value1 > value2) || done) && (!done || (value2 > value1))`	true

Short-Circuited Operators

The processing of `&&` and `||` is "short-circuited"

If the left operand is sufficient to determine the result, the right operand is not evaluated

```
if (count != 0 && total/count > MAX)
   System.out.println ("Testing.");
```

If `count` equals `0`, then `total/count > MAX` is never evaluated

This type of processing should be used carefully

Indentation

The statement controlled by the `if` statement is indented to indicate that relationship

The use of a consistent indentation style makes a program easier to read and understand

The compiler ignores indentation, which can lead to errors if the indentation is not correct

"Always code as if the person who ends up maintaining your code is a violent psychopath who knows where you live."

-- Martin Golding

The if-else Statement

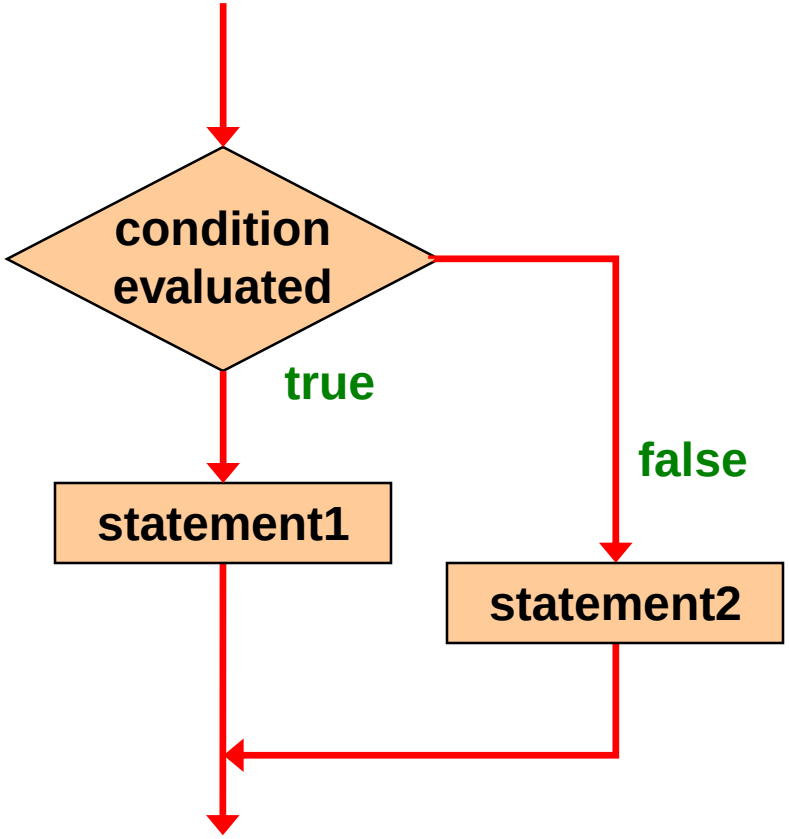
An ***else clause*** can be added to an `if` statement to make an *if-else statement*

```
if ( condition )
   statement1;
else
   statement2;
```

If the *condition* is true, *statement1* is executed; if the condition is false, *statement2* is executed

One or the other will be executed, but not both

Logic of an if-else Statement



The Coin Class

Let's look at an example that uses a class that represents a coin that can be flipped

Instance data is used to indicate which face (heads or tails) is currently showing

```
//********************************************************************
//  Coin.java       Author: Lewis/Loftus
//
//  Represents a coin with two sides that can be flipped.
//********************************************************************

public class Coin
{
   private final int HEADS = 0;
   private final int TAILS = 1;

   private int face;

   //-----------------------------------------------------------------
   //  Sets up the coin by flipping it initially.
   //-----------------------------------------------------------------
   public Coin ()
   {
      flip();
   }
```

Instance variables

Constructor

continue

```
//-----------------------------------------------------------------
//  Flips the coin by randomly choosing a face value.
//-----------------------------------------------------------------
public void flip ()
{
   face = (int) (Math.random() * 2);
}
```

flip method

```
//-----------------------------------------------------------------
//  Returns true if the current face of the coin is heads.
//-----------------------------------------------------------------
public boolean isHeads ()
{
   return (face == HEADS);
}
```

isHeads method

continue

```
    //-----------------------------------------------------------------
    //  Returns the current face of the coin as a string.
    //-----------------------------------------------------------------
    public String toString()
    {
        String faceName;

        if (face == HEADS)
            faceName = "Heads";
        else
            faceName = "Tails";

        return faceName;
    }
}
```

toString method

```
//********************************************************************
//  CoinFlip.java       Author: Lewis/Loftus
//
//  Demonstrates the use of an if-else statement.
//********************************************************************

public class CoinFlip
{
   //-----------------------------------------------------------------
   //  Creates a Coin object, flips it, and prints the results.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      Coin myCoin = new Coin();

      myCoin.flip();

      System.out.println (myCoin);

      if (myCoin.isHeads())
         System.out.println ("You win.");
      else
         System.out.println ("Better luck next time.");
   }
}
```

Sample Run

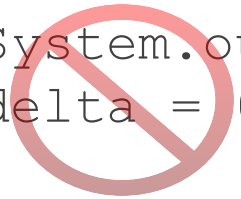
```
Tails
Better luck next time.
```

```
//********************                                  **********************
//  CoinFlip.java
//
//  Demonstrates the                                    .
//********************                                  **********************

public class CoinFlip
{
   //-----------------------------------------------------------------
   //  Creates a Coin object, flips it, and prints the results.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      Coin myCoin = new Coin();

      myCoin.flip();

      System.out.println (myCoin);

      if (myCoin.isHeads())
         System.out.println ("You win.");
      else
         System.out.println ("Better luck next time.");
   }
}
```

Indentation Revisited

Remember that indentation is for the human reader and is ignored by the compiler

```
if (depth >= UPPER_LIMIT)
   delta = 100;
else
   System.out.println("Reseting Delta");
   delta = 0;
```



Despite what the indentation implies, `delta` will be set to 0 no matter what

Block Statements

Several statements can be grouped together into a ***block statement*** delimited by braces

A block statement can be used wherever a statement is called for in the Java syntax rules

```
if (total > MAX)
{
   System.out.println ("Error!!");
   errorCount++;
}
```

Block Statements

The `if` clause, or the `else` clause, or both, could be governed by block statements

```
if (total > MAX)
{
   System.out.println ("Error!!");
   errorCount++;
}
else
{
   System.out.println ("Total: " + total);
   current = total*2;
}
```

```
//********************************************************************
//  Guessing.java       Author: Lewis/Loftus
//
//  Demonstrates the use of a block statement in an if-else.
//********************************************************************

import java.util.*;

public class Guessing
{
   //-----------------------------------------------------------------
   //  Plays a simple guessing game with the user.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      final int MAX = 10;
      int answer, guess;

      Scanner scan = new Scanner (System.in);
      Random generator = new Random();

      answer = generator.nextInt(MAX) + 1;
```

continue

continue

```
      System.out.print ("I'm thinking of a number between 1 and "
                        + MAX + ". Guess what it is: ");

      guess = scan.nextInt();

      if (guess == answer)
         System.out.println ("You got it! Good guessing!");
      else
      {
         System.out.println ("That is not correct, sorry.");
         System.out.println ("The number was " + answer);
      }
   }
}
```

Sample Run

```
I'm thinking of a number between 1 and 10. Guess what it is: 6
That is not correct, sorry.
The number was 9
```

```
      if (guess == answer)
         System.out.println ("You got it! Good guessing!");
      else
      {
         System.out.println ("That is not correct, sorry.");
         System.out.println ("The number was " + answer);
      }
   }
}
```

Nested if Statements

The statement executed as a result of an `if` or `else` clause could be another `if` statement

These are called ***nested if statements***

An `else` clause is matched to the last unmatched `if` (no matter what the indentation implies)

Braces can be used to specify the `if` statement to which an `else` clause belongs

```
//********************************************************************
//  MinOfThree.java       Author: Lewis/Loftus
//
//  Demonstrates the use of nested if statements.
//********************************************************************

import java.util.Scanner;

public class MinOfThree
{
   //-----------------------------------------------------------------
   //  Reads three integers from the user and determines the smallest
   //  value.
   //-----------------------------------------------------------------
   public static void main (String[] args)
   {
      int num1, num2, num3, min = 0;

      Scanner scan = new Scanner (System.in);

      System.out.println ("Enter three integers: ");
      num1 = scan.nextInt();
      num2 = scan.nextInt();
      num3 = scan.nextInt();
```

continue

continue

```
        if (num1 < num2)
            if (num1 < num3)
                min = num1;
            else
                min = num3;
        else
            if (num2 < num3)
                min = num2;
            else
                min = num3;

        System.out.println ("Minimum value: " + min);
    }
}
```

continue

```
        if (num1 < num2)
            if (num1 < num3)
                min = num1;
            else
                min = num3;
        else
            if (num2 < num3)
                min = num2;
            else
                min = num3;

        System.out.println ("Minimum value: " + min);
    }
}
```

Sample Run

```
Enter three integers:
84   69   90
Minimum value: 69
```

Quick Check

What output is produce by the following code fragment given the assumptions below?

```
if (num1 < num2)
   System.out.print (" red ");
if ((num1 + 5) < num2
   System.out.print (" white ");
else
   System.out.print (" blue ");
System.out.println (" yellow ");
```

Assuming the value of num1 is 2 and the value of num2 is 10?

red white yellow

Assuming the value of num1 is 10 and the value of num2 is 2?

blue yellow

Assuming the value of num1 is 2 and the value of num2 is 2?

blue yellow

Group Exercises

Ex: 5.1

Ex: 5.2

Ex: 5.3

Ex: 5.4

Ex: 5.5

Assignment for Class 14

Review Coin, CoinFlip, Guessing, MinOfThree

Read Chapter 5.3, 5.4, 5.5, 5.6