# Fundamentals of Java

Selection

Copyright © 2012 University of Technology, Sydney

# The "if" statement

```
if (condition)
{
    statement1;
    statement2;
    statement3;
}
```

If the condition is true,
then execute the statements.

# The "if" statement - example

```
System.out.print("What is your score? ");
int score = keyboard.nextInt();

if (score >= 85)
{
    System.out.println("Well done!");
    System.out.println("Your grade is HD.");
}
```

# The "if/else" statement

```
System.out.print("What is your score? ");
int score = keyboard.nextInt();

if (score >= 85)
{
    System.out.println("Your grade is HD.");
}
else
{
    System.out.println("Better luck next time.");
}
```

# Nested if/else statements

```
if (score >= 85)
{
    System.out.println("Your grade is HD.");
}
else
{
    if (score >= 75)
    {
        System.out.println("Your grade is D.");
    }
    else
    {
        System.out.println("Better luck next time.");
    }
}
```

# Braces

The opening { and closing } braces are used to group a sequence of statements together as a single unit.

Braces are required around more than one statement.

Braces are not required around a single statement.

# Optional braces removed

```
if (score >= 85)
    System.out.println("Your grade is HD.");
else
    if (score >= 75)
        System.out.println("Your grade is D.");
    else
        if (score >= 65)
            System.out.println("Your grade is C.");
        else
            if (score >= 50)
                System.out.println("Your grade is P.");
            else
                System.out.println("Your grade is Z.");
```

# Cascading if/else

A long chain of nested if/else statements is called a cascading if/else statement.

(It looks like a cascading waterfall).

The problem: cascading if/else statements quickly drift to the right, and off the page.

Programmers have a special way to format cascading if/else statements (next slide...)

# Optional braces removed

```
if (score >= 85)
    System.out.println("Your grade is HD.");
else if (score >= 75)
    System.out.println("Your grade is D.");
else if (score >= 65)
    System.out.println("Your grade is C.");
else if (score >= 50)
    System.out.println("Your grade is P.");
else
    System.out.println("Your grade is Z.");
```

This is exactly the same code sequence, only with different whitespace formatting.

The compiler ignores whitespace, so this is exactly the same program to the compiler.

# Common mistake #1

```
int x = 92;
if (x < 10)
    System.out.println("hello");
    System.out.println("hello");

System.out.println("hello");
```

How many times is "hello" printed?

# Common mistake #1

```
int x = 92;
if (x < 10)
    System.out.println("hello");
    System.out.println("hello");

System.out.println("hello");
```

How many times is "hello" printed?

Answer: twice!

# Common mistake #2

```
int x = 92;
if (x < 10);
    System.out.println("hello");

System.out.println("hello");
```

How many times is "hello" printed?

# Common mistake #2

```
int x = 92;
if (x < 10);
    System.out.println("hello");

System.out.println("hello");
```

How many times is "hello" printed?

Answer: twice!

# Indentation style

Java ignores TABs, spaces and newlines.

But humans do not ignore them.

Use TABs, spaces and newlines to make your code readable by other humans.

# Indentation style

Use a TAB to shift code to the right in the following cases:

- The contents of a class declaration
- The contents of a method declaration
- The contents of an "if"
- The contents of an "else"
- ... more cases to come...

# Compound conditions - AND

```
if (score >= 65 && score < 75)
    System.out.println("Your score is average.");
```

If the score is greater than or equal to 65
AND the score is less than 75
THEN print "Your score is average."

# Compound conditions - OR

```
if (yearsMarried == 25 || yearsMarried == 50)
    System.out.println("Happy Anniversary!");
```

If yearsMarried equals 25
OR yearsMarried equals 50
THEN print "Happy Anniversary!"

# Compound conditions - NOT

!A means NOT A.

```
if (!(yearsMarried == 25 || yearsMarried == 50)
    System.out.println("It is not your
anniversary.");
```

An equivalent expression is:

```
if (yearsMarried != 25 && yearsMarried != 50)
    System.out.println("It is not your
anniversary.");
```

# Translating English to Java

If your age is at least 18 and either your bank balance is over $10,000 or your salary is at least $40,000 then ...

# Which translation is correct?

If your age is at least 18 and either your bank balance is over $10,000 or your salary is at least $40,000 then …

## Translation #1

```
if (age >= 18 && balance > 10000 || salary >= 40000) …
```

## Translation #2

```
if (age >= 18 && (balance > 10000 || salary >= 40000)) …
```

# Which translation is correct?

If your age is at least 18 and either your bank balance is over $10,000 or your salary is at least $40,000 then …

Translation #1 (incorrect)

```
if (age >= 18 && balance > 10000 || salary >= 40000) …
```

Translation #2 (correct)

```
if (age >= 18 && (balance > 10000 || salary >= 40000)) …
```

# The "boolean" primitive type

The condition of an "if" statement is an expression of type "boolean". You can declare a variable of type boolean.

```
boolean passed = grade >= 50;
if (passed)
    System.out.println("You passed!");
```

This is equivalent to:

```
if (grade >= 50)
    System.out.println("You passed!");
```

# Boolean expressions

| | |
|---|---|
| A == B | A is equal to B |
| A != B | A is not equal to B |
| A > B | A is greater than B |
| A < B | A is less than B |
| A >= B | A is greater than or equal to B |
| A <= B | A is less than or equal to B |
| A && B | A is true AND B is true |
| A \|\| B | A is true OR B is true |
| !A | A is not true |
| (A) | A |

# Primitive vs object equality

```
int x = 3;
int y = 3;
if (x == y) System.out.println("equal");

Circle c1 = new Circle(2.0);
Circle c2 = new Circle(2.0);
if (c1 == c2) System.out.println("equal");
```

Is x == y?

Is c1 == c2?

# Primitive vs object equality

```
int x = 3;
int y = 3;
if (x == y) System.out.println("equal");

Circle c1 = new Circle(2.0);
Circle c2 = new Circle(2.0);
if (c1 == c2) System.out.println("equal");
```

Is x == y? **YES**

Is c1 == c2? **NO!**

# Object equality

The == equality operator on objects compares whether the two operands refer to the SAME object.

In this case, c1 and c2 are NOT the same object. They are different circle objects that "happen" to have the same radius.

```
Circle c1 = new Circle(2.0);
Circle c2 = new Circle(2.0);
```

# Object equality

Compare:

```
Circle c1 = new Circle(2.0);
Circle c2 = new Circle(2.0);
Circle c3 = c1;
```

Is c1 == c2? No.

Is c1 == c3? Yes.

# Comparing similar objects

If you want to test whether two circles are the same size, then use:

```
if (c1.getRadius() == c2.getRadius())
```

Usually programmers define a method called "equals" which you should use:

```
if (c1.equals(c2))
```

# The equals method

```
public class Circle
{
    private double radius;
    ...
    public boolean equals(Circle other)
    {
        return this.radius == other.radius;
    }
}
```

# Switch statement

A more efficient alternative to the cascading if/else.

```
switch (place) {
    case 1:
        System.out.println("Gold medal!");
        break;
    case 2:
        System.out.println("Silver medal!");
        break;
    case 3:
        System.out.println("Bronze medal!");
        break;
    default:
        System.out.println("Sorry, you did not place.");
        break;
}
```

# Switch entry and exit points

The entry point of the switch statement is the case that matches the argument `place`.

The exit point of the switch statement is the first `break` statement to be reached after the entry point.

Normally, use one `break` for each case.

If you forget to `break`, the case will fall through to the next case!!!

# Switch - missing break

```
switch (place) {
    case 1:
        System.out.println("Gold medal!");
    case 2:
        System.out.println("Silver medal!");
    case 3:
        System.out.println("Bronze medal!");
    default:
        System.out.println("Sorry, you did not place.");
}
```

If `place` is 2, this will print:

```
Silver medal!
Bronze medal!
Sorry, you did not place.
```

# Intentionally missing a break

```
switch (month) {
    case 9:
    case 4:
    case 6:
    case 11:
        System.out.println("30 days.");
        break;
    case 2:
        if (year % 4 == 0)
            System.out.println("29 days.");
        else
            System.out.println("28 days.");
        break;
    default:
        System.out.println("31 days.");
        break;
}
```