06: Decisions

If Statements

If Statements

If Statements

- An `if` statement will make a decision as to whether it should or should not execute a block of code.

Code.

```
if (boolean expression is true) {
    execute these statements;
}
```

- If the boolean expression in the if condition is false, then the statements are never executed.

If Statements as a flowchart

```
   / \
  /   \
 /     \     Yes.
Is it cold? -----> Wear a coat.
 \     /                v
  \   /                 |
   \ /                  |
    | No.               |
    | <-----------------|
    v
```

If it is cold outside, we will put on a coat. Otherwise we do nothing.

If Statements

There are two forms of the `if` statement. You can use this if you need to execute only one statement.

```
if (boolean expression)
    statement;
```

The second form requires curly brackets for one or more statements.

```
if (boolean expression) {
    statement;
    statement;
    statement;
}
```

Curly brackets are optional if you have one statement. I always recommend using curly brackets regardless of how many statements you need to execute.

Relational Operators

- Relational operators are used to compare numbers to determine relative order.
  - $a > b$ True if a is greater than b.
  - $a < b$ True if a is less than b.
  - $a >= b$ True if a is greater than or equal to b.
  - $a <= b$ True if a is less than or equal to b.
  - $a == b$ True if a is equal to b.
  - $a! = b$ True if a is not equal to b.
- Helpful to remember: There are 4 relational operators comprised of 2 characters. In each case, the = comes second.
- Also helpful: = means “assignment”, == means “equals”.

Style

Each of these if statements are functionally the same. (Personally, I like the last one.)

```
if (score >= 90) grade = 'A';

if (score >= 90)
    grade = 'A';

if (score >= 90) { grade = 'A'; }

if (score >= 90) {
    grade = 'A';
}
```

Relational Operators and their opposites.

- It’s instinctive to think that the opposite of “less than” is “greater than”, but this is not correct.
- The opposite of “less than” is “greater than or equal to”.
- The opposite of “greater than” is “less than or equal to”.
- The opposite of “equals” is “not equals”.

Quick Example.

You should pay special attention to the next few examples. They represent simple uses or mistakes that new programmers will make. All four examples represent Java code which may or may not compile.

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == y) {
    System.out.println("1 is equal to 0!");
}
```

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == y) {
    System.out.println("1 is equal to 0!");
}
```

This code prints nothing to the screen because 1 is not 0.

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == 0) {
    System.out.println("x is equal to 0!");
}
```

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == 0) {
    System.out.println("x is equal to 0!");
}
```

This code prints “x is equal to 0!” because x does equal 0.

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x = y) {
    System.out.println("1 is equal to 0!");
}
```

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x = y) {
    System.out.println("1 is equal to 0!");
}
```

This code will not compile because “$x = y$” does not use one of our 6 relational operators.

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == 1); {
    System.out.println("x is equal to 1!");
}
```

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == 1); {
    System.out.println("x is equal to 1!");
}
```

This code prints “x is equal to 1!”. The semicolon represents a full statement. The following curly braces represent an unrelated code block. This is yet another easy mistake to make.

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == y)
    System.out.println("x is equal to y!");
    System.out.println("x is equal to 0!");
    System.out.println("y is equal to 1!");
```

Quick Example.

What will this code print?

```
int x = 0;
int y = 1;
if (x == y)
    System.out.println("x is equal to y!");
    System.out.println("x is equal to 0!");
    System.out.println("y is equal to 1!");
```

This code prints “x is equal to 0!” and “y is equal to 1!”. Since there is no use of curly brackets, the first statement under the if is associated with the if statement and the last two are not.

if Statement Notes

- Do not place ; after (expression).

New Project: GreenAndRed

- Create a new project called “GreenAndRed”.
- Import the TurtleLogo.jar file.
- We will ask the user for an integer representing the size of the desired square BUT will change the color of the square under certain conditions.
    - If the square size is greater than 50, we will make the square GREEN.
    - If the square size is greater than 100, we will make the square RED.
    - Otherwise, we will make the square the default color of BLACK.

Just like last time.

Modify the line containing “public class GreenAndRed {” to look like this.

```
public class GreenAndRed extends Sandbox
```

You will have to fix your code’s imports to make the error on “Sandbox” go away.

Start Programming.

In your main method, add one line of code. It will look like this.

```
public static void main(String[] args) {
    launch(args);
}
```

Place a turtle

Create a new method called “draw”.

```
@Override
public void draw() {
    Turtle turtle = new Turtle();
    add(turtle);
}
```

The error on “Turtle” will require you to fix your imports again.

Ask the user for a size.

```
Scanner keyboard = new Scanner(System.in);
System.out.println("Enter size of square: ");
int size = keyboard.nextInt();
```

Change the color based on the size.

If the size is greater than 50, we will set the color to green. If the size is greater than 100, we will set the color to red.

```
if (size > 50) {
    turtle.setColor(Color.GREEN);
}

if (size > 100) {
    turtle.setColor(Color.RED);
}
```

You will need to “Fix Imports” and you must select “import javafx.scene.paint.Color;”.

Draw the square.

Now we will draw a square based on the size that the user wants. Notice that we no longer use “100” in our call to `forward`.

```
turtle.forward(size);
turtle.left(90);
turtle.forward(size);
turtle.left(90);
turtle.forward(size);
turtle.left(90);
turtle.forward(size);
turtle.left(90);
```

Run your program several times.

You will need to run your program several times. Type different numbers at the prompt in NetBeans and look at the results. Try with these values:

- 25
- 50
- 75
- 100
- 125

Notice the color of the square each time the square is drawn.

Fill the rest of the hour

Add more if statements. Each time you add an if statement, check to see if your program works with all of the previous values.

- Make the square color BLUE if the size is greater than 150.
- Make the square color ORANGE if the size is exactly 200, but still BLUE if the size is 199 or 201.
- Make the square color PINK if the size is exactly 42.