# WIT COMP1000

## Simple Control Flow:

## `if-else` statements

# Control Flow

- Control flow is the order in which program statements are executed
- So far, all of our programs have been executed straight-through from the first statement to the last
- In general, you will need more complex control flow
- For example, to choose between two (or more) possibilities

# Branching

- Human Example: greeting two different people

# Basics Steps

- Thought process:
  - » Whom am I looking at?
  - » If I am looking at Harry Potter
    - Say "'ello Harry!"
  - » If I am looking at Hermione Granger
    - Say "'ello Hermione!"
- Java uses **`if-else`** statements to choose between alternatives

# if-else

- Example:

```
if (user_id == 1742) {
    System.out.println("'ello Harry!");
}
else {
    System.out.println("'ello Hermione!");
}
```

- Generic form:

```
if (BOOLEAN EXPRESSION) {
    YES/TRUE STATEMENTS
}
else {
    NO/FALSE STATEMENTS
}
```

# if-else

- Each **if-else** statement allows your program to do one of two different things
- In other words, you EITHER use one set of statements OR you use the other
- The statements in between the { } for the **if** and the **else** can contain any code you want, just like code that is not inside an **if-else**
  - » Even other **if-else** statements!

# Boolean Expressions

- Boolean expressions (like **`boolean`** variables) are either true or false, and are composed of comparisons
- Comparison Operators

| | | |
|---|---|---|
| `==` | Equal To | `if (x == 10)` |
| `!=` | Not Equal To | `if (x != 5)` |
| `<` | Less Than | `if (hours < 40)` |
| `<=` | Less Than or Equal To | `if (dollars <= 100)` |
| `>` | Greater Than | `if (dollars > 100)` |
| `>=` | Greater than or Equal To | `if (hours >= 40)` |

# Example

```
import java.util.Scanner;

public class ClassExamples {
     public static void main(String[] args) {

          Scanner input = new Scanner(System.in);

          double input_value;
          System.out.print("Please enter a number: ");

          input_value = input.nextDouble();

          if (input_value > 100) {
               System.out.println("That is greater than 100.");
          }
          else {
               System.out.println("That is less than or equal to 100.");
          }

          System.out.println("Thank you!");
     }
}
```

# Control Flow

- So, if the `if` condition is true, the statements between the next curly braces are executed, then it skips down past the `else` curly braces and continues executing
- If the `if` condition is false, the statements between the `if` braces are ignored, then the statements between the `else` braces are executed, then it continues executing whatever is next

# Execution Example

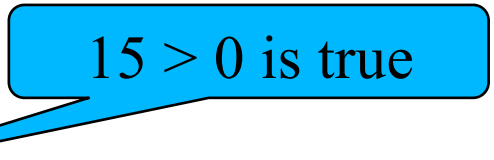
```
import java.util.Scanner;

public class ClassExamples {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        int x = 15;

        if (x > 0) {
            System.out.println("x is positive");
        }
        else {
            System.out.println("x in non-positive");
        }

        System.out.println("Thank you, and good night.");
    }
}
```



# Exercise

- Write a program that reads an integer from the user and determines whether or not the integer is even (evenly divisible by 2) or odd (not evenly divisible by 2)

# Answer

```
import java.util.Scanner;

public class ClassExamples {
     public static void main(String[] args) {
          Scanner input = new Scanner(System.in);

          int value;
          int remainder;

          System.out.print("Enter an integer: ");
          value = input.nextInt();

          remainder = value % 2;

          if (remainder ==  0) {
               System.out.println(value + " is even");
          }
          else {
               System.out.println(value + " is odd");
          }
     }
}
```

# String Equality

- Recall that `String` variables are actually *objects* which means that they follow slightly different rules
- If you want to compare two `String` variables together, you must use the `String equals()` method
  - Example: `string_variable.equals("Something")`
- You can also compare two `String` variables
  - Example: `str1.equals(str2)`

# String Comparison Example

```
Scanner input = new Scanner(System.in);

System.out.print("Enter a name: ");
String user = input.next();

if (user.equals("Harry")) {
    System.out.println("'ello Harry!");
}
else {
    System.out.println("'ello Hermione!");
}
```

# Omitting Braces

- You can omit the curly braces after an if or else ONLY if there is exactly one statement
- Example:

```
if (hours_worked > 8)
   System.out.println("It's quittin' time!");
else
   System.out.println("Get back to work!");
```

- I recommend always using the braces
  - » Easier to read
  - » Less chance of errors if you add more statements later on

# Multiple Choices

- Standard `if-else` statements give you two choices
- You can have more choices by adding `else if` statements in between the `if` and `else`
- Any number of `else if` statements can be added after an `if` statement
- Each will be tested in order until one of the conditions is true, and the `else` statements will only run if none of the conditions are true
- You can always omit the `else` statement if you want (even if you have no `else if` statements)

# else if Statements

```
if (EXPRESSION1) {
   //run statements if expression 1 is true
}
else if (EXPRESSION2) {
  //run statements if expressions 1 is false,
  //and expression 2 is true
}
else {
  //run statements if both expressions are false
}
```

# Example

```
if (x > 10) {
    System.out.println("x is greater than 10");
}
else if (x < 5) {
    System.out.println("x is less than 5");
}
else {
    System.out.println("x is between 5 and 10, inclusive");
}
```

# Example

```
if (x > 10) {
    System.out.println("x is greater than 10");
}
else if (x > 5) {
    System.out.println("x is between 6 and 10, inclusive");
}
else {
    System.out.println("x is less than or equal to 5");
}
```

# Multiple `else if` Statements

```
if (x > 10) {
    System.out.println("x is greater than 10");
}
else if (x > 5) {
    System.out.println("x is between 6 and 10, inclusive");
}
else if (x > 0) {
    System.out.println("x is between 1 and 5, inclusive");
}
else {
    System.out.println("x is less than 1");
}
```

# Exercise

- Write a program that reads a numeric score from the user and outputs a letter grade
  - If the score is >=90, output A
  - If the score is >=80 and <90, output B
  - If the score is >=70 and <80, output C
  - If the score is >=60 and <70, output D
  - If the score is <60, output F

# Answer

```
Scanner input = new Scanner(System.in);

int score;
System.out.print("Enter your score: ");
score = input.nextInt();

System.out.print("Letter Grade: ");
if (score >= 90) {
     System.out.println("A");
}
else if (score >= 80) {
     System.out.println("B");
}
else if (score >= 70) {
     System.out.println("C");
}
else if (score >= 60) {
     System.out.println("D");
}
else {
     System.out.println("F");
}
```

# Complex Boolean Expressions

- Multiple comparisons can be combined with the "and" and "or" operators
- && is the "and" operator
  - » Example: `(hours > 20) && (hours <= 40)`
  - » Entire expression is true only if *both* comparisons are true, and false if *either* is false
- || is the "or" operator
  - » Example: `(x < 17) || (y < 22)`
  - » Entire expression is true if *either* comparison is true, and false only if *both* are false

# Examples

```
// assume x = 5
(x >= 10) && (x != 12)
// false, (5 >= 10) is false, so it
   doesn't matter if (x != 12) is true

// assume x = 48
(x >= 10) && (x != 12)
// true, (48 >= 10) is true and
   (48 != 12) is true

// assume x = 12
(x >= 10) && (x != 12)
// false, (12 >= 10) is true but
   (12 != 12) is false

// assume x = 12
(x >= 10) || (x != 12)
// true, (12 >= 10) is true, so it
   doesn't matter if (x != 12) is true

// assume x = 5
(x >= 10) || (x != 12)
// true, (5 >= 10) is false, but
   (5 != 12) is true
```

# Time Example

```
Scanner input = new Scanner(System.in);

int hours;
System.out.print("Enter the hours in 24 hour notation: ");
hours = input.nextInt();

if ((hours >= 1) && (hours <= 11)) {
     System.out.println("That is " + hours + "am");
}
else if ((hours >= 13) && (hours <= 23)) {
     hours = hours - 12;
     System.out.println("That is " + hours + "pm");
}
else if (hours == 12) {
     System.out.println("That is " + hours + "pm");
}
else if (hours == 0) {
     hours = hours + 12;
     System.out.println("That is " + hours + "am");
}
else {
     System.out.println("That is not a valid hour!");
}
```

# Pin Hole Camera Example

```
Scanner input = new Scanner(System.in);

double focal_length; // f
double object_distance; // o
double image_distance; // i
double step1;
System.out.print("Enter the focal length: ");
focal_length = input.nextDouble();
System.out.print("Enter the object distance: ");
object_distance = input.nextDouble();

if ((focal_length == 0) || (object_distance == 0)) {
     System.out.println("Please enter non-zero values.");
     System.exit(0);
}

// solve the thin lens equation: 1/i = 1/f - 1/o
step1 = (1/focal_length) - (1/object_distance);
if (step1 == 0) {
     System.out.println("The image is infinitely far away.");
}
else {
     image_distance = 1 / step1;
     System.out.println("The image distance is " + image_distance + ".");
}
```

# Sanitizing User Inputs

- When you get input from the user, you should always check that the values they entered are reasonable
  - » Otherwise you might have a GIGO scenario: Garbage In, Garbage Out
  - » That is, if they input "bad" values into your program then they will get "bad" results
- You can stop your program early by adding a new statement: `System.exit(0);`
  - » Typically after you detect an error or bad input value
  - » When executed, your program end immediately

# Exercise

- Write a program that reads two exam scores from the user. If both exam scores are greater than 90, print "Way to go!". If either exam score is less than 70, print "Study more!". Otherwise, print "Keep it up!".

# Answer

```
Scanner input = new Scanner(System.in);

int exam1;
int exam2;

System.out.println("Enter your two exam scores:");
exam1 = input.nextInt();
exam2 = input.nextInt();

if ((exam1 > 90) && (exam2 > 90)) {
    System.out.println("Way to go!");
}
else if ((exam1 < 70) || (exam2 < 70)) {
    System.out.println("Study more!");
}
else {
    System.out.println("Keep it up!");
}
```

# Common Mistakes

- When using comparison operators, always use "==" for checking equality, never "="
  - Some situations will not give you any errors or warnings, but your program will not operate as you expect
- You can not put multiple comparisons together without the "and" or "or" operators
  - Incorrect: `if (10 < x < 15)`
  - Correct: `if ((10 < x) && (x < 15))`

# Take Home Points

- Use `if-else` statements when you need to choose between two or more possibilities
- Always put boolean expressions in parentheses
- Multiple expressions can be put together with the "and" (&&) and "or" (||) operators
  - Always put each comparison in its own set of parentheses
- Always use "==" for testing equality, never "="