Lecture 03
Selections
Back to tip calculator

• What happens if you entered a negative number for the total?

• Might seem silly, but when your data comes from a user or another source, you need to be able to handle it.

• Sanitizing input
Pseudocode

// Check to see if the entered number is negative
// if it's negative, print out an error
// if it's positive, finish the program
If statement: syntax

```java
if (boolean-expression){
    statement(s);
}

if (length == 10){
    System.out.println(length);
}
```
Boolean expression

- Check if something is true or false.
- `boolean` datatype can be used:
  - `boolean isPositiveNumber = true;`
- `true` and `false` are called literals. Reserved words that cannot be used for variable names, etc.
• <   less than

• <=  less than or equal to

• >   greater than

• >=  greater than or equal to

• ==  equal to

• !=  not equal to

**TIP!** to check if values are equal, use == not =. (That’s an assignment statement)
public class BooleanExpressions {

    public static void main(String[] args) {
        int number = 7;

        System.out.println(number < 0);
        System.out.println(number <= 0);
        System.out.println(number > 0);
        System.out.println(number >= 0);
        System.out.println(number == 0);
        System.out.println(number != 0);
    }
}
import java.util.Scanner;

public class TipCalculatorSanitized {

    public static void main(String[] args) {
        // Set up variables
        double subtotal;
        double tipAmount;
        double tipTotal;
        double total;
        Scanner input = new Scanner(System.in);

        // Read subtotal from console
        System.out.print("Enter the subtotal: ");
        subtotal = input.nextDouble();

        if (subtotal < 0){
            System.out.println("Sorry, that's a negative total");
        }
    }
}
If-else

```java
if (boolean-expression){
    statement(s);
}
else if (next-expression) {
    statement(s);
} else {
    statement(s);
}
```
import java.util.Scanner;

public class TipCalculatorSanitized {

    public static void main(String[] args) {
        // Set up variables
        double subtotal;
        double tipAmount;
        double tipTotal;
        double total;
        Scanner input = new Scanner(System.in);

        // Read subtotal from console
        System.out.print("Enter the subtotal: ");
        subtotal = input.nextDouble();

        if (subtotal < 0) {
            System.out.println("Sorry, that's a negative total");
        } else if (subtotal == 0) {
            System.out.print("Sorry, that total is 0");
        } else {
            // Read tipAmount from console
            System.out.print("Enter the tip percentage (ex. 20.0): ");
            tipAmount = input.nextDouble();

            // Calculate tip total
            tipTotal = subtotal * (tipAmount / 100 );
            System.out.println("Tip total: " + tipTotal);

            // Add tip total to food total
            total = subtotal + tipTotal;

            total = (int) (total * 100)/100.0;
            // Print out result
            System.out.println("Total bill: " + total);
        }
    }
}
Nested if statements

- If statements can be nested within each other
If-statement gotchas

• Forgetting braces
• Semicolon on the if line
• Rounding errors
• One = sign with boolean expression
public class RoundingError {

    public static void main(String[] args) {
        double x = 1.0 - 0.1 - 0.1 - 0.1 - 0.1 - 0.1;
        // System.out.println(x);
        System.out.println(x == 0.5);
    }
}

public class RoundingError {

    public static void main(String[] args) {
        double x = 1.0 - 0.1 - 0.1 - 0.1 - 0.1 - 0.1;
        System.out.println(x);
        System.out.println(Math.abs(x - 0.5) < .01);
    }
}

Random numbers

• Very important! We use random numbers all the time.

• Math.random()

• Let’s check the documentation! -> https://docs.oracle.com/javase/7/docs/api/java/lang/Math.html

• Math is a static class, so we can just use it. Don’t have to make an instance (like we do with Scanner)
Math.random()  

- Returns a double >= 0.0 and < 1.0  
- **Note!** this is *inclusive* on the lower bound and *exclusive* on the upper bound, meaning you could get 0.0, but never 1.0
Math.random()

(int)(Math.random() * 10)

double between 0.0 and 1.0

upper-bound exclusive (so highest will be 9)

Cast to an int (turn it into one)
Random in range (inclusive)

```java
int range = (max - min) + 1;
int randomInRange = min + (int)(Math.random() * range);
```

```java
int range = (5 - 1) + 1;
int randomNum = 1 + (int)(Math.random() * 5);
```
Logical Operators

• When you’re using an if statement and want to check more than one condition

• ex. if the number < 5 and the number > 2
• ! not
• && and
• || or
• ^ exclusive or (one or the other, not both)
Logical Operators

expression [operator] expression

```java
if( wage < 25000 || children > 4){
}
```

We only compare 2 things at a time in Java
Example: Fizz Buzz Lite

• Classic programming problem

• Ask for a number

• Rules:
  • If the number is divisible by 3, print “Fizz”.
  • If the number is divisible by 5, print “Buzz”.
  • If the number is divisible by both, print "Fizz Buzz”.
  • Otherwise, just print the number.
import java.util.*;

public class FizzBuzzLite {

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

        System.out.println("Enter a number: ");
        inputNumber = input.nextInt();

        if (inputNumber % 3 == 0 && inputNumber % 5 != 0){
            System.out.println("Fizz");
        } else if (inputNumber % 5 == 0 && inputNumber % 3 != 0){
            System.out.println("Buzz");
        } else if (inputNumber % 3 == 0 && inputNumber % 5 == 0){
            System.out.println("Fizz Buzz");
        } else {
            System.out.println(inputNumber);
        }
    }
}

}
Switch statement

• Useful for replacing nested if statements

• Syntax:

```java
switch (switch-expression){
    case value1:
        statement(s)1;
        break;
    case value2:
        statement(s)2;
        break;
    case value3:
        statement(s)3;
        break;
    default:
        statement(s);
        break;
}
```
Switch statement

• Switch expression must yield a `char`, `byte`, `short`, `int`, or `String` type
import java.util.*;
public class SwitchExample {

    public static void main(String[] args) {
        int month;
        Scanner input = new Scanner(System.in);
        String monthString;

        System.out.print("enter the number for the month (ex. 11 for Nov): ");
        month = input.nextInt();

        switch (month) {
            case 1:  monthString = "January";  break;
            case 2:  monthString = "February";  break;
            case 3:  monthString = "March";     break;
            case 4:  monthString = "April";     break;
            case 5:  monthString = "May";       break;
            case 6:  monthString = "June";      break;
            case 7:  monthString = "July";      break;
            case 8:  monthString = "August";    break;
            case 9:  monthString = "September"; break;
            case 10: monthString = "October";   break;
            case 11: monthString = "November";  break;
            case 12: monthString = "December";  break;
            default: monthString = "Invalid month"; break;
        }

        System.out.println(monthString);
    }
}
Switch statement (fall-through behavior)

```java
import java.util.*;
public class SwitchExample {

    public static void main(String[] args) {
        int month;
        Scanner input = new Scanner(System.in);
        String monthString;

        System.out.print("enter the number for the month (ex. 11 for Nov): ");
        month = input.nextInt();

        System.out.println("remaining months of the year:");
        switch (month) {
            case 1:  monthString = "January";
                     System.out.println(monthString);
            case 2:  monthString = "February";
                     System.out.println(monthString);
            case 3:  monthString = "March";
                     System.out.println(monthString);
            case 4:  monthString = "April";
                     System.out.println(monthString);
            case 5:  monthString = "May";
                     System.out.println(monthString);
            case 6:  monthString = "June";
                     System.out.println(monthString);
            case 7:  monthString = "July";
                     System.out.println(monthString);
            case 8:  monthString = "August";
                     System.out.println(monthString);
            case 9:  monthString = "September";
                     System.out.println(monthString);
            case 10: monthString = "October";
                     System.out.println(monthString);
            case 11: monthString = "November";
                     System.out.println(monthString);
            case 12: monthString = "December";
                     System.out.println(monthString);
            break;
            default: monthString = "Invalid month";
                    break;
        }
    }
}```
**Challenge!**

- Write a program that asks the user for a year, then prints out the emoji for the Chinese Zodiac of that year.
  
- For example, if the user entered `2017` the result would be:
  
  - “🐓 year of the rooster”


  - Start with monkey