Chapter 2: Objects and Primitive Data
Lab Exercises

Lab Exercises

Two Meanings of Plus 2
A Table of Student Grades 4

Area and Circumference of a Circle 5
Painting a Room 6

Computing Distance 7
Rolling Dice 8

Drawing a Face 9

Two Meanings of Plus

In Java, the symbol + can be used to add numbers *or* to concatenate strings. This exercise illustrates both uses.

When using a string literal (a sequence of characters enclosed in double quotation marks) in Java the complete string must fit on one line. The following is NOT legal (it would result in a compile-time error).

```
System.out.println ("It is NOT okay to go to the next line
                    in a LONG string!!!");
```

The solution is to break the long string up into two shorter strings that are joined using the *concatenation* operator (which is the + symbol). This is discussed in Section 2.2 in the text. So the following would be legal

```
System.out.println ("It is OKAY to break a long string into " +
                    "parts and join them with a + symbol.");
```

So, when working with strings the + symbol means to concatenate the strings (join them). BUT, when working with numbers the + means what it has always meant—add!

1. **Observing the Behavior of +** To see the behavior of + in different settings do the following:
   a. Study the program below, which is in file `PlusTest.java`.

```
// *******************************************************************
//   PlusTest.java
//
//   Demonstrate the different behaviors of the + operator
// *******************************************************************

public class PlusTest
{
    // -------------------------------------------------
    // main prints some expressions using the + operator
    // -------------------------------------------------
    public static void main (String[] args)
    {
      System.out.println ("This is a long string that is the " +
                          "concatenation of two shorter strings.");

      System.out.println ("The first computer was invented about" + 55 +
                          "years ago.");

      System.out.println ("8 plus 5 is " + 8 + 5);

      System.out.println ("8 plus 5 is " + (8 + 5));

      System.out.println (8 + 5 + " equals 8 plus 5.");
    }
}
```

   b. Open `PlusTest.java` in your `\Chapter 2` folder.

c. Compile and run the program. For each of the last three output statements (the ones dealing with 8 plus 5) write down what was printed. Next to each of those three output statements, select one of the reasons below why it printed what it did and write the corresponding letter. the following rules are used for +. Write out complete explanations.

   A. If both operands are numbers + is treated as ordinary addition. (NOTE: in the expression a + b the a and b are called the operands.)
   B. If at least one operand is a string the other operand is converted to a string and + is the concatenation operator.
   C. If an expression contains more than one operation expressions inside parentheses are evaluated first. If there are no parentheses the expression is evaluated left to right.

```
System.out.println ("8 plus 5 is " + 8 + 5);

System.out.println ("8 plus 5 is " + (8 + 5));

System.out.println (8 + 5 + " equals 8 plus 5.");
```

d. The statement about when the computer was invented is too scrunched up. How should that be fixed? Fix it in your code!

A Table of Student Grades

Write a Java program called `StudentGrades.java` that prints a table with a list of at least 5 students together with their grades earned (lab points, bonus points, and the total) in the format below.

```
///////////////////\\\\\\\\\\\\\\\\\\\
==          Student Points          ==
\\\\\\\\\\\\\\\\\\\///////////////////

Name            Lab     Bonus   Total
----            ---     -----   -----
Joe             43      7       50
William         50      8       58
Mary Sue        39      10      49
```

The requirements for the program are as follows:

1. Print the border on the top as illustrated (using the slash and backslash characters).
2. Use *tab character* (an escape sequence `\t`) to get your columns aligned and you must use the + operator both for addition and string concatenation. (i.e., use addition to calculate the `Total` column using `Lab` + `Bonus` columns.)
3. Make up your own student names and points—the ones shown are just for illustration purposes. You need 5 names.

Area and Circumference of a Circle

Class `Circle.java` describes a circle with a given radius. The radius has the type `double`, which is a primitive data type used for representing real numbers. The `CircleTest.java` class is the driver program that prompts the user to enter a number for the radius, creates a `Circle` object of that radius and displays its area by callig the `Circle's getArea()` method.

Open the file, `Circle.java`, in your `\Chapter 2` folder and modify it as follows:

1. Copy and paste the `getArea()` method. Name the new method `getCircumference().`
2. Modify the code in the `getCircumference()` method to calculate and return the circumference of the circle object.
3. Compile the `Circle.java` program and fix any syntax errors.

Open the file, `CircleTest.java`, in your `\Chapter 2` folder and modify it as follows:

1. Look at the statement which creates a `double` variable named `area`. This statement creates the variable and assigns it the value returned when calling (invoking) the `circle1` objects `getArea()` method. Note the name of this `Circle` object is `circle1`.
2. Add a statement to create a `double` variable named `circumference`. Call (invoke) the `circle1` object's `getCircumference()` method.
3. Add a statement to print out `circle1's` circumference.
4. Copy and paste the correct statements to prompt the user for another radius, create a second `Circle` object named `circle2` and print `circle2's` area and circumference.
5. Modify the `println` statements to indicate which `Circle` object is being printed, either `circle1` or `circle2`.

Painting a Room

File `Paint.java` contains the partial program below, which when complete will calculate the amount of paint needed to paint the walls of a room of the given length and width. It assumes that the paint covers 350 square feet per gallon.

```
//****************************************************************
//File: Paint.java
//
//Purpose: Determine how much paint is needed to paint the walls
//of a room given its length, width, and height
//****************************************************************
import java.util.Scanner;

public class Paint
{
    public static void main(String[] args)
    {
        final int COVERAGE = 350;  //paint covers 350 sq ft/gal
        //declare integers length, width, and height;
        //declare double totalSqFt;
        //declare double paintNeeded;
        //declare and initialize Scanner object

        //Prompt for and read in the length of the room

        //Prompt for and read in the width of the room

        //Prompt for and read in the height of the room

        //Compute the total square feet to be painted--think
        //about the dimensions of each wall

        //Compute the amount of paint needed

        //Print the length, width, and height of the room and the
        //number of gallons of paint needed.
    }
}
```

Open the `Paint.java` file in the `\Chapter 2` directory and do the following:

1. Fill in the missing statements (the comments tell you where to fill in) so that the program does what it is supposed to. Assume that your are only painting the walls, not the ceiling or floor. Compile and run the program and correct any errors.

   Test your program with a length 12 feet, width 16 feet and height 10 feet. *(1.6 gallons needed is answer)*

2. Suppose the room has doors and windows that don't need painting. Ask the user to enter the number of doors and number of windows in the room, and adjust the total square feet to be painted accordingly. Assume that each door is 20 square feet and each window is 15 square feet.

   Test your program with the dimensions above plus 2 windows and 1 door. *(1.457 gallons needed is answer)*

Computing Distance

Write a complete Java program `Distance.java` to compute the distance between two points. Recall that the distance between the two points (x1, y1) and (x2, y2) is computed by taking the square root of the quantity $(x1 – x2)^2 + (y1 – y2)^2$.

Your program should prompt the user for two points as decimal values and read in those values. Create a variable to hold the distance. Compute the distance between the two points and store in your distance variable.

Print out an appropriately labeled statement, including the two points entered and the computed distance.

Test your program using the following data: The distance between the points (3, 17) and (8, 10) is 8.6023... (lots more digits printed); the distance between (–33, 49) and (–9, –15) is 68.352.…

Rolling Dice

Write a complete Java program, `Dice.java,` that simulates the rolling of a pair of dice. For each die in the pair, the program should generate a random number between 1 and 6 (inclusive). It should print out the result of the roll for each die and the total roll (the sum of the two dice), all appropriately labeled. You must use the `random( )` method of the `Math` class. Look at your Chapter 2 notes for an example on how to use `Math.random( ).`

Drawing a Face

Write an applet that draws a smiling face. Give the face eyes with pupils, ears, a nose, and a mouth. Use at least three different colors, and fill in some of the features. Name this file `Face.java`. View your applet using the applet viewer.

Use appropriate and true colors for your face, hair and eyes. Google RGB skin tone values for help.

HINT: Look at the `Snowman.java` file for help on creating your face.

Create the following html file and put in the `\Chapter 2` folder.