CSGE601021 Dasar-Dasar Pemrograman 2
(Foundations of Programming 2)
Lab 01: Fundamental Data Types and Scanner

Build a Java project for this lab, using your favorite IDE.

Part 1. There are four primitive types for integers: `int`, `byte`, `short`, and `long`. Associated with each primitive type is a class with a similar name, so the primitive type `int` is associated with the class `Integer`, the primitive type `byte` is associated with the class `Byte`, `short` is associated with the class `Short`, and `long` is associated with the class `Long`.

These associated classes are sometimes called "wrapper classes". The wrapper classes for integer types contain constants that represent the maximum and minimum values that an integer of the given type can take on. For example, in the wrapper class `Integer`, the constant `Integer.MAX_VALUE` represents the maximum integer value that can be represented with an int variable. Its value is 2147483647.

In the Java API (Application Programming Interface) documentation, you can find the wrapper classes in the package `java.lang`.

The program listed below prints the minimum and maximum values for the `int` primitive type. Modify the program below so that it also prints the minimum and maximum values for `byte`, `short`, and `long`.

```
public class IntegerDemo
{
   public static void main(String[] args)
   {
      System.out.println("Maximum int value:  " + Integer.MAX_VALUE);
      System.out.println("Minimum int value:  " + Integer.MIN_VALUE);

      //Add statements to print the minimum and maximum values for
      //byte, short and long.
   }
}
```

Part 2. We often need to convert a value from a `String` to a primitive data type. For example, the program below converts the `String` `"12345"` to an `int` using the static method `parseInt` that is contained in the wrapper class `Integer`. Remember that a static method can be executed without applying it to an object.

Here is a list of parsing methods in wrapper classes:

```
// These two methods are in the Byte class
public static byte parseByte(String s)
public static byte parseByte(String s, int radix)
// These two methods are in the Short class
public static short parseShort(String s)
public static short parseShort(String s, int radix)
// These two methods are in the Integer class
public static int parseInt(String s)
public static int parseInt(String s, int radix)
// These two methods are in the Long class
public static long parseLong(String s)
public static long parseLong(String s, int radix)
// These two methods are in the Float class
public static float parseFloat(String s)
public static float parseFloat(String s, int radix)
// These two methods are in the Double class
public static double parseDouble(String s)
public static double parseDouble(String s, int radix)
```

Complete the program below by filling in the code specified in the embedded comments.

```
public class StringConversion
{
   public static void main(String[] args)
   {
      String value1 = "12345";
      int intValue = Integer.parseInt(value1);
      System.out.println("intValue = " + intValue);

      String value2 = "12.345";
      // Convert value2 to a double here
      // Print the converted value

      String value3 = "9876543210";
      // Convert value3 to a long here
      // Print the converted value

      String value4 = "321";
      // Convert value4 to a short here
      // Print the converted value

      String value5 = "-28";
      // Convert value5 to a byte here
      // Print the converted value

      String value6 = "-45.237";
      // Convert value6 to a float here
```

```
        // Print the converted value


        String value7 = "3BABE";
        //Assume that value7 contains a hexadecimal number. Convert it
        //    to an int here
        //Print the converted value

        String value8 = "11001";
        //Assume that value8 contains a binary number. Convert it
        //    to an int here
        //Print the converted value

    }
}
```

Part 3. Study and run the program below, which uses both variables and constants:

```
//************************************************************
//  Circle.java
//
//  Print the area of a circle with two different radii
//************************************************************


public class Circle
{
    public static void main(String[] args)
    {
     final double PI = Math.PI; //3.141592653589793

     double radius = 10.5;
     double area = PI * radius * radius;

     System.out.println("The area of a circle with radius "
        + radius + " is " + area);

     radius = 20;
     area = PI * radius * radius;

     System.out.println("The area of a circle with radius "
        + radius + " is " + area);

    }
}
```

Some things to notice:

- The first three lines inside *main* are declarations for PI, radius, and area. Note that the type for each is given in these lines: *final double* for PI, since it is a floating point constant; *double* for radius, since it can contain a fraction, and *double* for area, since it will hold the product of the radius and PI, resulting in a floating point value.
- These first three lines also hold initializations for PI, radius, and area. These could have been done separately, but it is often convenient to assign an initial value when a variable is declared.
- The next line is simply a print statement that shows the area for a circle of a given radius.
- The next line is an assignment statement, giving variable radius the value 20. Note that this is not a declaration, so the *double* that was in the previous radius line does not appear here. The same memory location that used to hold the value 10.5 now holds the value 20—we are not setting up a new memory location.
- Similar for the next line—no *double* because area was already declared.
- The final print statement prints the newly computed area of the circle with the new radius.

Modify the program as follows:

1. The circumference of a circle is two times the product of PI and the radius. Add statements to this program so that it computes the circumference in addition to the area for both circles. You will need to do the following:
   - Declare a new variable to store the circumference.
   - Store the circumference in that variable each time you compute it.
   - Add two additional print statements to print your results.

Be sure your results are clearly labeled.

2. We want the program to read in values for the radius from the user using the keyboard instead of having them written into the program ("hardcoded"). Modify your program as follows:
   - Add the line

     ```
     import java.util.Scanner;
     ```

     This tells the compiler that you will be using methods from the Scanner class. In the main method create a Scanner object called *scan* to read from **System.in**.
   - Instead of initializing the radius in the declaration, just declare it without giving it a value. Now add two statements to read in the radius from the user:
     - A *prompt* statement, that is, a print statement that tells the user what they are supposed to do (e.g., "Please enter a value for the radius.");
     - A read statement that actually reads in the value. Since we are assuming that the radius is a floating point value, this will use the nextDouble() method of the Scanner class.
   - Repeat to get the second value for the radius.

Through the link at SCeLE, submit all your project files (1 project folder), zipped into a file: lab01_<class>_<TACode>_<YourName>_<YourNPM>.zip

Selamat Mengerjakan!

'Met Ngoding! ☺

L.Y.Stefanus & the Asdos Team