ComS 207: Programming I
Midterm 1, SAMPLE SOLUTIONS

Student Name: Mr. Perfect
Student ID Number: 555-55-555
Recitation Section: 1

1. **True/False Questions (10 x 1p each = 10p)**

   (a) This is a valid identifier in Java: 207Rocks — FALSE

   (b) This is a valid identifier in Java: C$_207_Rocks — TRUE

   (c) A constructor method must have the same name as its class. — TRUE

   (d) A class can have only one constructor method. — FALSE

   (e) Unicode characters are a subset of the ASCII characters. — FALSE

   (f) A constructor must be declared for each class. — FALSE

   (g) A constructor must have a void return type. — FALSE

   (h) Each object has a state, defined by its variables, and a set of behaviors defined by its methods. — TRUE

   (i) Integers can store a maximum of three values at the same time. — FALSE

   (j) No more than one object can be created from each class. — FALSE

2. **Short Answer Questions (5 x 2p each = 10p)**

(a) What is an enumerated type?

(From p. 135 in the textbook) An Enumerated type establishes all possible values of a variable of that type by listing, or enumerating them. The values are identifiers, and can be anything desired.

(b) What is an object?

In object-oriented programming languages, and object is an instance of a class. In Java, objects can be created using the new operator. The new operator calls the class constructor which takes care of initializing the newly created object.

(c) What it the difference between a class and an object?

(From p. 198 in the textbook) A class is the blueprint of an object. It defines the variables and methods that will be a part of every object that is instantiated from it. But a class reserves no memory space for variables. Each object has its own data space and therefore its own state.

(d) What is autoboxing?

(From p. 141 in the textbook) Autoboxing provides automatic conversions between primitive values and corresponding wrapper objects. Because of autoboxing an int value can be assigned to an Integer object reference variable.

(e) What is the difference between NumberFormat and DecimalFormat?

NumberFormat can be used without instantiating a reference variable of that type. DecimalFormat cannot be used in this way. We must always instantiate an object of that type using the new operator.

3. **Random Numbers (5 x 2p for each sub-problem = 10p)**

(a) Write a snippet of code that produces a random number in the specified range. You can assume that the following line of code occurs somewhere before your snippet of code: Random rand = new Random();

Hint: Use the version of the nextInt method in the Random class which accepts a single integer parameter.

```
- i.    25 to 49

        int iNum = rand.nextInt(25) + 25;


- ii.   -5 to 6

        int iNum = rand.nextInt(12)-5;


- iii.   Generate a random odd integer from 1 to 100

        int iNum = rand.nextInt(50)*2 +1;
```

(b) This is the same as (a) but with floating point numbers. The upper boundary in not inclusive in this case however.

```
- i.     1.5 to  1.7

         double dNum= rand.nextDouble()*0.2 + 1.5;


- ii.   -1.2 to -0.2

         double dNum = rand.nextDouble() - 1.2;
```

4. **Bug Chase (5p + 10p = 15p)**

Find all bugs in the following programs. Circle the location of each bug and write with words what is wrong with the code at that location.

```
(a) (5p)
======= File: Buggy1.java ========
public class Buggy1
{
   public static void main (String[] args) {
       String s= new String("John");
       System.out.println("My name is " + s);
       return;
   }
}
```

```
(b) (10p)
======= File: Account.java ========
public class Account
{
   private String accName;

   Account(String name)   {
       accName = name;
   }

   public void printInfo() {
       System.out.println("Account Name Is: " + accName);
   }
}
========= File: Bank.java ==========
public class Bank
{
   public static void main (String[] s)   {
       Account account= new Account("John Smith");
       account.printInfo();
       return;
   }
}
```

5. **Expressions and Assignments (10 x 2p each = 20p)**

   For each of the following, write down the value that will be stored in result

```
double result;
int num1 = 5, num2 = 12, num3 = 2;
double val1 = 5.0, val2 = 12.0, val3 = 2.0;

(a)  result = num2 / num1;                               2.0

(b)  result = val3 + num2 / num1;                        4.0

(c)  result = num3 + val2 / num1;                        4.4

(d)  result = num2 / num3 / num1;                        1.0

(e)  result = val2 * num1 + num2 / num3;                 66.0

(f)  result = (int) (val2 * (num1 + num2) / num3);     102.0

(g)  result = (val2 * (num1 + num2)) / num3;           102.0

(h)  result = val2 * ((num1 + num2) / num3);            96.0

(i)  result = num1 * num3 * 4 % num2 / val3;             2.0

(j)  result = num1+++ ++num2;                           18.0
```

6. **Programming Projects (TOTAL 65p, but each has a different weight)**

(a) Rectangles (15p)

Write a complete Java program which asks the user to enter the coordinates of the lower left and the upper right corners of a rectangle. In other words, the program must read two pairs of floating point numbers which represent the x and y coordinates of two points: $P_{LL} = (x_1, y_1)$ and $P_{UR} = (x_2, y_2)$. The program must then calculate and print on the screen the following four values associated with the rectangle: 1) the length of the horizontal side; 2) the length of the vertical side; 3) the perimeter; 4) the area.

```
public class Rectangles
{
   public static void main (String[] args)
   {
                double x1, y1, x2, y2;
                Scanner scan = new Scanner(System.in);

                System.out.print("Please enter lower left  x: ");
                x1=scan.nextDouble();
                System.out.print("Please enter lower left  y: ");
                y1=scan.nextDouble();

                System.out.print("Please enter upper right x: ");
                x2=scan.nextDouble();
                System.out.print("Please enter upper right y: ");
                y2=scan.nextDouble();

                // Take absolute values. This will work even if the user
                // entered the corners in the wrong order.
                double hSide= Math.abs(x2 - x1);
                double vSide= Math.abs(y2 - y1);
                double perimeter= hSide*2.0 + vSide*2.0;
                double area = hSide * vSide;

                System.out.println("Horizontal side = " + hSide);
                System.out.println("Vertical   side = " + vSide);
                System.out.println("Perimeter       = " + perimeter);
                System.out.println("Area            = " + area);
        }
}
```

(b) Password Generator (15p)

Write a complete Java program that generates a random password for a computer account. The password must have a length of exactly 5 characters. The first and the fourth characters must be capital letters from the English alphabet (A-Z). The second and the fifth characters must be numbers (0-9). The third character must be a dash (i.e., '-'). Here are two examples of valid passwords: “R2-D2” and “C3-P0”.

Hint: The only difficult part in this program should be figuring out how to print a single character (without printing a new line). To do this you can use the following snippet of code where num is defined as an int:

```
 System.out.printf("%c", 'A' + num);
```

For example, if num=3 the result should be D; if num=0 the result should be A.

```
import java.util.Random;

public class PasswordGenerator
{
   public static void main (String[] args)
   {
                Random rand = new Random();

                // Get the two letters (1st and 4-th caracters)
                int  firstChar = rand.nextInt(26); // 0 to 25
                int fourthChar = rand.nextInt(26); // 0 to 25

                // Get the two numbers (2nd and 5-th caracters)
                int secondChar = rand.nextInt(10); // 0 to 9
                int  fifthChar = rand.nextInt(10); // 0 to 9

                // Print the password
                System.out.print("Your password is: [");
                System.out.printf("%c", 'A'+firstChar);
                System.out.print(secondChar);
                System.out.print("-");
                System.out.printf("%c", 'A'+fourthChar);
                System.out.print(fifthChar);
                System.out.print("]\n");
        }
}
```

(c) Radians to Degrees (15p)

Write a Java program that converts radians to degrees. The program must ask the user to enter a floating-point number (representing the angle in radians). The program must then convert the angle from radians to degrees and output the result. The output must be formatted into degrees, minutes, and seconds (all of these must be printed as integers, i.e., no decimal points).

Sample input: 2.0

Sample output: 114 degrees, 35 minutes, 29 seconds

Hint 1: degrees = (radians / Math.PI) * 180.0;

Hint 2: $114°35'29'' = 114 + 35 * (1/60) + 29 * (1/60) * (1/60) = 114.5913889°$

```
import java.util.Scanner;
public class Radians2Degrees
{
   public static void main (String[] args)
   {
                Scanner scan = new Scanner(System.in);
                System.out.print("Enter an Angle in Radians: ");
                double radians = scan.nextDouble();

                // Calculate the degrees
                double degrees = (radians/Math.PI) * 180.0;
                int deg = (int) Math.floor(degrees);

                // Calcualate the minutes
                double minutes = 60.0 * (degrees - deg);
                int min = (int) minutes;

                // Calculate Seconds
                double seconds = 60.0 * (minutes - min);
                int sec = (int) seconds;

                System.out.print(radians + " radians = ");
                System.out.print(deg + " degrees, ");
                System.out.print(min + " minutes, ");
                System.out.print(sec + " seconds.\n");
        }
}
```

(d) Formatting Names (20p)

Write a Java program that asks the user to enter his three names on a single line. You can assume that the user will always enter his/her names in the following format: "First Middle Last". In other words, the first character typed in will be the first letter of the first name; the names will be separated with a single space; the last character typed in will be the last character of the last name; and finally all names will have only their first letter capitalized. The program must then format and print the name in the following format: "LAST, First M." In other words, the last name must be printed first with all capital letters, followed by a comma, followed by the full first name as it was typed in, followed by the first letter of the middle name, followed by a period.

Hint: Use the methods of the String class to find the positions of the separators in the original string. Then chop that string into three separate pieces and print them in the desired order and format.

```
import java.util.Scanner;

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

                // Reads the names, assuming this format: First Middle Last
                System.out.print("Please enter your three names: ");
                String allNames= scan.nextLine();

                System.out.println("\nDebugging Information:");
                System.out.println("You entered: [" + allNames + "]");

                // Extract the first name
                int firstSpaceIdx = allNames.indexOf(' ');
                System.out.println("First Space is at Index = " + firstSpaceIdx);

                String firstName= allNames.substring(0, firstSpaceIdx);
                System.out.println("First Name = [" + firstName + "]");

                // Extract the Second Name
                int secondSpaceIdx = allNames.indexOf(' ', firstSpaceIdx + 1);
                System.out.println("Second Space is at Index = " + secondSpaceIdx);
                String secondName= allNames.substring(firstSpaceIdx + 1, secondSpaceIdx);
                System.out.println("Second Name = [" + secondName + "]");

                // Extract the third name
                String thirdName = allNames.substring(secondSpaceIdx + 1, allNames.length());
                System.out.println("Third Name = [" + thirdName + "]");

                // Print the name in the format: LAST, First M.
                System.out.println("\nThis is the formatted name:");

                System.out.print(thirdName.toUpperCase());
                System.out.print(", ");
                System.out.print(firstName);
                System.out.print(" ");
                System.out.print(secondName.charAt(0));
                System.out.print(".");
                System.out.print("\n");
        }
}
```