Lab08
Java Branching Statements

Submission Details

In this lab, you are required to write **TWO** Java programs to solve the given problems shown in the section "Lab Exercises". To make the program implementation easier, you are suggested to write your program using **eclipse** instead of doing it directly using paper & pencil. After you have completed the implementation using eclipse, submit your program files (i.e. LeapYear.java and PaperScissorsRock.java) to the assignment page of Canvas. The deadline would be **ONE** week after your lab section is conducted. You are reminded to double-check your solution to verify everything in correct before submission.

Note: You are required to put your name, student ID and lab section at the beginning of your source file. Also, please DON'T ZIP your files for submission. Instead, you should upload all .java files to Canvas.

Note: Only answer of selected question will be graded.

Objective

The objective of this lab is to reinforce the programming concepts (e.g. basic Java input / output, variable declarations, use of operators and ***branching statements***) taught in the last few lectures.

Software Required

The following software is required for this lab.

1. Java Development Kit, eg JavaSE 1.8
2. An Integrated Development Environment, e.g. eclipse

Introduction

In this lab, you will again practice:

1. Forming a program skeleton by declaring a class with a main method / operation
2. Declaring variables to store values in Java programs
3. Obtaining inputs from keyboard using methods / operations in the Scanner class
4. Solving problems by performing arithmetic operations (+, -, *, /, %) and performing checking and selection using ***branching statements***
5. Displaying the solution of problems on screen using methods / operations in the System class

Background

1. **Form a program skeleton by defining a class with a main method / operation**

   - **Syntax:**

```
// The filename of the Java source file should be exactly the same
// as the <class name>
public class <class name>
{
    // Define a main method / operation, which is the starting point of the program
    public static void main(String[] args)
    {
       // Program statements here
    }
}
```

2. **Declare variables for storing values**

   - **Syntax:**

```
// Declare a variable with name <name of variable> in type <type>
<type> <name of variable>;
// Declare a variable with name <name of variable> in type <type> with
// initial value <initial value>
<type> <name of variable> = <initial value>;
// Declare a number of variables with name <variable name1>,
// <variable name2>, … all in type <type>
<type> <variable name1>, <variable name2>, …;
// Declare a number of variables with name <variable name1>,
// <variable name2>, … all in type <type> with initial value <initial value1>,
// <initial value2>, …, respectively
<type> <variable name1> = <initial value1>, <variable name2> = <initial value2>, …;
where <type> could be in one of the following types:
byte, short, int, long, float, double, char, boolean
```

3. **Obtain inputs from keyboard and store them in variables:**

   Syntax:

```
// Get the class definition of the Scanner class from java.util package
import java.util.Scanner;
// Create a Scanner object with name <object name>
// System.in is the “standard” input stream, which is keyboard
Scanner <object name> = new Scanner(System.in);
// Use the Scanner class’s operations (methods) to read input
<variable name> = <object name>.<method name>();
```

4. **Perform calculation using arithmetic operations:**

- **Syntax:**

```
// Add the two values stored in variable 1 and variable 2
<name of variable for the result> = <name of variable 1> + <name of variable 2>;
// Subtract the value stored in variable 2 from variable 1
<name of variable for the result> = <name of variable 1> - <name of variable 2>;
// Multiply the value stored in variable 1 by the value stored in variable 2
<name of variable for the result> = <name of variable 1> * <name of variable 2>;
// Divide the value stored in variable 1 by the value stored in variable 2
<name of variable for the result> = <name of variable 1> / <name of variable 2>;
// Find the remainder of value stored in variable 1 divided by the value stored in
// variable 2
<name of variable for the result> = <name of variable 1> % <name of variable 2>;
```

5. **Do checking and selection using branching statements:**

- **Syntax:**

```
// If <boolean expression> is evaluated to true, <statement> is executed.
// Otherwise, just skip it.
if(<boolean expression>)
   <statement>;

// If <boolean expression> is evaluated to true, <statement 1>, <statement 2>,
// <statement 3>, …,    are executed. Otherwise, just skip them.
if(<boolean expression>)
{
   <statement 1>;
   <statement 2>;
   <statement 3>;
   // …
}

// If <boolean expression> is evaluated to true, <statement A> is executed.
// Otherwise, <statement B> is executed.
if(<boolean expression>)
   <statement A>;
else
   <statement B>;
```

```
// If <boolean expression> is evaluated to true, <statement A1>, <statement A2>,
// <statement A3>, …,    are executed. Otherwise, <statement B1>,
// <statement B2>, <statement B3>, …, are executed.
if(<boolean expression>)
{
   <statement A1>;
   <statement A2>;
   <statement A3>;
   // …
}
else
{
   <statement B1>;
   <statement B2>;
   <statement B3>;
   // …
}
```

6. Display information on screen:

- **Syntax:**

```
// Print data on screen and the cursor will stay at the end of the string printed
System.out.print(<data> [+ <data>]);
// Print the data on screen and move the cursor to the beginning of the next line
System.out.println(<data> [+ <data>]);
// Print nothing, but move the cursor to the beginning of the next line
System.out.println(“”);
```

Example:

A palindrome is a sequence of characters that reads the same backward as forward. For example, each of the following five-digit integers in a palindrome: 12321, 55555, 45554 and 11611. Write a program that reads a number and determines whether the input number is a palindrome. If the input number is not five digits long, display an error message.

The expected outputs of your program are as follows:

Enter a 5-digit number: 54345
54345 is a palindrome!!!

Enter a 5-digit number: 12345
12345 is not a palindrome!!!!

Enter a 5-digit number: 231321
Input should be 5 digits long.

Java code for the example

```
import java.util.Scanner;

public class Palindrome
{
   public static void main(String[] args)
   {
      Scanner sc = new Scanner(System.in);
      int number;
      int digit1, digit2, digit3, digit4, digit5;

      System.out.print("Enter a 5-digit number: ");
      number = sc.nextInt();

      if(number / 10000 == 0 || number / 100000 >= 1)
        System.out.println("Input should be 5 digits long");
      else
      {
        digit5 = number % 10;
        digit4 = (number/10)%10;
        digit3 = (number/100)%10;
        digit2 = (number/1000)%10;
        digit1 = number/10000;

        if( (digit1 == digit5) && (digit2 == digit4) )
          System.out.println(number + " is a palindrome");
        else
          System.out.println(number + " is not a palindrome");
      }
   }
}
```

Lab Exercises

In this lab, you will be asked to solve **TWO** problems using Java. Please refer to the problem specifications below for the requirements.

Question 1:

Write a Java program "LeapYear.java" that takes the year as input, and report the number of days in that year. The number of days in February is 29 in a leap year and is 28 in a non-leap year. A year **IS LEAP** if the input year satisfies **ONE** of the following conditions:

1. It is divisible by 4 **AND** it is **NOT** divisible by 100
2. It is divisible by 400

Hint: you can use if-else statement.

Other requirement:

You should also check to see if the user input is positive value or not. If not, output an error message.

Expected outputs of your program:

```
Leap Year Calculation
Enter the year: 1900
1900 is NOT a leap year
The number of days in 1900 is 365.

Leap Year Calculation
Enter the year: 2008
2008 is a leap year
The number of days in 2008 is 366.

Leap Year Calculation
Enter the year: -2
Invalid input
```

Question 2:

Write a Java program "PaperScissorsRock.java" to implement the game Paper, Scissors, Rock. First, your program should read in an integer value, 0, 1 or 2 from the user, which represents "paper", "scissors" and "rock" respectively. After the integer value is obtained, your program should output a text that corresponds to the user's choice, i.e.

- If the user input is 0, your program should print "Player picks paper".
- If the user input is 1, your program should print "Player picks scissors".
- If the user input is 2, your program should print "Player picks rock".

Second, your program should randomly generate an integer between 0 to 2, which represents the computer's choice of "paper", "scissors" or "rock". Similarly, your program should output a text that corresponds to the computer's choice, i.e.

- If the randomly generated integer is 0, your program should print "Computer picks paper".
- If the randomly generated integer is 1, your program should print "Computer picks scissors".
- If the randomly generated integer is 2, your program should print "Computer picks rock".

Note:

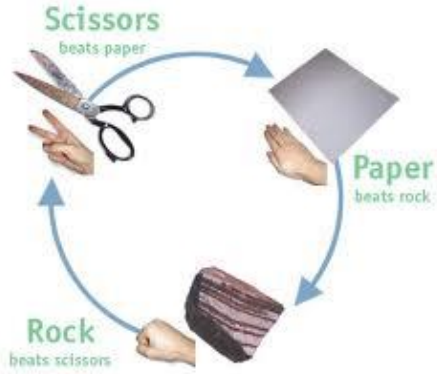
The following example demonstrates how to generate a random value in the range of 0 to 2.

```
import java.util.Random;

public class RandomGenerator
{
   public static void main(String[] args)
   {
      Random randomGen = new Random();
      // nextInt() returns a pseudorandom, uniformly
      // distributed int value between 0 (inclusive)
      // and the specified value (exclusive), therefore,
      // the following generates a random value between 0 to 2
      int val = randomGen.nextInt(3);
      System.out.println("Random value is " + val);
   }
}
```

Finally, your program should determine and output the winner of the game. To do so, the following rules are applied:

1. Rock beats scissors (i.e. 2 beats 1)
2. Scissors beats paper (i.e. 1 beats 0)
3. Paper beats rock (i.e. 0 beats 2)
4. If the choice picked by the user and computer are the same, game draws.



If computer wins, your program should print "Computer Wins".
If user wins, your program should print "Player wins".
If game is drawn, your program should print "Draw".

Other requirement:

You should also check to see if the user's choice is valid or not. If the input is not valid, i.e. the input value is not equal to 0, 1, 2 or -1, output an error message.

Expected outputs of your program:

[Sample output 1]

```
Paper, Scissors, Rock
Enter 0 for paper, 1 for scissors, or 2 for rock (-1 to quit): 1
Player picks scissors
Computer picks paper
Player Wins
```

[Sample output 2]

```
Paper, Scissors, Rock
Enter 0 for paper, 1 for scissors, or 2 for rock (-1 to quit): 1
Player picks scissors
Computer picks rock
Computer Wins
```

[Sample output 3]

```
Paper, Scissors, Rock
Enter 0 for paper, 1 for scissors, or 2 for rock (-1 to quit): 1
Player picks scissors
Computer picks scissors
Draw
```

[Sample output 4]

```
Paper, Scissors, Rock
Enter 0 for paper, 1 for scissors, or 2 for rock (-1 to quit): 10
Invalid input
```

[Sample output 5]

```
Paper, Scissors, Rock
Enter 0 for paper, 1 for scissors, or 2 for rock (-1 to quit): -1
```

Marking Scheme

The marking of this exercise will be based on the following criteria.

Graded items	Weighting
1. Correctness of program (i.e. whether your code is implemented in a way according to the requirements as specified.)	50%
2. Presentation of the Java codes (i.e. whether the program is properly indented, how close you follow the common conventions as mentioned in class, etc.)	30%
3. Documentation (with reasonable amount of comments embedded in the code to enhance the readability)	20%
	100%

Be aware of plagiarism! DON'T copy the program file from your friends or classmates. If any identical copy is found, 5% of the coursework marks will be deducted for each of the file owner.

Program Submission Checklist

Before submitting your work, please check the following items to see you have done a decent job.

Items to be checked	☑ / ☒
1. Did I put my name, student ID and lab section at the beginning of all the source files as comment?	☐
2. Did I put reasonable amount of comments to describe my program?	☐
3. Are they all in .java extension and named according to the specification?	☐
4. Have I checked that all the submitted code are compliable and run without any errors?	☐
5. Did I zip my source files using Winzip / zip provided by Microsoft Windows? Also, did I check the zip file and see if it could be opened? ***(Only applicable if the work has to be submitted in zip format.)***	☐
6. Did I submit my lab assignment to Canvas before the deadline?	☐