Introduction to Java (cs2514): Assignment 1

Playing the Lotto (*Due: February 3rd. Marks: 5*)

Assignment 1: Playing the Lotto (5 *Marks*)

Introduction

For this assignment you will implement a `Ticket` class. Each instance of the class represents a lotto ticket. The `main( )` method, which is defined in the `Ticket` class reads in the numbers on the `Ticket`, creates a `Ticket` instance, and outputs how much prize money the `Ticket` instance is worth.

By the end of this assignment you will know how to:

- Define a simple `java` class;
- Define class methods in the class;
- Call the class methods without the need of any explicit object reference;
- Define a constructor which creates instances of the class;
- Call the constructor;
- Define instance methods;
- Call the instance methods using an explicit or implicit `Ticket` object reference;
- Define a `main()` (class) method;
- Compile the class with the `javac` compiler; and
- Execute the `main( )` method using the `java` virtual machine.

Main Details

The following are the details of the assignment.

- There should be a (single) constructor. The constructor should take in an `int` array which stores the numbers on the lotto ticket.
- For simplicity we assume the winning numbers are fixed. For this assignment the winning numbers are 1, 4, 6, 7, 21, and 30. For simplicity we don't have a bonus number.
- There should be one instance method `printNumbers( )` for printing the numbers of the (current) `Ticket` instance.
- There should be an instance method `printPrizeMoney( )` for printing the prize money of the (current) `Ticket` instance.
- For simplicity, the following are the rules for determining the prize money:

  6 numbers correct 1 000 000 Euro;

  5 numbers correct 30 000 Euro;

  4 number correct 30 Euro;

  3 number correct 3 Euro;
- The `main( )` method should read in the numbers of one single `Ticket` instance using a *class* method called `readTicketNumbers( )`, construct a `Ticket` instance with these numbers, and print out the prize money of the `Ticket` instance.
  - ⋆ The class method `readTicketNumbers( )` should return an `int` array consisting of 6 numbers, each of which should be a unique `int` in the range 1–42.

    - ⋆ The method should read the numbers one at a time.
    - ⋆ When the user enters an invalid `int` the method should keep on reading until the user enters a valid number.
    - ⋆ When the user has entered all numbers, the method should return the `array`.
    - ⋆ *Hint: You can read an `int` from the keyboard by creating a `Scanner` object (please see Lecture 3 for the details), and by calling the `Scanner`'s instance method `nextInt( )`.*
- ○ Please remember to use good object oriented and software engineering standards.
  Please re-read the previous sentence because implementing a class that satisfies the requirements is not the only *objective: the class should have a good, maintainable design.*

The difference between class and instance methods is crucial:

- ○ Inside the `ticket` class you can call the class method `readTicketNumbers( )` by writing (`readTicketNumbers( )` or by writing `Ticket.readTicketNumbers( )`.
- ○ To call an instance method, say `method()`, you need a object reference to an instance of the class that defines the instance method. For eaxmple, assume you have a `scanner` object reference variable which references instance of the `Scanner` class: `Scanner scanner = new Scanner( System.in )`. The `Scanner` class defines the instance method `nextInt( )`. You call this instance method by writing `scanner.nextInt( )`.

1 Restrictions

The following are some restrictions:

- ○ Please do not use packages;
- ○ You are allowed to use the `Scanner` class. Please do not use any other classes.

Submission Details

- ○ Before you submit this assignment, please read the remainder of this section.
- ○ Please use proper `JavaDoc`, which should include a `JavaDoc` comment for the class and `JavaDoc` comments for all public class methods and public instance methods. You should use the `@author` tag in the class `JavaDoc` for your name and ID:

```
/**
 * One-line comment that describes the class.
 * More comments if needed.
 *
 * @author: YOUR NAME (YOUR ID)
 */
```

- ○ Use the CS2514 `Canvas` site to upload your program as a single *.tgz* archive called *Lab-1.tgz* before 23.55pm, February 3rd, 2020. To create the *.tgz* archive, do the following:
    - ⋆ Create a directory `Lab-1` in your working directory.
    - ⋆ Copy `Ticket.java` into the directory. Do not copy any other files into the directory.
    - ⋆ Run the command '`tar cvfz Lab-1.tgz Lab-1`' from your working directory. The option 'v' makes `tar` very chatty: it should tell you exactly what is going into the `.tgz` archive. Make sure you check the `tar` output before submitting your archive.
    - ⋆ Notice that file names in `Unix` are case sensitive and should not contain spaces.
- ○ Notice that the format is `.tgz`: please do *not* submit `zip` files, do *not* submit `gzip` files, do *not* submit `tar` files, do *not* submit `bzip` files, and do *not* submit `rar` files. If you do, it may not be possible to unzip your assignment.
- ○ Marks are deducted for poor choice of variable names and/or poor layout.
- ○ No marks shall be awarded for programs that do not compile.