**Institute of Technology Tralee**

**Computing Department**

**Object Oriented Programming 1**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Session 5 – Review of some of this years CAs on methods**

The game of “roll the dice” is to be created which pits you against the computer. In this game a pair of dice are “rolled” for both the computer and you, the player. When a pair of dice are rolled the sum of the values on each face of the dice will give a number between 2 and 12 inclusive. You will be using the **Math** class’ random() method to generate these dice rolls in a random fashion. Note that this method is static. If the sum of the dice roll for the computer exceeds that for the player then the computer wins the game, if the sum of the dice roll for the player exceeds that for the computer then the player wins the game, otherwise the game ends in a draw.

Your main() will contain a data-sentinel controlled **while** loop that will allow the user to play as many games of “roll the dice” as they wish. At the end of each game, the user will be asked whether they wish to play another game. If they say “no” then the loop finishes and the application will give the user a farewell message thanking them for playing the game. If the user says “yes” then the loop iterates again and another game of “roll the dice” will take place.

After each game is played, a message dialog will appear which tells the user how many games have been played at that point as well as how many games both the computer and the user have won, and how many games have been drawn.

**rollTheDice**() will be a user-defined method that will take **no arguments** at all and will use the random() method twice within it to generate the dice roll for the computer first of all and then the player – these values will be displayed in turn on message dialogs. The dice roll number generated for the computer will then be compared with that for the player and the method will simply return the winner of the game as a **char** as follows: ‘c’ for the computer, ‘p’ for the player or ‘d’ for a draw (when both rolls generate the same number).

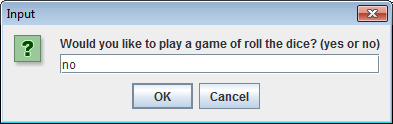
Back in the main(), when the rollTheDice() method returns its outcome, there will be code which keeps track of the number of games played, as well as the number of wins recorded for both the computer and player, along with the number of draws.

For full marks here your program should, along with a logically correct solution for the problem above, include the usual **single-line** and **multi-line comment** at the top of the program. The multi-line comment should briefly explain the purpose of the program.

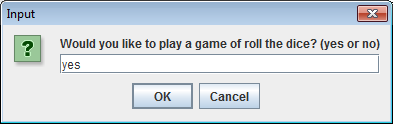
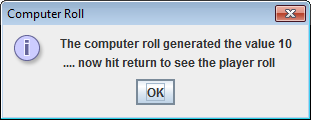
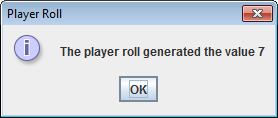
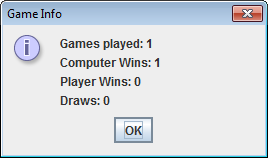
Your program should run as indicated in the following sample screenshots. Of course, as random number generation is used in this program, your output will not match that in my own screenshots below exactly.

**Sample Screenshots**

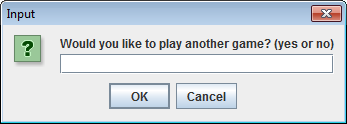
**Run 1 – user decides they don’t want to play a game at all here, they just get the farewell message dialog straight away**

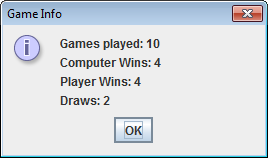
**Run 2 – user decides to play a game, a message dialog appears indicating the value of the computer dice roll, then another indicating the value of the player dice roll, then a final one indicating the current game statistics**

**Once the statistics dialog disappears, an input dialog appears again to ask the user if they want to play another game:**



**The process then repeats until the user eventually decides they’ve played enough games by entering “no” on this dialog, at which point the “farewell” dialog appears. The message dialog below shows what the games statistics looked like in my case after playing the game 10 times – yours would be different to this of course, depending on the random numbers generated.**



**(b)**

The hexadecimal number system uses the digit characters 0-9 inclusive as well as the letter characters A-F inclusive so, for example, 1204ABC and FD956DB are both valid hexadecimal numbers. For simplicity in this exam we will take it that **only the capital letters A-F are valid**.

You must write a program where, in your main() method, you will read in exactly 4 String values from the user, one at a time, using a **for** loop, and each of these values will then be passed in turn to a user-defined method called isHexadecimal()

The isHexadecimal() method will take a single String as an argument and will determine whether or not that String value constitutes a valid hexadecimal number. If it turns out that the number is hexadecimal then the method will return the boolean value **true**, otherwise it will return the boolean value **false**. You will need to use a **loop** in this method in order to cycle over all the characters in the String passed to the method. You can use any type of loop you want to for this part.

Once the isHexadecimal() method has returned its result each time, the main() will use this result to keep track of the number of values that were valid hexadecimal numbers and will also keep a list of those values that were valid.

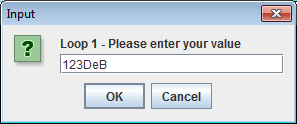
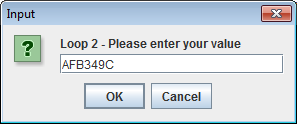
Once the for loop in the main() is completed, the program will finally display a message dialog showing the **percentage** of valid hexadecimal values entered and the list of valid values.

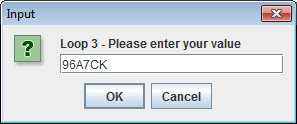
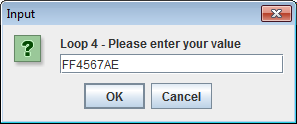
For full marks here your program should, along with a logically correct solution for the problem above, include the usual **single-line** and **multi-line comment** at the top of the program. The multi-line comment should briefly explain the purpose of the program.

Your program should run as indicated in the following sample screenshots. You can use the values indicated when testing out your own program.

**Sample Screenshots**

**The user enters their string values, one by one, and each gets tested by the method. The program keeps track of the number of valid hexadecimal values and also keeps a list of them as the for loop progresses.**

**Once the for loop completes the program results are displayed as follows:**

