ICT373: Lab FOUR

Lab Practice Session 4

1. Open a DOS (command prompt) window and write a small application program in Java to read in a sequence of positive integer numbers from the keyboard (terminate input with a negative number) and display the numbers, 4 numbers per line, followed by their average, a count of how many numbers were greater than the average and a count of how many numbers were smaller than the average. Compile and run this in the command prompt window.

   Now use the NetBeans IDE to create and run a small project with a similar program.

2. Copy the Java example program **`eg1.java`** (see at the bottom of the page) and compile and run it. Then do the following modifications and enhancements:

   a) Separate t3eg1.java into a Fraction class file and a client class file. Make the Fraction class variables numerator and denominator private. Add a default constructor to the Fraction class.

   b) Add an "add" method to the Fraction class to add two fractions together to get a third fraction (recall a/b + c/d = (a*d+b*c)/(b*d)). The method should take the second fraction as a parameter and return the result fraction via the **return** statement. Test it via a client program.

   c) Add a "multiply" method to the Fraction class to multiply two fractions together to get a third fraction. There is no need to simplify the fraction. Test it via a client program.

   d) Add a “divide” method to the Fraction class to divide the first fraction by the second fraction to get a third fraction. Test it via a client program.

   e) Add a private method “simplify” to the Fraction class which converts the fraction to its simplest form. For example, the fraction 20/60 should be stored as 1/3.

   f) Add a “toString” method to the Fraction class which returns the fraction as a String in the form x/y, where x and y are numerator and denominator respectively

   g) Modify the client program to loop around getting pairs of fractions from the user and then displaying the two fractions together with their sum, product and division fractions. The program should loop around until the user does not want to process any more fractions (or until the user enters a zero fraction).

3. Write an application that plays "guess the number" as follows: Your program chooses the number to be guessed by selecting a random integer in the range 1 to 1000. The application displays the prompt Guess a number between 1 and 1000. The player inputs a first guess. If the player's guess is incorrect, your program should display Too high. Try again. or Too low. Try again. to help the player "zero in" on the correct answer. The program should prompt the user for the next guess. When the user enters the correct answer, display Congratulations. You guessed the number!, and allow the user to choose whether to play again. [*Note*: The guessing technique employed in this problem is similar to a *binary search*.]

4. Modify the above program to count the number of guesses the player makes. If the number is 10 or fewer, display "Either you know the secret or you got lucky!" If the player guesses the number in 10 tries, display "Aha! You know the secret!" If the player makes more than 10 guesses, display "You should be able to do better!" Why should it take no more than 10 guesses? Well, with each "good guess", the player should be able to eliminate half of the numbers, then half of the remaining numbers, and so on.

An Example (eg1.java)

```
class Fraction {
        int numerator, denominator;

        Fraction(int a, int b) {
                numerator=a;
                denominator=b;
        }

        void print(){
                System.out.println( numerator + " / " +
                        denominator );
        }
}

public class eg1 {
        public static void main(String[] args) {
                Fraction f = new Fraction(2,3);
                f.print();
        }

}
```