Warm-up with Object Oriented Programm in Java

Goals:

- Write your own Java classes and methods.
- Learn how to get user input and print output properly.
- Understand how to create Java objects for testing.

First of all, I am giving some sample questions and answers to show you what kind of programs you are expected to write and submit. Especially, the example codes illustrate how to get input from users.

Example 1. Write a class that has the following methods:

- `public int sumOfDigits(int n)` – to find the sum of the integer parameter's digits and return the result.
- `public void inputAndProcess(Scanner in)` – to input an integer, call sumOfDigits and display the result.

Hint: % is the division operator that returns the remainder. Try using it to divide by 10.

Answer:

```
import java.util.Scanner;

class Example1
{
  public int sumOfDigits(int n)
  {
    int sum = 0;
    n = Math.abs(n);
    while (n > 0)
    {
      sum += n % 10;
      n /= 10;
    }
    return sum;
  }

  public void inputAndProcess(Scanner in)
  {
    System.out.print("Type an integer: ");
    int n = in.nextInt();
    System.out.print("The sum of the digits of: " + n);
    System.out.println(" is: " + sumOfDigits(n));
  }

  // The main method should do no more than creating the objects
  // and call a method to do the work.
  public static void main(String[] args)
  {
    Scanner in = new Scanner(System.in);
    Example1 myObject = new Example1();
    myObject.inputAndProcess(in);
    in.close();
```

```
  }
}
```

Notes:

- Math.abs. This is a function implemented as a utility method in class Math. The abs function returns the absolute value of a double, basically meaning that it removes the minus sign from any negative number.
- The Scanner class is used to get user input, and it is found in the java.util package. To use it, create an object of it and use any of the available methods found in the Scanner class documentation (https://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html and https://www.w3schools.com/java/java_user_input.asp). In this example, we use nextInt() method to read an integer. You can also use nextLine() to read Strings.

Example 2. Write a one-class program that has a method to ask for and return the name of a text file, a method taking a file name (a String) as a parameter that reads and displays each line of text in the file on the screen, and a method to call the first two methods.

Answer:

```
import java.util.Scanner;
import java.io.*;

class Example2
{
  private void displayFileContent(String filename) throws IOException
  {
        File file=new File(filename);    //creates a new file instance
        FileReader fr=new FileReader(file);     //reads the file
        BufferedReader br=new BufferedReader(fr);     //creates a buffering character
input stream
        String line;
        while((line=br.readLine())!=null)
        {
              System.out.println(line);
        }
        fr.close();    //closes the stream and release the resources
  }


  private String getFileName()
  {
    Scanner in = new Scanner(System.in);
    System.out.print("Enter filename: ");
    String filename = in.nextLine();
    in.close();
    return filename;
  }

  public void showFile() throws IOException
  {
    String filename = getFileName();
```

```
    displayFileContent(filename);
  }

  public static void main(String[] args) throws IOException
  {
    Example2 myObject = new Example2();
    myObject.showFile();
  }
}
```

Note:

- The File, FileReader and BufferedReader classes are used to read line by line from a text file. They can be found in the java.io package. To learn how to use them, check the java documentation and (https://docs.oracle.com/javase/7/docs/api/java/io/File.html, https://docs.oracle.com/javase/8/docs/api/java/io/BufferedReader.html, https://docs.oracle.com/javase/7/docs/api/java/io/FileReader.html, and https://www.javatpoint.com/how-to-read-file-line-by-line-in-java). Look at the way that data is read from the file in the while loop. The readLine() method will return NULL when there is no more line to read from the file.
- Note that the methods displayFileContent and getFileName have been made private. The public showFile method, acting as a controller, is the only one that can be called for an object in the main method.

Questions:

Question1: Write a one-class program that has the following methods:

- a method that inputs and returns a double value,
- a method that takes two double parameters, adds them together, and returns the square root of the result,
- a main method to create an object and call the other two methods, displaying the result of calling the second method.

Question2: Write methods to do the following:

- Convert from millimeters to feet.
- Convert from meters to inches.
- Convert from kilometers to yards.

Include the methods in an interactive program that lets the user select which conversion to perform, and then inputs a value and displays the result.

An interactive program means one that displays a simple text menu, such as

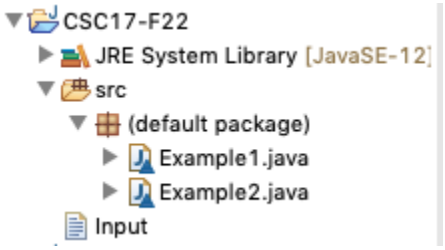
1. Convert millimeters to feet.
2. Convert meters to inches.
3. Convert kilometers to yards.
4. Quit

and then asks the user to type in a value from 1 to 4 to select which action to take. A loop will continually redisplay the menu until the user selects Quit (number 4) to terminate the program.

Notes: The conversion methods should take the value to be converted as a parameter and return a result. They should not do any input or display the result. Add additional methods if they will help structure the program.

Question3: Write a one-class program with suitable methods to read a text file and count the numbers of lines and English letters.

Hints: You will need to use the File, FileReader, and BufferedReader classes. See example2. The Read() method of BufferedReader can be used to read file character by character. The file you are reading should be in the same directory as the program you are working on. For example, the following screenshot shows that in a Java project created in Eclipse, the file "input" is placed in the folder where the java source codes are.



Question4: Write a one-class program with suitable methods to read an integer typed at the keyboard and display the nearest prime number. For example, if the input is 10, then 7 should be displayed.

Note: Search the web for information about prime numbers and algorithms for finding them - there are some excellent websites.

Submission:

You would need to upload one java file for each question. They should be named Question1.java, Question2.java, Question3.java, and Question4.java. Along with the java files, you are also required to submit a **screenshot** of the testing output for each question in Eclipse. Make sure you test your program before submission to find errors and get the full credits. A clearly-commented program is highly appreciated.