Java 8 Homework

Study the code that I gave you!!

Look at the Stream interface, and the Collectors (good code examples) class in the Oracle Java API.

1. Short answer
   1. Name the differences between imperative and functional programming
   2. Explain the meaning of declarative programming. Give an example.
   3. Name the benefits of including functional style programming in Java

2) Use **Lambdas and Streams** to do the following. Hard code all the data that you need and Test it Thoroughly!

**a)** In ‘Program 4’ of the file I gave you, of complete Java 8 programs (At first ignore the **// group Employees by department code** in that program) (Run it) :

1) Count the number of last names that begin with the letter ‘B’. Print out this number.

2) Print out all of the Employee objects whose last name begins with the letter ‘B’ in sorted order.

3) Print out all of the Employee objects whose last name begins with the letter ‘B’ and change their first name and last name to be All capital letters. **Do this in two different ways; the first way does Not change the original list, and the second way does change the original list.**

4) Print out All of the employee objects, but if the last name begins with the letter ‘B’, then capitalize all the letters in the last name.

4.1) Use the Collectors.joining method to print out All Employee objects. (See my presentation file ‘Do\_Last\_This\_Was\_Lesson 9\_Streams\_E.docx’.)

4.2) Use the Collectors.joining method to print out All Employee objects, and separate each one with a delimeter of “---\n---“. (See my presentation file ‘Do\_Last\_This\_Was\_Lesson 9\_Streams\_E.docx’.)

5) Print out all of the Employee objects’ last names, whose last name begins with the letter ‘I’ in sorted order, and get rid of all the duplicates. Print out only the last names.

6) Print out the average of all the salaries.

7) Use the ‘reduce’ method to print out the total salary of all employees.

8) Print out only the first names of all the employees. Use the ‘map’ method to accomplish this.

9) Create an infinite stream of even numbers (0, 2, 4, …) and then, eventually print out only the first 20 even numbers from this stream.

**Ask me for the student solution that I have for the above problems!**

**Keep asking!!**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

3) a) Implement a method with the following signature and return type:

public int countWords(List<String> words, char c, char d, int len)

which counts the number of words in the input list words that have length equal to len, that contain the character c, and that do not contain the character d. Create a solution that uses a lambda expression.

Hint : Look at ALL of the methods that are available in the String class. This will make it easier for you.

b) Use reduce to concatenate the Strings in the Stream below to form a single, space-separated String. Print the result to the console.

public static void main(String[] args) {

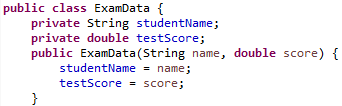
Stream strings = Stream.of("A", "good", "day", "to", "write", "some", "Java");

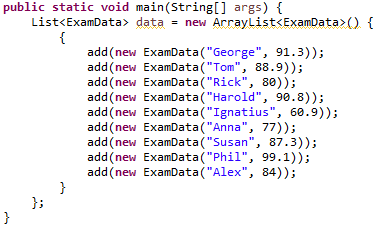
}

c) The many Java standard methods that are in the files I gave you; create programs to use them and make sure the results are correct! (For example, test findfirst, findany, the Optional object (use generics here), orElse inside an Optional, and others …

**Level 3 begins here :**

4) Use DoubleSummaryStatistics to output to the console the top test score, lowest test score, and average among all test scores in a given list.





--------------------------------------------------------------------------------------------------------------

5) Redo the first lab you had, the School lab, using Java 8 streams and lambda expressions.

Do one level at a time.

6) Redo Labs 3, 3.1, and 3.2 using Java 8 streams and lambda expressions.

7) Level 3 : Study ‘Program 4’ of the file I gave you, of complete Java 8 programs.

Understand All of the code. Look at the ‘**// group Employees by department’**

**// collect(Collectors.groupingBy**

code at the bottom, and understand what it is doing so that you can explain it to me or to

another student. Run the code. Then try to do the following :

a) Print out each department and the average salary for the department.

b) Print out each department and the maximum salary for the department.

c) Print out each department and all of the employees who work at that department.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Extra Credit 1** : Look at (from the Oracle Java API) some of the methods in the Stream interface and the Collectors class, that we have Not covered in class. Understand what they do. Then use them in a real Java program, and make sure your results are correct. Upload these java files to Sakai.

**Extra Credit 2** : Take some, or many of the Stream methods, like filter, map, etc.,

and write your own version of these methods.

For example, write, ‘myFilter’, ‘myMap’ etc.

Test your new methods thoroughly!