Using Interfaces

Programming Projects:

1) An interface

- Start Eclipse and create a new Java project called ‘lab2’
- Create an interface (not a class) called **Counter**
- Add the following method declarations to the interface –

```
int countWords(String sentence);
int countLetters(String sentence);
int getLength(String sentence);
```

- Add comments that describe each method’s functionality (you should be able to guess functionality from the names).

2) An implementing class

- Add a class called **WordProcessor**, which ‘implements’ the above **Counter** interface.
- Provide an implementation for each method declared within the interface. Use the following hints –

  a) To count words, just split on spaces.
  b) The length of the sentence can include spaces.
  c) When deciding whether each character is a letter consider using something like –

```
if ( Character.isLetter(sentence.charAt(i)) )
            letters++;
```

- Create a **Driver** class which asks the user to input some text.
- Create an instance of the **WordProcessor** class, then call each of the methods passing in the user’s input.
- In the **Driver** class change the DataType of the **WordProcessor** instance to be **Counter**. Is this allowed? If so why?

3) Adding fall-back text

- Add a **String** type attribute to the **WordProcessor** called ‘text’
- Write getters and setters to allow the text to be accessed.
- Update the implemented methods so that they use this ‘text’ attribute if a **null** value is passed to any of the implemented methods.
- Update the **Driver** class so that it tests the implemented methods by passing in both user input text and **null** string values.
- What effect does this have on the datatype of the **WordProcessor** instance within the **Driver** class?

Think you’re finished? Check you have commented your code and correctly used ‘public’, ‘private’ and @Override