COW – Strings

Level 1

Create a StringPrinter Class With the following methods:

Name: `printOutEachChar`
Input: `String word`
Output: nothing
Action: print out all the letters in the string on separate lines with "Char" followed by the letter number are the start of the line. So the String "Cat" would print out:

```
Char 1: C
Char 2: a
Char 3: t
```

Create a StringBuilder Class With the following methods:

Name: `combineWords`
Input: `String word1, String word2, String word3`
Output: `String combination`
Action: takes in three words and combines them into one String. Each word should be separated by a space (" ") in the combined text. So combineWords("Varun", "came", "here") is returned as "Varun came here".

Create a StringModifier Class With the following methods:

Name: `turnIntoAllCaps`
Input: `String text`
Output: `String allCaps`
Action: returns the text in all capital letters. Hint – there is a very useful String method for this.

Name: `exclaimWord`
Input: `String word`
Output: `String result`
Action: Adds an exclamation mark to the end of the word. So "apple" becomes "apple!" and "dog" becomes "dog!"

Create a StringAnalyzer Class With the following methods:

Name: `countSentences`
Input: `String text`
Output: `int countOfSentences`
Action: counts the number of time a '.', '?', or '!" appears in text.

Level 2

Add the following method to the StringPrinter Class:

Name: `printOutInReverse`
Input: `String word`
Output: nothing
Action: print out all the letters in the string on separate lines in reverse with "Char" followed by the letter number are the start of the line. So the String "Cat" would print out:

```
Char 3: t
Char 2: a
Char 1: C
```

Create a StringBuilder Class With the following methods:

Name: combine`NumTimes`
Input: `String word, int num`
Output: `String repeatedWord`
Action: returns a String with word repeated num times. So combineNumTimes("Cat", 5) would return: "CatCatCatCatCat"

Name: `combineWordsInOrder`
Input: `String word1, String word2`
Output: `String combination`
Action: takes in two words and combines them into one String in alphabetical order. Each word should be separated by a space (" ") in the combined text. So combineWordsInOrder("Ant", "Bear") is returned as "Ant Bear" while combineWordsInOrder("Dog", "Cat") is returned as "Cat Dog". Hint – use compareTo

Add the following method to the StringModifier Class:

Name: `twistWord`
Input: `String word`
Output: `String result`
Action: Takes the second half and places it in the front of the first half. So “marking” becomes “kingmar” and “face” becomes “cefa”.

Add the following method to the StringAnalyzer Class:

Name: `countVowels`
Input: `String word`
Output: `int countOfVowels`
Action: counts the number of vowels in the word passed in. Assume that there can be both upper and lower case vowels.

Level 3

Add the following method to the StringPrinter Class:

Name: `printTwoWord`
Input: `String word`
Output: nothing
Action: takes in a String with two words and prints out each word's letters. It prints out "Word #" followed by all the letters in the word on separate lines with "Char" followed by the letter number are the start of the line. So the String "Cat Hair" would print out:

```
Word #1
Char 1: C
Char 2: a
Char 3: t
Word #2
Char 1: H
Char 2: a
Char 3: i
Char 4: r
```

Create a StringBuilder Class With the following methods:

Name: `hideText`
Input: `String text`
Output: `String hiddenText`
Action: It should return the text passed in but with all consonants replaced by 'X', all vowels replaced by 'O', and all spaces replaced by '+'. So hideText("You totally should eat turkey if you are hungry") should return "XOO+XOXOXXX+XXOOXX+OOX+XOXXOX+OX+XOO+OXO+XOXXXX".

Name: `reverseWord`
Input: `String word`
Output: `String reversedWord`
Action: takes in a word and returns a String that is the reverse of the original word. So "mathematics" is returned as "scitamehtam"

Add the following method to the StringModifier Class:

Name: `respondToAction`
Input: `String text`
Output: `String noun`
Action: Takes in a String in the format "<noun> made <something>". Returns "<something> was created by <noun>". So respondToAction("Mayewsky made a test") would return "a test was created by Mayewsky" and respondToAction("Yaro Mayewsky made dinner") would return "dinner was created by Yaro Mayewsky". You do not need to worry about capitalizing letters. Hint – use a combination of indexOf and substring.

Add the following method to the StringAnalyzer Class:

Name: `countConsonants`
Input: `String word`
Output: `int countOfConsonants`
Action: count the number of consonants in the word passed in. Assume that there can be both upper and lower case consonants. Also assume that there are no special characters. (Hint – there is a supper easy way to program this method)

Level 4

Add the following methods to the StringPrinter Class:

Name: `printOutBirthday`
Input: `String birthday`
Output: nothing
Action: takes in a String that stores a birthday in the format: "month day, year". This method prints out each component on separate lines. Hint – use a combination of indexOf and substring. So "January 13, 1978" would print:

```
Month: January
Day: 13
Year: 1978
```

Create a StringBuilder Class With the following methods:

Name: `getStartingLetters`
Input: `String text`
Output: `String staggeredText`
Action: returns a String that stores the first letter of every word. So getStartingLetters("You can do it and so can I") will returns: "YcdiascI". You may assume that the starting letter of each word is either the first letter or a letter after a space.

Name: `combineWordsInOrder`
Input: `String word1, String word2, String word3`
Output: `String combination`
Action: takes in three words and combines them into one String in alphabetical order. Each word should be separated by a space (" ") in the combined text. So combineWordsInOrder("Cat", "Ant", "Bear") is returned as "Ant Bear Cat"

Add the following methods to the StringModifier Class:

Name: `makeMoreDramatic`
Input: `String text`
Output: `String result`
Action: Takes in a String in the format "I like <subject>." or "I dislike <subject>.". If the String has the word like then the phrase returned is "Do you really like <subject>, or do you love <subject>?". If the String has the word dislike then the phrase "Do you really dislike <subject>, or do you hate <subject>?" is returned.

Add the following method to the StringAnalyzer Class:

Name: `countOccurences`
Input: `String text, String word`
Output: `int count`
Action: counts the number of times the word appears in text. Words within words should count and capital letters should be ignored. So countOccurences("I am Sam and I like spam. Am I dammed?", "am") returns 5.

Level 5

Add the following methods to the StringPrinter Class:

Name: `printOutAddress`
Input: `String address`
Output: nothing
Action: takes in a String that stores an address in the format: "<Street Number> <Street Name>, <County>, <State Abbreviation> <Zip Code>". This method prints out each component on separate lines. So "211 Baker Street, Sterling, VA 20165" would print:

```
Number:  221
Street Name:  Baker Street
County:  Sterling
State:  VA
ZIP:  20165
```

While "18515 North Crescent Avenue, Prince William, MD 91481" would print:

```
Number:  18515
Street Name:  North Crescent Avenue
County:  Prince William
State:  MD
ZIP:  91481
```

Create a StringBuilder Class With the following methods:

Name: `elongateWord`
Input: `String word`
Output: `String combination`
Action: takes in a word and returns a String with each consonant repeated twice unless is at the beginning or end of the String. Every vowel is repeated four times unless it is adjacent to another vowel in which case it will only be repeated three times. So "cat" is returned as "caaaat", "cart" returns as "caaaarrt", "prediction" returns as "prreeeeddiiiiccttiiiooon", and "aunt" returns as "aaauuunnt".

Modify the following methods in the StringModifier Class (change is in bold):

Name: `convertPhoneNumber`
Input: `String phoneNumber`
Output: `String phoneNumber`
Action: takes in ten digit phone number as a String. It could have a leading one such as "17035555555", parenthesis such as "(703)5555555", or dashes such as "703-555-5555", or some combination of all three. Then it returns it in the format "7035555555".

Name: `pigLatinateWord`
Input: `String word`
Output: `String result`
Action: takes in a word and does the following:

- If a word starts with a vowel then it's first letter is placed at the end with a "hay" added after it. So "orbit" becomes "orbitohay".
- If a word starts with a consonant then the word will have **all the consonants until the first vowel** removed from the front and placed at the back. Then an "ay" added to the end. So "happy" becomes "appyhay" and **"scram" becomes "amscray"**.
- You do not need to worry about capital letters or punctuation

---

Add the following methods to the StringAnalyzer Class:

Name: `countTotalOccurance`
Input: `String text, String [] searchTerms`
Output: `int countOccurances`
Action: count the number of times that any of the searchTerms appear in the text passed in. Note that words within words should count. So if run is in the array searchTerm and the sentence passed in is "He runs quickly" then it should count even though an 's' is attached to run. For Example:
text : "I am so so so great." searchTerms: {"so", "you", "am", "eat"}
returns 5 since "so" appears 3 times, "am" appear once, "eat" appears once, and 3 plus 1 plus 1 adds up to 5.

Level 6

Add the following methods to the StringPrinter Class:

Name: `printWords`
Input: `String text`
Output: `nothing`
Action: print out the words in the string on separate lines. So the String "My code won't compile" would print:

```
Word 1: My
Word 2: code
Word 3: won't
Word 4: compile
```

Create a StringBuilder Class With the following methods:

Name: `repeatAndReverse`
Input: `String word, int n`
Output: `String result`
Action: takes in a word and returns a String with word repeated n number of times. But every other word should be a reverse of the original. Each word should be separated by a space (" ") in the combined text. So repeatAndReverse("Go", 5) returns "Go oG Go oG Go".

Add the following methods to the StringModifier Class:

Name: `pigLatinateText`
Input: `String text`
Output: `String result`
Action: takes in text and pigLatinates each word in the text.
- You do not need to worry about punctuation or capital letters.

Name: `staggerCapitals`
Input: `String text`
Output: `String result`
Action: returns a String with what is stored in text so that every other letter is capitalized. So "I went to the store. I bought milk." returns "I wEnT tO tHe StOrE. i BoUgHt MiLk.". The only non-characters will be a space or period.

Add the following methods to the StringAnalyzer Class:

Name: `findWords`
Input: `String text, String [] searchWords`
Output: `String [] foundWords`
Action: returns an array of all the searchWords that were found in the text. For Example:
text : "I am so so so great." searchTerms: {"so", "you", "am"} returns {"so", "am"}