Eric Roberts
CS 106A

Handout #31
April 30, 2012

Section Handout #4
String Processing

1. Scrabble scoring (Chapter 8, exercise 5, page 286)

In most word games, each letter in a word is scored according to its point value, which is inversely proportional to its frequency in English words. In Scrabble, the points are allocated as follows:

Points	Letters
1	`A`, `E`, `I`, `L`, `N`, `O`, `R`, `S`, `T`, `U`
2	`D`, `G`
3	`B`, `C`, `M`, `P`
4	`F`, `H`, `V`, `W`, `Y`
5	`K`
8	`J`, `X`
10	`Q`, `Z`

For example, the Scrabble word `FARM` is worth 9 points: 4 for the *F*, 1 each for the *A* and the *R*, and 3 for the *M*. Write a `ConsoleProgram` that reads in words and prints out their score in Scrabble, not counting any of the other bonuses that occur in the game. You should ignore any characters other than uppercase letters in computing the score. In particular, lowercase letters are assumed to represent blank tiles, which can stand for any letter but which have a score of 0.

2. StanfordSpeak

Monday's lecture focused on codes and a variety of cryptographic techniques for sending secret messages. Stanford, of course, has its own codes that outsiders often have trouble understanding, mostly in its fondness for abbreviations such as *CoHo* for *Coffee House, SoCo* for *Sophomore College, MemChu* for *Memorial Church,* and so on. As a general rule, such abbreviations are formed by combining initial substrings of each word into a single string. That substring extends up to the first vowel, but also includes the following consonant if that letter is an *m* or an *n* (in linguistics, these consonants are called ***nasals*** or ***sonorants***).

Write a method

```
private String createStanfordAbbreviation(String str)
```

that returns the StanfordSpeak version of a multiword string. Make sure that your documentation includes what happens in various special cases, such as what happens when the string contains nonletters or a word contains no vowels. Once you have written this function, write a `run` method to test it.

3. Deleting characters from a string

Write a method

```
public String removeAllOccurrences(String str, char ch)
```

that removes all occurrences of the character `ch` from the string `str`. For example, your method should return the values shown:

```
removeAllOccurrences("This is a test", 't')  returns  "This is a es"
removeAllOccurrences("Summer is here!", 'e') returns  "Summr is hr"
removeAllOccurrences("---0---", '-')         returns  "0"
```

4. Adding commas to numeric strings (Chapter 8, exercise 13, page 290)

When large numbers are written out on paper, it is traditional—at least in the United States—to use commas to separate the digits into groups of three. For example, the number one million is usually written in the following form:

```
1,000,000
```

To make it easier for programmers to display numbers in this fashion, implement a method

```
private String addCommasToNumericString(String digits)
```

that takes a string of decimal digits representing a number and returns the string formed by inserting commas at every third position, starting on the right. For example, if you were to execute the main program

```
public void run() {
   while (true) {
      String digits = readLine("Enter a numeric string: ");
      if (digits.length() == 0) break;
      println(addCommasToNumericString(digits));
   }
}
```

your implementation of the `addCommasToNumericString` method should be able to produce the following sample run: