CS 106A Section 3 Handout (Week 4)

Strings

1. **Adding Commas to Numeric Strings.** Write a method named **`addCommas`** that takes a `String` representing a number and returns a `String` with a comma at every third position, starting from the right. (e.g. one million is written as 1,000,000). For example, your method should return the values shown below:

```
addCommas("147")                    returns "147"
addCommas("2014")                   returns "2,014"
addCommas("37897987")               returns "37,897,987"
```

2. **Removing Characters.** Write a method named **`removeAll`** that takes a `String` and a character as parameters, and removes all occurrences of the character (without using the String `replace` method.) For example:

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

3. **AlT cApS.** Write a method named **`convertToAltCaps`** that takes a `String` as a parameter and returns a version of the `String` in alt caps (where alternating letters are uppercase and lowercase). For example, your method should return the values shown below:

```
convertToAltCaps("hello")           returns "hElLo"
convertToAltCaps("section is AWESOME")  returns "sEcTiOn Is AwEsOmE"
```

Reading Files

4. **Class Presidents.** Somehow, the vote tallies for sophomore and junior class presidents got mixed up. Write a method called **`classPresidents`** that accepts a file of vote results as a `Scanner` and outputs the next sophomore and junior class presidents. The file format looks like the following:

```
Jared s 25 Sophie j 12 Tom j 44 Isaac s 30 Emily s 60 Russ s 23 Madison j 20
```

The file repeats the pattern *name year votes*. The year is either "s" for sophomore or "j" for junior. Your program should output the candidate in each presidential race with the most votes, and how many votes they got.

Specifically, if this file were named `candidates.txt`, your method could be called in the following way:

```
Scanner input = new Scanner(new File("candidates.txt"));
classPresidents(input);
```

and should produce the following output:

```
Sophomore Class President: Emily (60 votes)
Junior Class President: Tom (44 votes)
```

5. **Pig Latin.** Write a method named **`pigLatin`** that accepts as a parameter a `Scanner` representing an input file. Your method should, preserving line breaks, print out the input file's text in a simplified version of Pig Latin, a (silly) English variant where the first letter of each word is moved to the end. The rules for translating a word to Pig Latin are as follows:

If the word starts with a vowel (aeiou), simply append "yay".

English	Pig Latin
elephant	elephantyay
aardvark	aardvarkyay
easel	easelyay

If the word starts with a consonant, move the consonant to the end, and append "ay".

English	Pig Latin
switch	witchsay
welcome	elcomeway

As an overall example, if the input file `lincoln.txt` contains the following text:

```
four score and
seven years ago our
fathers brought forth on this continent a new nation
```

We might call your method as follows:

```
Scanner input = new Scanner(new File("lincoln.txt"));
pigLatin(input);
```

In this case, your method should print the following console output:

```
ourfay coresay andyay
evensay earsyay agoyay ouryay
athersfay roughtbay orthfay onyay histay ontinentcay ayay ewnay ationnay
```

To extend this program further, you can optionally implement the more advanced Pig Latin translation rules for words starting with constants. Specifically, if the word starts with constants, move all constants up to the first vowel (not just the first consonant) to the end, and append "ay".

English	Pig Latin
switch	itchsway
string	ingstray

6. **NegativeSum**. Write a method **`negativeSum`** that accepts a Scanner reading from a file containing a series of integers. Your method should print a message to the console indicating whether the sum starting from the first number is ever negative. You should also return true if a negative sum can be reached and false if not. For example, suppose the file contains the following text:

```
38 4 19 -27 -15 -3 4 19 38
```

Your method would consider the sum of just one number (38), the sum of the first two numbers (38 + 4), the sum of the first three numbers (38 + 4 + 19), and so on up to the end. None of these sums is negative, so the method should return false after printing the following message to the console: `no negative sum`

Suppose instead that the file contains the following numbers:

```
14 7 -10 9 -18 -10 17 42 98
```

In this case, the method finds that a negative sum (-8) is reached after adding 6 numbers together (14 + 7 + -10 + 9 + -18 + -10). It should return true, indicating that a negative sum can be reached, after printing the following message to the console: `-8 after 6 steps`

Note: For each of the problems above, we ask you to write a method that takes input as parameters and, in some cases, returns some result to the caller. For extra practice, try writing code to call each of these methods (and pass in the appropriate parameters) within the **`run()`** method!