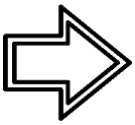
Assignment

Overview

Pig Latin is a fun play on words in which words in English are altered, usually by adding a fabricated suffix or by moving the onset or initial consonants of a word to the end and adding a vocalic syllable to create such a suffix. There can and are different rules that people make up/follow when converting words to pig latin so what you see here may be different elsewhere.

You are to create a program that takes words and converts them to their pig latin form.



Directions

To Get Started:

1. Create a new java project in IntelliJ(or your preferred IDE) called YOUR_MCC_Name-Assignment5.

2. Under the src folder create a new java class called **PigLatinConverter.java** this will hold the code for that assignment.

For our pig latin rules we are going to use the following:

- If the word starts with a vowel, you will simply add a "yay" to the end of it
  - For example, apple translates to appleyay, elephant would be elephantyay, etc.
- If it doesn't start with a vowel you will need to find the first occuring vowel then move all the letters in front of it to the back and add "ay" to the end.
  - For example, programming translates to ogrammingpray, banana would be ananabay, etc.
- If there are no vowels (like acronyms/abbreviations) just keep the word and add "ay" to the end of it.
  - For example MCC short for our college, should be mccay.

The simple algorithm for this would be to loop through the word each character at a time. See if the letter is a vowel, when you find it, save the index location and break out of the loop. Then use substring methods to take from where you found the vowel to the end of the word, concatenated with the beginning to the first vowel and add the "yay" or "ay" to the end.

Now we could assume the user will enter in one word, but that is super simple. We have the tools to do more than just one word AND this is a Java 2 class. We can do more!

The program should take a complete sentence for input and convert each word. We have the tools to break a string up via a space (" ") character and saves it to a String array. You simply then need to loop through the String array and convert each word and print it out as a sentence. Potentially you could make your own method that you send a word to and it returns back the converter word to save/add to an array.

Extra Credit:

There are multiple ways for some extra credit here if you would like.

- Sentences are cool and all but what about a whole text file? If you can modify your program to ask for the name of a text file, then open that file, read in the words from a text file, convert them and write to another file will give you some bonus points. You will need to pay attention to periods and some capitals letters to make the words look good.
- Speaking of capital letters, if you take the word letter for letter any words that start with a capital letter will have that letter possibly towards the end of the word. So to fix this you can add some logic that checks if the first letter of a word is capital and if so when doing the conversion make all letters lowercase then upper case the new letter at the beginning. We still have those capital letters in proper words and beginning of sentences. So again more bonus points if you add this in.

Example Output(user input bolded):

```
Enter in words to convert: Programming classes at MCC are cool
Conversion: ogrammingpray assesclay atyay mccay areyay oolcay


Enter in words to convert: Live for today and let tomorrow come later
Conversion: ivelay orfay odaytay andyay etlay omorrowtay omecay aterlay
```

Submission

1. Compress the IntelliJ project folder and submit it to this assignment.

   to compress: On Windows, right click -> send to -> compressed .zip file

   On Mac, right-click -> Compress

Hints/Tips (Before Submitting):

- *Consider checking for periods in words, or just make sure to leave them out for the base translations*
- *Again it would be useful to have a separate method to which you can send a word to as a String argument that then loops through and sends back the converted word.*
- *Get the main program done then make a copy of the file and add in the extra credit work if you choose to do that.*
- *When testing a text file just create your own, copy book text or some open source text to see that it works. I have my own file to test.*
- *Don't forget to have a header at the top of your file and include a **Resource statement**.*
- *Use comments throughout for full points.*
- *Follow all Java Styling Guides as covered*