Pig Latin

Part 1 Step 1

Pig Latin is a pretend language.

The basic rule is simple: move the leading consonants to the end of the word and append "ay".

pig -> igpay
latin -> atinlay
this -> isthay
strange -> angestray
bcdfgh -> **** INVALID ****

If the first letter is a vowel then just append "way".

apple -> appleway
eye -> eyeway

STEP 1: Write a program that translates words from English to Pig Latin. Prompt for an English word and output the translation in Pig Latin. If the word has no vowel, print "**** INVALID ****". Repeat this sequence until the user wants to quit.

Program to import – PigLatin_Shell.java

Special Cases – Part 1 Step 2

1) **If the first vowel is a "u" and the letter before it is a "q" then the "u" also goes to the end of the word.**

question -> estionquay
squeeze -> eezesquay

2) **Treat "y" as a consonant if it's the first letter of a word; otherwise, treat "y" as a vowel if it appears earlier than any other vowel.**

yes -> esyay
rhyme -> ymerhay
try -> ytray

STEP 2: Modify your program to handle these special cases.

Capitalization and Punctuation – Part 1 Step 3

If the first letter is capitalized in English then the first letter should be capitalized in Pig Latin also. Capital letters in the middle of the word are ignored and remain capitalized.

Thomas -> Omasthay
Jefferson -> Effersonjay
McDonald -> OnaldmcDay
McArthur -> Arnoldmcay

If before-and-after punctuation is used in English then it should remain in Pig Latin. Apostrophes are ignored and remain where they are.

What? -> Atwhay?
Oh! -> Ohway!
"hello" -> "ellohay"
"Hello!!!!" -> "Ellohay!!!!"
don’t -> on’tday

STEP 3: Modify your program to handle capitalization and punctuation.

Part II: PigLatinize a Text File

Modify your program to prompt the user for the name of a text file (“`PigLatin.txt`”) to read from, and a text file to output to (“`PigLatinOutput.txt`”). Translate all the words in that file from English to Pig Latin, then output the words into the new text file.

Use one Scanner objects to read in the filename. Use another Scanner to actually read the file. Use try-catch blocks to catch the exception. Look it up if you don't remember the syntax.

Preserve the line structure as you pig-latinize each line. There are several ways to do this, but they all use nested loops. You may want to look at `nextLine`, `next`, `split`, or `StringTokenizer`. Your code should work for any file. That is, there may or may not be blank lines, so you should test for them.

Files to Use:

```
PigLatin.txt
LittleFrog.txt
declaration.txt
```

Extension

Output each Pig Latin word in reverse, but preserve capitalization. For example, “Atwhay?” becomes “Yahwta?”