CS198 Intermediate Systems Administration DeCal

Lab #6

Michael Gasidlo

This lab will test your skills with shell scripting. You need, need, need to have the lecture (or an equivalent shell scripting guide) open while doing this lab, as you're going to be using things you learned in class. There's only going to be three questions because two involve you writing significant shell scripts! The third is mostly for fun, but that doesn't mean you don't gotta do it! Email your *code* to mgasidlo+decal@ocf.berkeley.edu.

I. TV Episode Renamer

You work for a television company. For archival, the company is keeping copies of the first season of its new show, "America's Next Top Student Model." However, the files are named oddly:

s01e01.avi
s01e02.avi
s01e03.avi
... more ...
s01e10.avi

You have been given a list of official episode names and you have been tasked with renaming the files into this format:

1$EPISODE - $TITLE.avi

where $EPISODE is zero-padded (10 remains 10, but 3 becomes 03.)

The episode names are as follows:

Top Student Ahram
Top Student Dwight
Top Student Hubert
Top Student Micah
Top Student Jessica
Top Student Antony
Top Student Jordi
Top Student David
Top Student Nathan
Top Student Jose

(thus, episode 1 would be properly named "101 - Top Student Ahram.avi.")

Write a shell script that uses this file as input and renames each file in the above format. I should be able to run it with:

```
$ rename-tv < episode_names
```

in a directory that contains all the original filenames.

Hints:
- Remember to prefix your script with #!/bin/sh and then chmod +x it before you run it. If it won't load, do you need to put "./" in front of the program name?
- Copy and paste the episode names into a file and use that for your testing.
- If you want to create empty files with the original filenames mentioned above, do this:

```
   for i in `seq 1 9`; do touch s01e0$i.avi; done; touch s01e10.avi
```

II. Pig Latin Revisited

Last lab, we tried to figure out a regular expression to convert a sentence into Pig Latin. It turns out that it is much easier to do it on a word by word level with a shell script, and you've learned all you need to do it.

Assume sentences only contain letters and spaces, don't consider having to special case numbers. You can either use standard input (read) or pass the sentence on the command line, whichever you think is easier. For the former, I should be able to run it like this:

```
$ echo "The quick brown fox jumped over the lazy dog" | piglatin
```

for the latter,

```
$ piglatin The quick brown fox jumped over the lazy dog
```

Here's how pig latin works:
- for words that start with a consonant, move the first letter to the   end then tack on -ay.
- for words that start with a vowel, just add -ay.

Here are some tips:
- if you intend to use a read loop, read will read a whole *line* into a variable. To split it into words, you can include it into an inner loop like this:

```
  for word in $sentence; do
        ...
  done
```

  Due to the way the shell behaves, each space-separated word in the sentence will be iterated over one at a time. Compare this to

```
  for word in "$sentence"; do
```

  In this case, there will be only one item in the loop and it will be the entire contents of $sentence.
- You can compare the first character of a word using case.

```
  case "$word" in
        [aeiou]*) ... vowel case ... ;;
        *) ... everything else ... ;;
  esac
```

  (note: yours has to handle uppercase letters too.)

III. FizzBuzz!

At least one software company uses a test called FizzBuzz! to weed out candidates for programming jobs. The test is simple: for the numbers 1-100, print the number. However, if the number is divisible by 3, print Fizz instead. If the number is divisible by 5, print Buzz. If the number is divisible by both 3 and 5, print FizzBuzz. Sounds simple, right? Turns out that 9 in 10 applicants for programming jobs can't do it (including CS majors and self-proclaimed “senior programmers”). Can you?

If you're confused, please come to office hour!

REMINDER: Project proposals are due in <u>hardcopy</u> at the beginning of next lecture! For a sample proposal, check the website. If you have no idea what to do, email the facilitators and we'll try to help you come up with something.