Week 09 Homework 💻 🐙

New stuff learned this week:

- the regexp pipe `|` metacharacter is for *alternation* (like an OR) — it is usually used with parentheses, so `(goat|llama)pack` would match *both* `goatpack` and `llamapack`
- in a regular expression, parentheses *capture the thing matched*
- things matched inside parentheses are available in the *replacement string* by special numbered codes, called *backreferences,* because they reference backwards to things matched in the regular expression, like this `sed -E 's/paint(ing|ed)/draw\1/` would change the phrase `I like painting, I painted a goat` to `I like drawing, I drawed a goat`
- the `g` flag works in `sed` like in `vim` — it causes the replacement to apply *globally* — that is, to every occurrence of the match on the line, `sed -E 's/foo/bar/g'`
- the `i` flag makes the match *case-Insensitive* — meaning, it matches whether the letters are uppercase or lower case, like `sed -E 's/foo/bar/i'`
- flags in regular expression can be combined by just smushing them together with no space, and the order doesn't matter, so you can do `/gi` or `/ig`
- In computer science, ***types*** are used to give *meaning* to bits of computer memory, which are all ones and zeros. Types allow us to say, "this group of bits represents a word", "this group represents a number", "this bit means a true/false value", etc.
- The most common, basic types in computer science (often called "primitive types") are `boolean` (true or false), `string` (sequences of text-characters, like words and sentences), and `number` (which are sometimes, depending on the level of the language, broken out into sub-types of numbers like `integer` `float` etc…)

Touch Typing Links:

- http://touchtype.co
- https://www.how-to-type.com

Homework plan:

- 1 day creating just a few more flash cards
- 1 day reviewing all flash cards

- 2 days CLI practice
- 2 day `vim` practice
- 2 days touch-typing practice
- watch CCCS#8 — twice

Homework day 1:

- do flashcard assignment (see below)
- touch typing practice
- `vimtutor` - just lessons 1.3, 1.4, 1.5, all of lesson 2, and all of lesson 3

Homework day 2:

- CLI practice #1
- watch CCCS#8

Homework day 3:

- touch typing practice
- `vimtutor` — Everything *except* Lesson 7

Homework day 4:

- CLI practice #2
- watch CCCS#8
- review all your flash cards

Flashcard assignment

Add new cards for REGEXP including:

- `()` - parens for *capturing*
- `|` - alternation
- `\1\2` - backreferencing

CLI Homework #1

1. carefully review the “New stuff we learned” section above ^
2. `ssh` into your home dir, and make a new directory called `week9`
3. move into the new `week9` directory, and in a single command, create 3 new empty files called `one.txt` , `two.txt` and `three.txt`
4. make a new directory inside of `week9` called `numbers`

5. in one command, using shell expansion so you don't have to type out all 3 file names, move all three files you created in step 2 into the `numbers` dir
6. now, using shell expansion again so you don't type both filenames, *copy* the `two.txt` and `three.txt` files back up into the `week9` dir. Your shell expansion should select only those two files and *not* copy `one.txt` .
7. change directory down into the `numbers/` dir
8. make a *bash variable* called `DAY` that contains the current day of the week (Tuesday, Wednesday, etc…)
9. Use the bash variable `DAY` and a command to make the contents of the file `one.txt` become the string of text: `Today is <day-of-week>`
10. Type a command to add a line of text to `one.txt` that reads `Tomorrow is not <day-of-week>`
11. Type a command to see the contents of `one.txt` — it should have two lines.
12. Type a command that copies the contents of `one.txt` into the file `two.txt` *but adds a period at the end of each line.*
13. Type a command that prints your current working directory to standard out
14. type a command that puts your current working directory into the file `three.txt` but changes the beginning from `/home/` to `/home-sweet-home/` (so for my user, the contents would end up being `/home-sweet-home/ubuntu`
15. rename the `one.txt` file to be `day.txt` and then rename `three.txt` file to be `home.txt`
16. still from your `numbers/` dir, in a single command make a copy of the whole `numbers/` directory, resulting in a new directory called `cool-stuff` inside of your `week9` dir.
17. use a command to look around both the `numbers` and the `cool-stuff` dirs to make sure they both have all of the files
18. change directory up in your home dir *by typing only TWO characters*
19. remove the entire `week9/numbers` dir in one command, including it's contents
20. in one combined command, make a new directory inside of `week9` called `regex` and move into it (hint: remember that you can chain together bash commands on a single line using the `&&` operator)
21. copy a file from my home directory called `pr1.txt` into your `regex` folder, which should be where you are currently, after step 19.
22. `cat` out the `pr1.txt` file and transform it using `sed` so that the first line reads: `I like to FOOBAR, but FOOBAR is hard.`
23. Now, change the `sed` command so that the first line becomes `I like to phone, but phoning is hard.`
24. Now, modify your last command a bit so that the second line remains untouched, it should still read `What is your genotype?`
25. Change your `sed` expression so that the third line reads `FOO FOO FOO dan fooban lan`
26. Change your `sed` expression so that the third line reads `BAR man BAR dan BAR lan`
27. Write a new `sed` expression so that the fourth line starts with `Chickens are cute. I like my chicken.` (make sure the `c` 's are capitalized correctly to match the original)
28. **Extra Credit:** ✨ Modify your command from step 25 so that the end of the line still reads `cat is also a unix command` , it should *not* say `chicken is also a unix command` .

CLI Practice #2

1. `ssh` into your home dir and move into your `week9` dir
2. copy the file `pr2.txt` from my home dir into your `regex` dir, without leaving your `week9` dir
3. print the exit code of the last command to standard out
4. move into your `regex` dir and `cat` out the `pr2.txt` file
5. rename the `pr2.txt` file to be `sick.txt`
6. type a command that will print out only the first 5 lines of `sick.txt`
7. type a command that will print out only the last 5 lines of `sick.txt`
8. type a command that will print out a chunk in the middle of `sick.txt` starting with the line `My hip hurts…` and ending with the line starting with `I have a sliver…` .
9. type a command that will print to the screen only the first two lines of the poem, but changed so that both the word `today` and `McKay` are changed to `FOOBAR` .
10. change your regular expression so that every occurrence of the lowercase letter `a` is changed to `@`
11. Now, using a *flag* make it so the `A` in `Peggy Ann McKay` is also changed to a `@`
12. Now, repeat step 10, but this time do it without a flag, using *alternation* instead
13. Type a command to see all of the bash commands you've recently typed, and find the one that you used correctly to solve step 7. Then use `!<number>` to repeat that command
14. Use the up arrow to repeat the command from step 12, and this time change it with a `sed` command so that the last two lines read `My nose is cold, my toes are nerd.` and `I have a sliver in my therd.`
15. `cat` out only the first 9 lines of the poem, and in a single `sed` command change the last two lines so that `sixteen` becomes `ninety-six` and `seventeen` becomes `ninety-seven`
16. change your regular expression so that the first 3 characters of every line are preserved, and the last three characters are also preserved, but everything in between is shortened to three dots. So, for instance, the first line should become `"I …,"` And the second line `Sai…ay.`
17. In about five commands, without using `vim`, make a new file contains the poem from `sick.txt` but has a line at the top that says `Sick` and one at the bottom that says `by Shel Silverstein` . There should also be a blank line after the title of the poem, and another one before the last line. Name it `poem.txt`
18. `cat` out the contents of `poem.txt` and pipe it to `sed`, changing `Peggy Ann McKay` to your full name, then redirect the output to a new file called `my-poem.txt`
19. make a copy of `my-poem.txt` in my home directory, and name it `poem-<yourname>.txt`