**13 – Navigating Code**

**Activities**

COMP190 – Tools and Techniques for Software Development

Dickinson College

Fall 2022

**Name:**

**Introduction:**

In this class we saw two tools that are useful for finding your way around large code bases - find and grep. The find tool will allow you to find the location of a file or collection of files that might be important to your work (e.g. all of the html files, or all of the files that contain test code). The grep tool allows you to search within files to determine which files have content that might be relevant for your work (e.g. contain the code for a particular UI element, or access a particular URL/api, use a specific variable, or call a particular function). We learned about the *regular expressions* that grep, and lots of other search tools, use to specify what is to be matched (i.e. searched for). These activities will give you some additional practice with find, grep and regular expressions. You will then use these tools to assist you in doing some tasks in the FarmData2 project that are typical in software development.

**Getting Started:**

1. You’ll be using the KitClient container and the GitKit\_FarmData2 repository that you used for the prior few assignments for this assignment as well. You’ll want to refer to your prior assignments for the run command that you need to get the KitClient container running.

Nothing is required here. Just ensure that you are able to run the KitClient container.

**The find Tool:**

Recall that the general form of the find command to search for files by name is:

find <starting dir> -name "<pattern>" -print

2. Use a find command to find the full path to the PULL\_REQUEST\_TEMPLATE.md file in the GitKit\_FarmData2 repository.

Paste a screenshot of your find command and its output as your answer.

3. Recall that earlier in the semester we learned about bash shell scripts. FarmData2 uses a number of these scripts. Use a find command to find all of the bash shell scripts in the GitKit\_FarmData2 repository. You should find that there are 7 of them. Hint: Recall that we saved our bash scripts in files with a .bash extension.

Paste a screenshot of your find command and its output as your answer.

4. FarmData2 uses a tool called Cypress to write automated tests. These test files are called spec files and they all have a filename like: info.spec.js where the first part of the name (e.g. info) specifies what is being tested, and the rest of the file name (.spec.js) indicates that the file contains tests (specs) for JavaScript (js). Use a find command to find all of the files in the GitKit\_FarmData2 repository that contain Cypress tests. There are 13 of them.

Paste a screenshot of your find command and its output as your answer.

5. On a Linux system all of the executable programs are typically stored in directories named bin. The name bin is short for binaries, which is just what Unix/Linux calls its executable files.

a. Use the find command to find all of the bin directories on the system. Hint: Think about what your <starting dir> should be. Your result will likely contain a mixture of paths to bin directories and “Permission denied” errors. List just the paths to the bin directories here. There should be 6 of them.

b. Briefly explain why your output from part a also contains “Permission denied” errors. Hint: Think back to earlier lessons on file permissions.

**Regular Expressions:**

The exercises in this section will give you some practice understanding and writing regular expressions. The following are not required reading but if you find yourself having trouble with the exercises, would like to see regular expressions explained in a different way, or just want to review, these sites do a nice job. They each cover a few things that we haven’t, so feel free to skip those parts.

* <https://regexone.com/> (Through Lesson 11)
* <https://ryanstutorials.net/regular-expressions-tutorial/> (Through Grouping)
* <https://www.datacamp.com/cheat-sheet/regular-expresso> (A cheat sheet)

6. Each of the following parts gives a regular expression and a list of strings. Highlight the strings that are matched by the regular expression.

a. .l.nk

blank

flank

rank

bank

a blank bank cheque

b. ab+c?\d\*

abc

ac27

ac

ab

abbb

abbb99

bbc

abbcc12

c. ^a(ab)\*a

aa

abababa

aaba

aababbbb

aabbaa

aaaba

aabababa

d. (very )+(cute )?(big|small) [dcf]

very cute dog

cute big cat

very very cute small duck

very very very big cow

small fish

very very small flamingo

7. Consider the following regular expression:

(dog){2} (cat){2,4} (fish){,3}

a. Give the shortest string that will match the above expression.

b. Give the longest string that will match the above expression.

8. Each of the following questions gives a task that can be solved using a regular expression. Use the regular expression tester at the URL below to develop and a regular expression that solves the problem: <https://www.regextester.com/>. Be sure to check the “multiline” option under “Flags” in the tester for best results.

a. Create a regular expression that matches the decimal numbers that are highlighted in the following text.

Speed of light in vacuum 299792458 m/s

Acceleration of gravity 9.80665 m/s^2

Standard atmosphere 101325 Pa

Earth to sun distance 149600000 km

Circumference to diameter ratio 3.141592

Gas constant 8.3144621 J/mol\*K

Paste a screenshot from the RegEx Tester site showing your regular expression and the test string showing the above text with the matches highlighted.

b. Match the entire lines for the movies produced before 1990.

1 The Shawshank Redemption (1994)  
2 The Godfather (1972)  
3 The Godfather: Part II (1974)  
4 Pulp Fiction (1994)  
5 The Good, the Bad and the Ugly (1966)  
6 2001 a Space Odyssey (1968)  
7 12 Angry Men (1957)  
8 Schindler's List (1993)  
9 The Lord of the Rings: The Return of the King (2003)  
10 Fight Club (1999)

11 Blues Brothers (1980)

12 Shrek (2001)

13 1920 (2008)

14 1984 (1984)

Paste a screenshot from the RegEx Tester site showing your regular expression and the test string showing the above text with the matches highlighted.

c. Match hexadecimal numbers with between 2 and 6 digits. A valid hexadecimal number will start with the prefix 0x and then be followed by the digits

0,1,2,…,9,A,B,C,D,E,F

For example: 0x3D89F2 is a valid hexadecimal number with 6 digits.

0x72AC

0x01AF89

0xABCDEF

0xAB

0xAG03

F793

0xAABBCCDD

0x7

123456

Paste a screenshot from the RegEx Tester site showing your regular expression and the test string showing the above text with the matches highlighted.

**Finding code with egrep:**

One of the challenges in working on a large project is finding the files that need to be modified to fix an issue or to add a new feature. The exercises in this section will give you some practice using egrep with regular expressions to find the files relevant to an issue. You should be working from the command line in the KitContainer for these activities.

9. In the issue tracker for the GitKit-FarmData2 repository that we have been using, find the ticket for the issue that asks that the total row/bed information be removed from the Seeding Report summary table.

a. What is number & title of this ticket and what labels does it have?

b. What is the text of the line of output that is to be removed?

c. If you were to decide to work on this issue you would need to find the source code file that produces that output in the running application. Use the egrep command below to find the relevant file in the GitKit-FarmData2 repository.

Paste a screenshot of your egrep command and its output as your answer.

d. Examine the output from part c. Which line of which file produces the output that is to be removed?

10. In the issue tracker for the GitKit-FarmData2 repository that we have been using, find the ticket for the issue that requests that the default input for creating a seeding log be bed/feet.

a. What is number & title of this ticket and what labels does it have?

b. Use egrep to find the file and line number that you would need to change to address this issue. Hint: As was shown in the previous question you can use some of the text that appears in the user interface (UI) to search for the file that contains it. It may take you several tries to find a regular expression that will narrow your search down to find the correct line. Don’t worry too much about fully understanding all of the code that you find - when you find the right line, you will be able to see the text from the UI in the code.

Paste a screenshot of your egrep command and its output as your answer.

c. Examine the output from part b. Which line of which file would need to be modified?

**Putting grep to work in FarmData2:**

As you know FarmData2 is built as an extension of another FOSS project named farmOS. As such some of the documentation that appears in the FarmData2 repository, in its markdown (.md) files links to pages in the farmOS documentation. The activities in this section ask you to find and fix some links in the FarmData2 documentation so that they point to the correct pages in the farmOS site.

11. In the issue tracker for the GitKit-FarmData2 repository that we have been using, find the issue about Replacing links to the farmOS API.

What is number & title of this ticket and what labels does it have?

12. Use egrep to find all of the places in your GitKit-FarmData2 repository that would need to be modified to address the issue you found in the prior question. There should be four lines across two markdown (.md) files that will need to be modified. Give a screenshot showing your egrep command and its output.

13. Use the git/GitHub workflow that we learned in the prior classes to fix the issue and make a pull request for your fix. Remember to synchronize your main branch with the upstream before beginning your work, and to commit your changes to a feature branch with a meaningful commit message for each change. When making your pull request, be sure to use, a descriptive message and to identify the issue being fixed by including a “Fixes” or “Closes” line.

Your pull request should be able to be merged automatically.

Give the full URL to your pull request here:

14. Please complete the following very short (~5 minute) post-course survey that pairs with the one that you completed at the start of the semester. This will help us to evaluate and improve this course.

<https://forms.office.com/r/a7tsqE2tBi>

**Optional:** To help us improve and scope these activities for future semesters please consider providing the following feedback.

a. Approximately how much time did you spend on this activity outside of class time?

b. Please comment on any particular challenges you faced in completing this activity.