

Please complete the following questions. If you need help, feel free to ask questions both from other students and from me. If you cannot finish during the class time then you should finish this lab at home. After you finish, feel free to clean up (*i.e.*, remove) any files or directories that were created, you won't need them in the future. But be careful not to remove any other important files you may have in your home directory.

1. Preliminary.

- (a) Change your location to the 'cse399' directory you created in previous homeworks or labs (create it again if you deleted it). Copy the directory /home1/m/mcorliss/teaching/cse399/lab3 and all its subcontents to your current location. Change your location to this new directory (lab3).

2. *find*.

- (a) Notice that most files in the directory lab3 are named using the form "<str1>-<str2>-<str3>-<str4>.txt". Use 'find' with the '-regex' flag to find all files of this form where <str3> is 'cse399'.

Answer:

```
bash$ find . -regex "[^]*-[^]*-cse399-.*"
```

- (b) One of the four strings in the filename is a numeric value (*i.e.*, 1-9). Find all files where this string in the filename is "3" or "9".

Answer:

```
bash$ find . -regex "*/[39].*"
```

- (c) Using the OR flag (-o) with 'find', find all the files from the previous exercise as well as all the files that begin with 'cse399'.

Answer:

```
bash$ find . -regex "[^]*-[^]*-cse399-.*" -o -regex "\./cse399.*"
```

- (d) Repeat the find from the previous exercise, except instead of using the OR flag, use only one regular expression. This will require a new regular expression symbol that was not discussed in the slides '|' (OR – needs to be escaped). "\\(expr1\\)|\\(expr2\\)" will match on either expr1 or expr2.

Answer:

```
bash$ find . -regex "\\([[^]*-[^]*-cse399-.*\\)|\\(\\./cse399.*\\)"
```

3. *grep*.

- (a) In the file 'webpage.html', select all lines that contain a header tag (*e.g.*, <h1>, <h2>). Your command should not select other non-header tags such as <hr>.

Answer:

```
bash$ grep "<h[1-6]>" webpage.html
```

- (b) In the file 'assembly.txt' is some assembly information for a program. Select all lines that contain a non-alphanumeric character that is not a space.

Answer:

```
bash$ grep "[^A-Za-z0-9]" assembly.txt
```

- (c) Each line in the file 'assembly.txt' is separated into columns. Use grep to select lines that have a non-zero "Value" field. Your command should also exclude lines that do not have a numeric value in the Value field such as the first few lines of the file.

Answer:

```
bash$ grep "^[^|]* [0-9]" assembly.txt
```

- (d) In the data file 'data-1.txt' there are three types of lines: 1) lines of the form "!", 2) lines of the form "<str1>:<str2>:<str3>", and 3) lines of the form "<str1>:<str2>:<str3>:<str4>:<str5>". Use grep to select the second type of lines ("<str1>:<str2>:<str3>"). Do *not* use the negation flag (-v).

Answer:

```
bash$ grep "^[^:]*:[^:]*:[^:]*$" data-1.txt
```

4. sed.

- (a) The file 'data-2.txt' contains a number of columns separated by ':'. Use sed to remove only the first column (printing the output to the screen).

Answer:

```
bash$ sed -e 's/^[^:]*://' data-2.txt
```

- (b) Use sed to interchange columns 2 and 3 in the file 'data-2.txt'.

Answer:

```
bash$ sed -e 's/([[:*])\(:[[:*])\(:[[:*])\1\3\2/' data-2.txt
```

- (c) Repeat exercise 2(a), but pipe the output to sed. Use sed to change the string "unix" in the filename to "linux", but only when it appears as <str2> (where the filenames are of the form "<str1>-<str2>-<str3>-<str4>.txt").

Answer:

```
bash$ find . -regex "[^]*-[^]*-cse399-.*" | sed -e 's/([[:*])\(-unix-\1-linux-/'
```

- (d) Repeat exercise 3(c), but pipe the output to sed. Use sed to extract only the Value field and to remove all preceding 0's (e.g., '0430' becomes '430').

Answer:

```
bash$ grep "^[^|]* [0-9]" assembly.txt | sed -e 's/^[^|]* [0-9]*\([0-9]*\).*\1/'
```