Important: Before you leave, log out of the computer. Do this with `Ctrl-d` or `exit` .

If you don't finish the lab today, the scripts are due by 4:00 PM on Friday. They must all be turned in with turnin.

Part 1

This assignment is similar to the glob assignment from last week. Instead of listing files with ls, it will use egrep to list matches from the contents of files.

1. Some of you failed to have your globbing script cd to the discworld directory. As a result, it failed. It's partly my fault - I didn't include that instruction in the lab. However, I did add it in class. On the other hand, if you did this and turned the script in anyway, you never tested it and then ignored the obvious incorrect output when you ran it through turnin. So edit globbing.sh and add the line

   ```
   cd /home/mathcs/courses/cs246/discworld
   ```

   after the comments at the top. There is an example of this below. After fixing your script, resubmit it with turnin.

2. Log in and cd to your cs246 directory.

3. Edit a shell script called regex.sh and insert comments at the top with your name, the assignment, and a description. You will also need a cd command to change the directory of the script.

   ```
   #!/bin/bash

   # Name: Your Name
   # Assignment: regex
   # Purpose: Regular Expressions Practice

   cd /home/mathcs/courses/cs246/discworld
   ```

   Save the file with `Ctrl-x Ctrl-s` .

4. You're going to need to use the shell while editing the script. Split the screen vertically with `Ctrl-x 3` . Then run a shell in the other window with `Meta-x shell` . On our keyboards, the Meta key is Alt.

5. You should now have an edit window for your script on the left and a shell window on the right. You can switch back and forth with `Ctrl-x o` (the o is for other window). Try this out.

6. Next move to the shell window and split it vertically using `Ctrl-x 2` . You are going to need a second shell window. Notice that anything you type in one shell window is also displayed in the other.

7. To get a second independent shell you need to rename the first one. Do that with `Meta-x rename-buffer`. It will ask you for a buffer name. Give it `discworld` (or anything that will identify this shell).
8. Now do `Meta-x shell` again. This will start a new shell in the other window called shell. Switch to that shell with `ctrl-x o`.
9. In this window, use `cat regex.sh` to make sure that you have saved your script and that you are in your cs246 directory.
10. Move back to the discworld shell window and cd to the directory containing the discworld files.

    ```
    cd /home/mathcs/courses/cs246/discworld
    ```

11. List the files in this directory with `ls -l`.
12. Use `cat` to display some of the files so that you can see what they look like.
13. Use cd to move into the discworld directory. Then list the files with ls. This directory contains several files whose names are characters in the Discworld series of book by Terry Pratchett. Each file contains the title of the books in which the character appears, but we won't use the contents today.
14. To practice regular expressions, you are going to use regular expressions to list the contents of all the files in this directory that match various patterns. For each one, you will add three lines to the shell script in the other window.
    - Use echo to print the part number (actually letter) of the exercise part from the list below.
    - Use the egrep command with the -h option to list the lines from the files for a particular pattern. The -h option means don't print the name of the file containing the match.
    - Your egrep command will look like this:

      ```
      egrep -h regex * | sort | uniq
      ```

      where regex is the regular expression for a particular pattern. Most of them will require quotes.
    - Use echo to print a blank line.
15. For each pattern, test the pattern in the terminal window first by simply typing the egrep command. Then switch back to the edit window and add it to the script. Do not use options other than -h. Make sure your commands are **exactly** as shown below (capitalization, spelling, spacing).

16. You should also test the script periodically by saving it with `Ctrl-x Ctrl-s`, switching to the correct shell window, and running it. The script should look like this (I've done the first two parts for you):

```
#!/bin/bash

# Name: Your Name
# Assignment: regex
# Purpose: Regular Expression Practice

cd /home/mathcs/courses/cs246/discworld

echo Part 1
egrep -h '^S' * | sort | uniq
echo

echo Part 2
egrep -h 'h$' * | sort | uniq
echo

 ...
```

Here are the patterns:

1. Titles beginning with S.
2. Titles ending with h.
3. Titles beginning with T and ending with t.
4. Titles containing ou.
5. Titles containing ou or in.
6. Titles containing r followed by a, possibly with characters in between.
7. Titles containing either a followed by r or r followed by a, possibly with characters in between.
8. Titles containing non-space, non-alphabetic characters.
9. Titles containing an exclamation point.
10. Titles containing exactly one exclamation point.
11. Titles whose length is a multiple of 6.
12. Titles beginning with T whose length is a multiple of 3.
13. Titles that do not contain either an o or an e (upper or lower case).

Part 2

1. You won't need the discworld window anymore so switch to it and make it go away with `Ctrl-x 0`.
2. Make sure you are in your cs246 directory and use seq to create a file containing the numbers 1-20.

   ```
   seq 20 > numbers
   ```

3. Use cat to list the file.

   ```
   cat numbers
   ```

4. Read the man page for the head command.

   ```
   man head
   ```

5. Use the head command to list the first 10 lines of the file.

   ```
   head numbers
   ```

6. Use the head command to list the first 5 lines of the file using the file name as an argument.

   ```
   head -n 5 numbers
   ```

7. Use the head command to list the first 5 lines using redirection.

   ```
   head -n 5 < numbers
   ```

8. Use the head command to list all but the first 5 lines (the man page tells you how).
9. Read the man page for the tail command.
10. Use the tail command to list the last 10 lines of the numbers file.
11. Use the tail command to list the last 5 lines.
12. Use the tail command to list the lines starting with line 5 (the man page tells you how).
13. Now use a combination of head and tail in a pipeline to list lines 7 though 13, using head first. You'll need to replace the question marks below with the right options.

    ```
    head ???? numbers | tail ????
    ```

14. Next use a combination of head and tail in a pipeline to list lines 7 though 13, but use tail first. You'll need to replace the question marks below with the right options.

    ```
    tail ???? numbers | head ????
    ```

15. Now write a shell script called select1 that takes 3 command line arguments: a file name and 2 integers. If we call the integers m and n, then the script will print lines m through n of the file. Remember that in the script, the arguments are $1, $2, $3. So $1 will be the file name, $2 will be m, and $3 will be n. Use a pipeline with head and then tail to print the lines.

16. Make sure that it's executable and test it out.

17. For the next script, you'll need to compute the number of lines in a file. Try

    ```
    wc -l numbers
    ```

    The `-l` option of wc makes it only list the number of lines.

18. We need to strip the file name from the output of wc to just get the length. Try

    ```
    wc -l numbers | cut -d ' ' -f 1
    ```

    Cut normally expects field to be separated by tab characters. The `-d` option specifies a different delimiter (in this case, a space). The `-f 1` option selects the first field.

19. Finally we need to store the length in a variable. This requires command subsitution.

    ```
    length=$(wc -l numbers | cut -d ' ' -f 1)
    ```

    Try this out.

20. In the next script you'll need to do a subtraction. Try this out:

    ```
    a=15
    b=3
    c=$(($a - $b))
    ```

    You can do arithmetic inside double parentheses, but you need the dollar sign to capture the output.

21. Now write a script called select2 that is like select1 except that it uses a pipeline with tail first and then head. You'll need to compute the length and do a subtraction. It must work with files of any size.

Make sure all of your scripts are excutable and turn them in with turnin.