This assignment assumes you have read Chapters 2, 3, 4 and 5 of Sobell. Some specific cross-references may also be given in the questions below.

Prepare your answers to the questions below in a single plain ASCII text file[1]. Submissions in other formats will be ignored. If you work with a partner, make sure the submitted file contains a properly-completed copy of the partners form posted on the assignments page. Failure to do that will result in at least one of you not receiving credit for the assignment.

Submit your file to the Curator system, under the heading `L02`, by the posted deadline for this assignment. No late submissions will be accepted.

For questions 1-5, suppose that your userid is `cs2505` and that a listing (`ls -l`) of the contents of your current working directory provides the following information:

```
cs2505@beech submissions]$ ls -l
total 900
-rw-rw---- 1 cs2505 cs2505 16407 Aug 29  2018 aac7594.txt
-rw-rw---- 1 cs2505 cs2505  8988 Aug 29  2018 abhir.2.txt
-rw-rw---- 1 cs2505 cs2505  6239 Aug 29  2018 ajbarnes.txt
-rw-rw---- 1 cs2505 cs2505 10746 Aug 29  2018 alamon.txt
-rw-rw---- 1 cs2505 cs2505  4097 Aug 29  2018 alek6.txt
-rw-rw---- 1 cs2505 cs2505  6413 Aug 29  2018 amm28053
-rw-rw---- 1 cs2505 cs2505  7744 Aug 29  2018 andysin.txt
-rw-rw---- 1 cs2505 cs2505  1630 Aug 29  2018 apeace.txt
-rw-rw---- 1 cs2505 cs2505 11402 Aug 29  2018 arbid17.docx
-rw-rw---- 1 cs2505 cs2505 10715 Aug 29  2018 arman1.txt
-rw-rw---- 1 cs2505 cs2505  2191 Aug 29  2018 atg115.txt
-rw-rw---- 1 cs2505 cs2505  7633 Aug 29  2018 benw94.txt
-rw-rw---- 1 cs2505 cs2505  5194 Aug 29  2018 bir1997.ascii
-rw-rw---- 1 cs2505 cs2505  4287 Aug 29  2018 bousquet.txt
. . .
-rw-rw---- 1 cs2505 cs2505  6179 Aug 29  2018 willpr13.txt
-rw-rw---- 1 cs2505 cs2505  5110 Aug 29  2018 wjenny24
-rw-rw---- 1 cs2505 cs2505  3089 Aug 29  2018 wryan8.txt
-rw-rw---- 1 cs2505 cs2505  6769 Aug 29  2018 zakkl13.txt
-rw-rw---- 1 cs2505 cs2505  7629 Aug 29  2018 zc219.txt
-rw-rw---- 1 cs2505 cs2505  4270 Aug 29  2018 zchryb.txt
```

The directory contains hundreds of files, and you do not know all the file names. You do know that most, but not all, of the file names end with the extension `".txt"` (because some of the students who submitted them failed to understand the importance of using proper extensions). As you might guess, the first part of each file name is a PID. You may assume each file name either contains no period character, or contains a single period as part of the file extension.

You may benefit from consulting the `man` page for the `ls` command.

1. **[18 points]** Write a single Linux command that could be executed to display the names of the files described.

a) All the files, if any, whose name ends with the extension `".text"`.

b) All the files, if any, whose name begins with the letter `'r'`.

c) **[Harder]** All the files, if any, whose name ends with the letter `'c'`, possibly followed by an extension.

2. **[12 points]** Suppose your current working directory is the one shown above, and that you are the user `cs2505`. For each part, give a single valid Linux command that will create the specified `tar` archive.

a) Bundle all the files in the current working directory that have the extension `".txt"` into a flat archive named `allFiles.tar`, so that the archive is in your home directory.

b) Bundle all the files in the current working directory that were submitted by students whose PID began with the characters `"li"` into a flat archive named `li_s.tar`, so that the archive is in the parent of the current directory.

3. **[12 points]** Under the same assumptions as in question 2, give a single valid Linux command that will create the specified zip archive.

a) Bundle all the files in the current working directory that have the extension `".txt"` into a flat archive named `allFiles.zip`, so that the archive is in the parent of the current working directory.

b) Bundle the current working directory into an archive named `submissions.zip`, so that the archive is in the parent of the current working directory. (This archive will not be flat, and unpacking it would create a directory named `submissions`, which would contain (copies of) all the files in the current working directory.)

4. **[12 points]** The user `cs2505` executes the following command in the parent of the working directory shown above:

```
ls --format=single-column ./submissions/*5.txt
```

The following output is produced:

```
submissions/atg115.txt
submissions/gceric5.txt
submissions/jose95.txt
submissions/leo1995.txt
submissions/nickf15.txt
submissions/seth15.txt
submissions/unicho5.txt
submissions/vinh0925.txt
```

a) Describe the output that the following command would produce:

```
ls --format=single-column ./submissions/*5.txt | wc
```

You actually have enough information to show the precise output, but you must describe it logically.

b) Describe the output that the following command would produce:

```
wc `ls --format=single-column ./submissions/*5.txt`
```

You do not have enough information to show the output, so you must describe it logically. Those funny quote marks in this command are called *backticks*, and they are important. You might want to look that up.

5. **[8 points]** The user `arbid17` has submitted a file shown in the listing given above; unfortunately, his file was not a plain ASCII text file, as required for the course. He attempts to fix the problem by resubmitting the same file, but renaming it `arbid17.txt`. Explain clearly why this will, or will not, fix the problem.

6. [10 points] Suppose a user executes the following command on a Linux system:

```
find ~ -maxdepth 2 -type f -name "*.docx"
```

What is the overall purpose of running this command, or in other words, why would a user wish to run it? Be very precise and complete. You may wish to examine the `man` page for the `find` command, and you may also want to experiment.

For question 7, suppose you are in the directory `J3Files`, which has the following contents, including a directory tree rooted in `CS3114`:

```
J3Files
|-- CS3114
|   `-- J3
|       |-- Client
|       |   `-- Point.java
|       `-- DS
|           |-- Compare2D.java
|           |-- Direction.java
|           |-- Lewis.class
|           `-- prQuadTree.java
|-- gradeJ3.sh
|-- LogComparator.jar
|-- PRGenerator.jar
`-- testDriver.java
```

7. [28 points] For each part, write a single Linux command that will achieve the specified action, with no unnecessary side-effects, assuming the command will be executed in the directory `J3Files`. The parts are independent; in other words, each will be performed with the system in the state shown above. You will find a number of `man` pages to be very useful. There will be no partial credit for these answers.

a) Copy the file `Direction.java` to the user's `bin` directory, created in the previous Linux assignment.
b) List the files in the subdirectory `DS`, sorted in ascending order by file size.
c) Copy the entire directory tree rooted in `CS3114`, including all the files therein, to the user's `bin` directory.
d) Move the directory DS, and all its contents, to the directory `J3Files`.
e) Remove the directory tree rooted in `J3`, including the directory `J3` itself.
f) Rename the directory `DS` to be called `Structures`.
g) Display the name of every Java source code file in the directory tree rooted in `CS3114`, assuming that the files are named using proper extensions.

---

[1] How can you tell if you've prepared a plain ASCII text file? The simplest way is to use a Linux text editor to create the file you are going to submit (e.g., gedit, geany, etc). Alternatively, use the `file` command in a Linux shell:

```
centOs > file Tolstoy.txt
Tolstoy.txt: ASCII text
```

It is possible you may get a different response; for example:

```
ASCII text, with very long lines
ASCII text, with CRLF line terminators
ASCII text, with very long lines, with CRLF line terminators
UTF-8 Unicode text
UTF-8 Unicode text, with very long lines
UTF-8 Unicode text, with CRLF line terminators
UTF-8 Unicode text, with very long lines, with CRLF line terminators
Non-ISO extended-ASCII text
```

```
Non-ISO extended-ASCII text, with very long lines, with CRLF line. . .
Pascal source, ASCII text
C source, ASCII text
```

Those are probably all acceptable. However:

- long lines indicate your file may be difficult to view in certain applications; try inserting line breaks around column 80 when you type your files
- CRLF line terminators indicate your file was produced on a Windows environment; that may also cause issues if your file is opened in certain applications
- Non-ISO extended ASCII text indicates you've used non-standard characters in your file; this may be due to the inclusion of garbage text you don't intend to submit; such characters may cause some applications to interpret the file incorrectly, or even to refuse to open it

On the other hand, these responses from the `file` command would indicate your file is not be acceptable:

```
Microsoft Word 2007+
PDF document, version 1.5
POSIX tar archive
gzip compressed data, from Unix. . .
```

You can also use the `cat` or `less` command to display your file to a Linux terminal window. If this displays anything other than the simple text of your answers, it's not a plain text file.

If you aren't sure, ask a TA to look at your file.