Sample solution to HW 1: Coreutils

For each of the following problems, substitute the number supplied into the following instruction:

Select ____ unique coreutil commands in the slides (not already used by the other two problems) that can be combined by piping. Construct a practical scenario where they could be applied together, and run the command on realistic input. Include a sample of input (if any), and the output after each pipe.

1. three

   Sample command:

   ```
   paste audio_cues.dat video_cues.dat | nl | csplit - "/yes.*yes/"
   ```

   Objective: Determine what times to split a video file based on when separate image and audio cue detections agree.
   Input: A text file containing the per-second detection classifications for audio and visual inputs. Audio detections:

   ```
   --------------------------
   no
   no
   yes
   no
   no
   --------------------------
   ```

   Visual detections:

   ```
   --------------------------
   no
   yes
   yes
   no
   no
   --------------------------
   ```

   After first pipe:

   ```
   --------------------------
   no no
   no yes
   yes yes
   ```

```
no no
no no
--------------------------
```

After second pipe:

```
--------------------------
     1  no      no
     2  no      yes
     3  yes     yes
     4  no      no
     5  no      no
--------------------------
```

After thirdpipe:

```
--------------------------
27
--------------------------
```

Split file `xx00`:

```
--------------------------
     1 no no
     2 no yes
--------------------------
```

Split file `xx01`:

```
--------------------------
     3 yes yes
     4 no no
     5 no no
     6 no no
...
--------------------------
```

2. four
   Sample command:

   ```
   shuf full_flashcard_commands.dat | tail -n1 | tee  answer | tr '-' '\r'
   ```

   Objective: Review abbreviated manpage definitions of Unix commands with randomly sampled commands. Answers are written to a file named `answer`.
   Input: A file containing the short definitions (i.e. `man -f`) of a collection of Unix commands. Sample input:

   ```
   --------------------------
   shuf (1)             - generate random permutations
   nl (1)               - number lines of files
   ```

```
join (1)             - join lines of two files on a common field
paste (1)            - merge lines of files
csplit (1)           - split a file into sections determined by context lines
tail (1)             - output the last part of files
fmt (1)              - simple optimal text formatter
fold (1)             - wrap each input line to fit in specified width
tr (1)               - translate or delete characters
unexpand (1)         - convert spaces to tabs
expand (1)           - convert tabs to spaces
--------------------------
```

After first pipe:

```
--------------------------
fold (1)             - wrap each input line to fit in specified width
tail (1)             - output the last part of files
paste (1)            - merge lines of files
csplit (1)           - split a file into sections determined by context lines
unexpand (1)         - convert spaces to tabs
join (1)             - join lines of two files on a common field
tr (1)               - translate or delete characters
nl (1)               - number lines of files
fmt (1)              - simple optimal text formatter
shuf (1)             - generate random permutations
expand (1)           - convert tabs to spaces
--------------------------
```

After second pipe:

```
--------------------------
expand (1)           - convert tabs to spaces
--------------------------
```

After third pipe:

```
--------------------------
expand (1)           - convert tabs to spaces
--------------------------
```

After fourth pipe:

```
--------------------------
convert tabs to spaces
--------------------------
```

In answer file:

```
--------------------------
expand (1)           - convert tabs to spaces
--------------------------
```

3. five
   Sample command:

```
cat [a-z][a-z][a-z][0-1][0-9]*.txt | sort | uniq -c | sort -r -k1 | cut -d' ' -f8 |  head -n2
```

   Objective: To determine the office hours with the highest votes
   Input: Text files of FSU ids (e.g. scm14f@my.fsu.edu) containing all of the days and hours in military time (in no particular order). Sample input:

```
--------------------------
TUESDAY,1430
FRIDAY,1400
FRIDAY,1000
MONDAY,1230
THURSDAY,1500
--------------------------
```

   After first pipe:

```
--------------------------
TUESDAY,1430
FRIDAY,1400
FRIDAY,1000
MONDAY,1230
THURSDAY,1500
WEDNESDAY,1030
TUESDAY,1600
MONDAY,1400
TUESDAY,1200
MONDAY,1300
THURSDAY,1130
MONDAY,1400
MONDAY,900
MONDAY,1000
WEDNESDAY,1330
MONDAY,930
TUESDAY,930
WEDNESDAY,1430
THURSDAY,1030
TUESDAY,1400
MONDAY,1600
THURSDAY,1130
TUESDAY,1230
THURSDAY,1530
TUESDAY,930
--------------------------
```

   After second pipe:

```
--------------------------
```

```
FRIDAY,1000
FRIDAY,1400
MONDAY,1000
MONDAY,1230
MONDAY,1300
MONDAY,1400
MONDAY,1400
MONDAY,1600
MONDAY,900
MONDAY,930
THURSDAY,1030
THURSDAY,1130
THURSDAY,1130
THURSDAY,1500
THURSDAY,1530
TUESDAY,1200
TUESDAY,1230
TUESDAY,1400
TUESDAY,1430
TUESDAY,1600
TUESDAY,930
TUESDAY,930
WEDNESDAY,1030
WEDNESDAY,1330
WEDNESDAY,1430
--------------------------
```

After third pipe:

```
--------------------------
      1 FRIDAY,1000
      1 FRIDAY,1400
      1 MONDAY,1000
      1 MONDAY,1230
      1 MONDAY,1300
      2 MONDAY,1400
      1 MONDAY,1600
      1 MONDAY,900
      1 MONDAY,930
      1 THURSDAY,1030
      2 THURSDAY,1130
      1 THURSDAY,1500
      1 THURSDAY,1530
      1 TUESDAY,1200
      1 TUESDAY,1230
      1 TUESDAY,1400
      1 TUESDAY,1430
      1 TUESDAY,1600
      2 TUESDAY,930
      1 WEDNESDAY,1030
      1 WEDNESDAY,1330
      1 WEDNESDAY,1430
--------------------------
```

After fourth pipe:

```
--------------------------
      2 TUESDAY,930
      2 THURSDAY,1130
      2 MONDAY,1400
      1 WEDNESDAY,1430
      1 WEDNESDAY,1330
      1 WEDNESDAY,1030
      1 TUESDAY,1600
      1 TUESDAY,1430
      1 TUESDAY,1400
      1 TUESDAY,1230
      1 TUESDAY,1200
      1 THURSDAY,1530
      1 THURSDAY,1500
      1 THURSDAY,1030
      1 MONDAY,930
      1 MONDAY,900
      1 MONDAY,1600
      1 MONDAY,1300
      1 MONDAY,1230
      1 MONDAY,1000
      1 FRIDAY,1400
      1 FRIDAY,1000
--------------------------
```

After fifth pipe:

```
--------------------------
TUESDAY,930
THURSDAY,1130
MONDAY,1400
WEDNESDAY,1430
WEDNESDAY,1330
WEDNESDAY,1030
TUESDAY,1600
TUESDAY,1430
TUESDAY,1400
TUESDAY,1230
TUESDAY,1200
THURSDAY,1530
THURSDAY,1500
THURSDAY,1030
MONDAY,930
MONDAY,900
MONDAY,1600
```

```
MONDAY,1300
MONDAY,1230
MONDAY,1000
FRIDAY,1400
FRIDAY,1000
--------------------------
```

After sixth pipe:

```
--------------------------
TUESDAY,930
THURSDAY,1130
--------------------------
```

Note on commands: Scenarios which are trivial or contrived may be penalized. If you wish to use more commands than the number specified, you are welcome to do so if no commands are repeated between questions. A command with a different flag is the same command (the count will be unchanged). Any flags not covered in class are fair game, but the use of coreutil commands not covered in the slides is prohibited. Partial credit will be given.

Trivial example:
`touch myfile |wc`
Problem: `touch` doesn't even print to standard out in this case.

Contrived example:
`md5sum rogue.wav | cut -f1 -d' '|fold -w1 |sort | uniq`
Problem: When would sorting a file's md5sum characters be useful?

Consider that you have a lot of flexibility in the content, structure, and volume of input you might use. For example:

- (Sequentially-named) photographs,
- plaintext sheet music,
- vocabulary words
- stock prices over time,
- a list of upcoming events,
- etc.