EECS2031

Lab 2

Summer 2016

Lab Objectives

In this lab, you will be introduced to more complex Unix commands. After this lab, you should be comfortable using Unix/Linux in the lab and as a platform for software development.

File names generation

When you enter file name for a command (cat filename) there is a way of generating a list of files according to a specific pattern. Some characters or patterns have a special meaning to the shell, we can use them to generate this list.

char/pattern	Description
*	Matches any string of characters including the null string but not a leading dot
?	Matches any single character
[abcf]	Matches any one of the characters in the list
[a-x]	Matches any characters between a to x (inclusive)
[^abc]	Matches any character other than a,b,c (matches any character not in the set)

So, for example

```
ls test*
```

Lists files that start with test (test1, test.txt, test.doc, test_1, testfinal, …. including a file named test if it exists)

```
ls test?1
```

Lists files such as test_1 testa1, test21, but not test1

```
ls test[ab].txt
```

Lists testa.txt, testb.txt but not textc.txt

```
ls test[a-d].txt
```

Lists testa.txt, testb.txt, testc.txt, and testd.txt if they exist in the current directory.

Try echo * what does it do?
If a special character (such as *) is quoted using \ as * looses its special meaning and acts like any other characters, for example

`ls test\*` lists only the file “test*” if it exists. Also \ is used to escape end of line character so it does not mean the end of the command, for example

```
tigger 193 % echo hi this is\
? a multi line echoing
hi this is a multi line echoing
tigger 194 %
```

sort

The command sort sorts the input file(s) according to some key. The input stream is treated as independent records of variable length delimited by new lines. Each record is treated as fields delimited by whitespaces or user specified single character. For a complete list of options, man sort. Some of the options are mentioned below.

The two most popular options are -t and -k
-t specifies the character that delimits fields, for example -t: specifies that fields are terminated by the character “:”white spaces are part of the field. Without specifying -t, fields are separated by white spaces and leading and trailing white spaces are ignored. Note the difference between using -t' ' and not specifying –t at all. In the first case, a record with space followed by A followed by space is a record with three fields, while not specifying -t interprets it as a single filed record (A).

The other popular option is -k where the field number is specified, if you use -k4 that means sort based on the 4th field.

Consider the file file1

```
sh-3.00$ cat file1
this is line 1
zsomething starts with 2
this is line 2
something starts with 2
this is happy line
this is line 21
this is line 200
this is line -5
line without numbers
sh-3.00$
```

Now if we use the default sort. The output is as follows

```
sh-3.00$ sort file1
line without numbers
something starts with 2
this is happy line
this is line -5
this is line 1
this is line 2
this is line 200
this is line 21
zsomething starts with 2
sh-3.00$
```

Here the sorting based on field 1, if tie use field 2 and so on.

```
sh-3.00$ sort -k4 file1
line without numbers
this is line -5
this is line 1
something starts with 2
this is line 2
zsomething starts with 2
this is line 200
this is line 21
this is happy line
sh-3.00$
```

Here the sorting based on the fourth field as a key the first line has no 4th field, i.e. empty key that comes before anything. Note that 200 comes before 21 since 200 is treated like a string, where 0 comes before 1. If we want to treat the field as integer, use the -n flag

```
sh-3.00$ sort -k4 -n file1
this is line -5
line without numbers
this is happy line
this is line 1
something starts with 2
this is line 2
zsomething starts with 2
this is line 21
this is line 200
sh-3.00$
```

The format -kn means the key starts at the nth field and continue to the end of the record. If the format is -kn,m the key starts at the beginning of the nth field and ends at the end of the mth field. If the format is -k2.3,4.1 it means the key starts at the 3rd character of the second field and ends at the first character of the 4th field.

uniq

Uniq removes duplicate records from data stream. Usually it is used with sort to sort the file then remove duplicate records. `sort -u` removes records with duplicate keys.

Example: `sort file |uniq`
Sorts the named file, then the output is pipelined to uniq to remove duplicate records.

cut

The cut command is used to extract a portion of the file. It can extract based on either fields or character positions. For example, `cut -c1-3 file`
Displays character 1,2, and 3 from each record.
While cut -f1-3 file
Displays fields 1,2, and 3 from each record

The default delimiter is a tab, but you can change it with the -d option.

tr

The command tr translates characters from a file.

Tr [options] string1 [string2]
Characters in string1 is translated into characters in string 2 character by character.

join

The join command merge records in **sorted** files based on a common key, for example, if we have two files, suers and susers

```
sh-3.00$ cat suers
cat: suers: No such file or directory
sh-3.00$ cat susers
jdoe John Doe 4/15/96
jhsu jack Hsu 1/2/93
lsmith laura Smith 3/12/96
pchen paul Chen 1/5/97
que extra field
sphilip Sue Philip 4/4/94
sh-3.00$ cat suers
jdoe John Doe 4/15/96 External
jhsu Jack Hsu 1/2/93 Internal
lsmisth Laura Smith 3/12/96 Internal
```

```
pchen Paul Chen 1/5/97 EX
sphilip Susan Philip 4/4/94 Internal
sh-3.00$ join -1 1 -2 1 susers susers
jdoe John Doe 4/15/96 John Doe 4/15/96 External
jhsu jack Hsu 1/2/93 jack Hsu 1/2/93 Internal
lsmith laura Smith 3/12/96 laura Smith 3/12/96 Internal
pchen paul Chen 1/5/97 paul Chen 1/5/97 EX
sphilip Sue Philip 4/4/94 Sue Philip 4/4/94 Internal
sh-3.00$
```

where join -1 1 -2 1 susers suers means join the two names files based on the first field of the first file (-1 1) and the first field of the second file (-2 1)
For complete details, look up `man join`

What happens if the files are not sorted?

C Code

Practice

Write an ANSI C program to do the following:
The program reads from the standard input a number of records, each record in a line. The record consists of the following fields

- The team name: a sequence of characters (letters, numerals and _) with a maximum of 30 characters
- The number of games won 1-99
- The number of games tied 1-99
- The number of games lost 1-99
- Streak, a positive streak indicates a winning streak, a negative streak indicates a losing streak. (read from the standard input until end of file)

Fields in the input are separated by white spaces (spaces or tabs). The program should read the data and display a list with the team name, points earned, and streak. The points are calculated as 3 point for win, 1 point for tie, and 0 for loss.

For example, one output should look like

Renegades\t□35\t□+5 where \t means a tab and □ is a space. Do not output the two characters \ (”slash”) and t, output a tab.

The number of points are to be written in 3 digits (right justified) and the streak is a sign follows by a number in 3 digits (right justified). The sign and the number together in 3 spaces

Right justified means proceeded by spaces to the left, so for example 23 in 5 spaces right justified is □□□23

Each record should end with a newline (including the last one).

After you display the standings, then an empty line followed by two lines as follows

The max points by any team is then followed by the maximum number of points by any team in 5 spaces right justified.

The longest winning streak is followed by the maximum winning streak in 5 spaces right justified. If there is no team with a winning streak, the streak should be 0.

The program should be able to deal with the following cases.

If the number of games (won, lost, or tied) is -ve, the program should output the name of the team, followed by a tab, then the string “negative number of games” followed by a new line. Else if the total number of games played is more than 99, the program should output the name of the team as above, followed by a tab, followed by “games played are more than 99” followed by a newline.

Sample input:

```
Renegades 22 10 1 3
Pintos          15 18 2 -2
Mississauga_1   -2 119 3 4
```

Sample output:

```
Renegades  76    +3
Pintos     63    -2
Mississauga_1   negative number of games

The maximum points by any team is   76
The longest winning streak is    3
```