CSE2031

Lab 2
Winter 2017

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 name 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, these characters/patterns are [much more on this later]

Char/Pattern	Description
*	Matches any string of characters including th null string but not a leading dot
?	Matches any single character
[abcf]	Matches any character in the list
[a-x]	Matches any character between a to x (inclusive)
[^abc]	Matches any character **not** in the set

So for example

```
ls test*
```

Lists files that starts with test (test1, test.txt, test.doc, test_1, testfinal, …. including a file names test if they exist.)

```
ls test?1
```

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

The command

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

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

The command

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

Lists testa.txt, testb.txt, testc.txt, and testd.txt if they exist in the3 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
On my machine

```
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 here in the examples below.
The two most popular options are -t and -k
-t specifies the character that delimits fields, for example -t: specifies that field 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$
```

Noe 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 filed 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 ket starts at the 3rd character of the second field and ends at the first character of the 4th field.

unique

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

join

he join command merge records in **sorted** files based on a common key
for example
if we have two files, suers and susers1

```
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 suers1
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 susers1
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 suers1 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?

There are no deliverable for this part. However, that, for sure, will be covered in the first labtest

C code

Specifications

Write a C program called invest.c
The input to the program is a sequence of share prices each line contains the closing price of the share every day. The price is in dollars and cents (floating point number with 2 digits after the decimal point).
The input is read from the standard input (keyboard) one entry per line. You should continue reading till the end of file. The end of file in Linux is CTRL-D at the beginning of a line
Your program starts with $10,000 to invest. You buy and sell shares according to some rule. At the end you calculate how much money did you gain/loose.

You decide to buy or sell based on the price of the stock. You buy or sell at the beginning of a day based on past performance. The rules is as follows:

- You buy and sell only integer number of shares.
- You have to limit your buy to the available cash you have (no credit)
- If the price of the share increased in 2 consecutive days by 10% or more, sell every thing.
- Else, if the price increased by more than 5% for 2 consecutive days, buy with half the money left in the account.
- If the price decreased for 2 consecutive days, buy as much as you can.

Keep in mind that the price represents the closing price at the end of the day, you buy or sell at the beginning of the day. Your decision should be based on previous performance and price. I am assuming here that the price at the beginning of the day is exactly the same as the closing price of the previous day.

Assume that there was no change in price for the first entry (the first entry is the same as the one before it).

After you process the file (EOF) then display your gain or loss as follows

Gain/Loss: followed by the number with 2 decimal points left justified with "$" before the number (to the left of the number) followed by a new line.

Check the test case for details, use the "od" to see what is in the file

Submit as submit 2031 L2 invest.c

Exercise

Here are some more exercises, do not submit them. They definitely will be of help in lab tests and the final exam:

Binary adder:
the input is three strings, the first and third are binary strings of size 32 max. The second string is either "+" or "-".
The first and third strings are binary representation of an integer number. Your program will perform addition or subtraction in binary and display the result in binary. Assume there is no overflow or underflow.

For example, the inputs are
01010010
+
00110101
____

1000111
Zeros to the left may be omitted, for example
11010
+
11
that means
11010
+
00011

11101

When you display the answer, display only the non zero bits followed by a newline.

Two's complement

The program reads a string that represent a binary number and displays the two's complement.
The input could be up to 32-bit number (a string of up to 32 characters of 1's and 0's only similar to the input in the previous exercise).
Display the number as 32-bit number, or you can omit the 0's or 1's to the left (you must have at least one 0, or 1 as the leftmost digit in order to differentiate between positive and negative numbers.

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 that
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 ×××📋🗑
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.