Faculty of Computer Science

DALHOUSIE UNIVERSITY
Inspiring Minds

CSCI 2132
Midterm 1 Solutions

Term: Fall 2018 (Sep4-Dec4)

1. (10 points) True-false questions: 2 points each. Justification is not necessary, but brief justification may be helpful if correct.

a) (2 points) When we use PuTTY to login to a server, the OS kernel responds with a prompt string and waits for our command.

> **Solution: False.** It is not the kernel, but the shell program responds with a prompt string.

b) (2 points) Given a textual file `a.txt` The following command is an example of a pipeline command: `a.txt > sort > b.txt`

> **Solution: False.** It is false for two reasons: (1) only one command is included and a pipline requires at least two, and (2) the syntax is wrong.

c) (2 points) If a file has the following permissions `------r--`, the owner of the file cannot read it. (The order of permissions is the standard as in the `ls -l` command.)

> **Solution: True.** The owner does not have the read permission, even though others (users other than the owner and not in the owner group) have this permission.

d) (2 points) Two different directories cannot share the same inode number.

> **Solution: True.** Directories cannot be hardlinked, so they cannot share inode numbers.

e) (2 points) If we use the C statement `scanf("%d", &n);` and enter the word "`zero`" in the input, an exception is thrown.

> **Solution: False.** C does not have exceptions. The function scanf will not convert the number but it will not throw an exception or stop the program.

2. (12 points) Multiple-choice. No justification necessary. Circle the *single* best answer.

a) (3 points) If we run the following three commands:
'`chmod 743 a.txt`', '`chmod 632 a.txt`', and '`chmod a+x a.txt`'
then the final permissions of the file '`a.txt`' will be as follows:

A. `rw--wx-wx`

B. `rwx-wx-wx`

C. `rw--wx-x-`

D. `rw--w--w-`

Solution: **B.** `rwx-wx-wx`
After the first command, the permissions are 111.010.011 (i.e., `rwx-w--wx`), after the second command they are changed to 110.011.010 (`rw--wx-w-`), and the third command will set all execute permissions on, so we get 111.011.011 (`rwx-wx-wx`), which is the same as B.

---

b) (3 points) After executing the following two C statements:
'`x = 3.0; n = scanf("%lf", &x);`' on the following input '` -.01e1 2.2`', the values of `int` variable `n` and `double` variable `x` are:

A. $n = 0$ and $x = 3.0$

B. $n = 1$ and $x = 3.0$

C. $n = 0$ and $x = -0.01$

D. $n = 1$ and $x = -0.1$

Solution: **D.** The string '` -.01e1` is successfully converted to $x = -0.1$ and the returned value is 1 due to one successful conversion.

---

c) (3 points) A file named `courses.csv` contains courses taken by a group of students, where each line has a format: *student name*, *course number*, *term*. An example of a line from such file would be:
`John Smith, CSCI 2132, Fall 2018`
Which command will print the number of different courses in the file?

A. `cut -d "," -f 2 < courses.csv | sort | uniq | wc -l`

B. `sort < courses.csv | cut -d "," -f 4 | uniq | wc -l`

C. `cut -d "," -f 2 < courses.csv | wc -l | uniq | sort`

D. `cut -d "," -f 4 < courses.csv | sort | wc -l | uniq`

Solution: **A.** We need to first get only the courses, then sort them, use uniq to remove duplicates, and finaly report the number of different courses. "B." may count duplicates, "C." and "D." will just print the number of the original lines.

---

d) (3 points) You created a new file in your working copy, and you want to save this file in the SVN repository. The commands that you need to use for this task are:

A. `svn checkout` and `svn commit`

B. `svn add` and `svn commit`

C. `svn update` and `svn commit`

D. `svn resolved` and `svn commit`

Solution: **B.** We have already a working copy so we need to add the file using the command `svn add` and save it to the repository using the command `svn commit`. "A." — no need to checkout since we have a working copy, "C." and "D." — we did not add the file, and "svn resolved" is not needed since we are not aware of any conflict.

3. (11 points) Give concise answers.

a) (3 points) Give example of an absolute and a relative pathname. (Label which one is absolute and which one is relative.)

Solution: Example of absolute path: `/home/csci2132`

Example of a relative path: `./a2`

An absolute path starts with '`/`' and a relative path not. The above examples are also two logical valid paths.

Using a path that starts with '`~`' is not acceptable since it is not clear how to classify it. Namely, since the first symbol is not '`/`' it seems like a relative path, but actually the shell will make a tilde expansion on it and replace it with an absolute path, so it is a shorthand for an absolute path in practice.

b) (4 points) When do we use wildcards and when regular expressions? Show by example a difference between their notations.

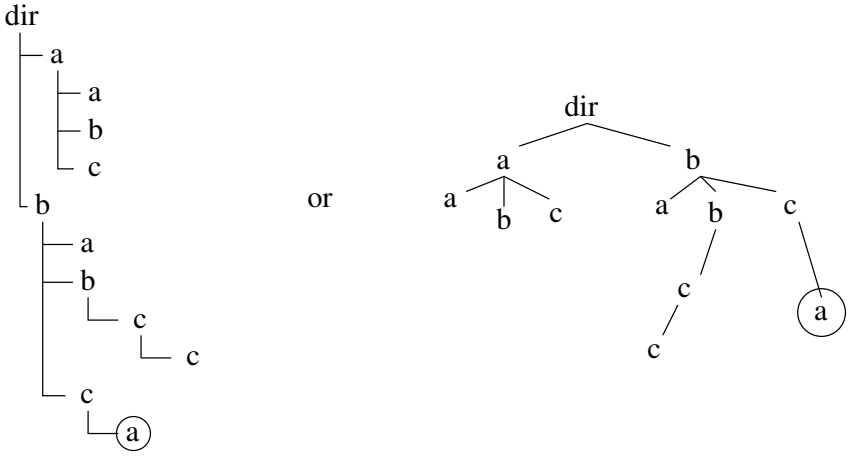
Solution: Whildcards, also known as filename substitutions or pathname substitutions, are used in shell to include a number of filenames in the system by using a pattern. Regular expressions are used by grep and similar programs to match textual lines in a file or standard input.

An example of difference is that in wildcards '`*`' is used to match an arbitrary string, and in regular expressions we use '`.*`' for the same purpose.

c) (4 points) Draw the directory structure created by the following commands and circle our current directory at the end.

```
mkdir dir ; cd dir
mkdir -p  a/a  a/b  a/c  b/a  b/b  b/c/a
cd b/c/a ; mkdir -p ../../b/c/c;  cd ../a
```

Solution: The created directory structure is as follows (this can be drawn as a tree as well):



4. (12 points) Command line. For each of the following questions write a single command line to perform the required task. You do not need to justify the answer. You are allowed to use only the following commands, with possible pipes: `cat`, `cut`, `echo`, `grep`, `egrep`, `ls`, `sort`, `uniq`, and `wc`.

a) (4 points) Print out a single integer that is the number of files in the directory `/home/joe/etc` whose names start with a lowercase letter, does not end with 'c' or 'd' or a digit, and are five characters long. An example of such filename would be `/home/joe/etc/abcdF`
Note that you are asked to print the number of such files.

Solution: Both lines below are correct:

```
ls /home/joe/etc/[a-z]???[!cd0-9] | wc -l
ls /home/joe/etc/[a-z]???[^cd0-9] | wc -l
```

Not as good solution, but also accepted:

```
ls /home/joe/etc | grep '^[a-z]...[^cd0-9]$' | wc -l
```

b) (4 points) Print out all lines from a file named `file.txt`, that contain the word 'printf' or 'scanf', and also contain the word 'return'. Make sure to match exact words, for example 'returning' should not be matched with 'return'.

Solution:

```
egrep -w '(printf|scanf)' file.txt | grep -w 'return'
```

c) (4 points) You are given a file named `/etc/passwd`, which contains a list in the following format:

```
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
user1:x:1000:1000:John Doe:/home/user1:/bin/tcsh
```

Each line contains seven fields separated by colon (':'). Write a command to sort the lines of the file by the fifth (5th) field, and then print only the first five fields of each line.

Solution:

```
sort -t: -k5 /etc/passwd | cut -d: -f1,2,3,4,5
```

or

```
sort -t: -k5 /etc/passwd | cut -d: -f1-5
```

5. (10 points) C Program.

(10 points) Given three positive numbers a_1, a_2, and a_3, the arithmetic mean (AM) and harmonic mean (HM) of these numbers are defined as:

$$AM = \frac{a_1 + a_2 + a_3}{3}, \qquad HM = \frac{3}{\frac{1}{a_1} + \frac{1}{a_2} + \frac{1}{a_3}}$$

Write a C program that asks a user to enter three positive floating-point numbers, separated by semi-colons. The program then computes the arithmetic mean and harmonic mean of these three numbers, and prints the results as floating-point numbers, keeping **4 digits after the decimal point**.

You can make your own (reasonable) assumptions about how the program handles the input and output, as long as your program will work if the user enters the input values using the format '`a1 ; a2 ; a3`', e.g. '`0.5 ; 2.5 ; 7`'. You can notice that user will use semi-colon between numbers and possibly some spaces.

You can assume that the user always correctly enters the positive numbers in the required format. No error handling is required. You are only allowed to use variables whose types are `float` or `double` in your program.

Solution:

```
#include <stdio.h>

int main() {
  float num1, num2, num3;

  printf("Enter three numbers, separated by commas\n");
  scanf("%f ; %f ; %f", &num1, &num2, &num3);

  printf("The Arithmetic mean is %.4f\n", (num1+num2+num3) / 3);
  printf("The Harmonic mean is %.4\n", 3.0/(1/num1+1/num2+1/num3));
  return 0;
}
```

Note: Due to a typo in the original question, it is accepted if students used comma ('`,`') instead of semi-colon.

Tenative marking scheme:
1 mark for include
1 mark for correct start of main
1 mark for variable definitions
2 marks for using scanf correctly
1 mark for computing arithmetic mean
1.5 mark for printing arithmetic mean
1 mark for computing harmonic mean
1.5 mark for printing harmonic mean