University of Toronto
CSC209H: Software Tools and Systems Programming
Spring 2001

Midterm

Duration:	50 minutes
Aids Allowed:	None

Make sure that your test has 6 pages (including this one). Write your answers in the spaces provided. You will be rewarded for concise, well-thought-out answers, rather than long ones. Write legibly.

Name:	Solutions	**Section:**	______
Student #:	______	**TA:**	______

1.	____ /	7
2.	____ /	4
3.	____ /	4
4.	____ /	5
5.	____ /	6
6.	____ /	8
TOTAL:	____ /	34

Question 1 [8 marks] (7 minutes)

a. [1] During the guest lecture, Greg Wilson claimed that he was a better programmer than you, not because he was smarter, but because of the kind of programs he keeps in his own `bin` directory. Explain what Greg has in his `bin` directory and how it fits with the Unix philosophy.

Many small programs (utilities, tools, scripts). It fits with the Unix philosophy of writing programs that do one thing well.

b. [1] How does the system know which shell to start as your default shell? Be specific.

The system examines a field in your entry of the /etc/password file.

c. [1] Briefly explain what a system call is.

System calls are the interface to the operating system. They are the mechanism that we use to call system services.

d. [1] Explain what happens with the I/O redirection in the following command:

```
myProg >& aFile
```

Both stdout and stderr are redirected to the file aFile.

e. [2] Explain how to kill a process that is running in the background.

1) Find its PID using ps or top and then use "kill" on the PID.
2) If the process is under job control, use "jobs" to determine the job number and `kill %<jobnumber>`.

f. [1] The command `date` prints out the date and time as "`Sun Feb 25 22:12:08 EST 2001`". Using `date`, write 2 lines of Bourne shell code that will print the time and day in the format "`22:12:08 Sun`"

```
set `date`
echo $4 $1
```

g. [1] Suppose the current working directory is a directory containing many mail messages, each in a separate file. I want to find all the messages that are about the midterm. Give the Unix command that displays all lines containing the word "midterm" from all the files in the current directory.

```
grep midterm *
```

Too many students did `cat * | grep midterm`. While technically correct, it is a redundant use of "cat" and requires starting an extra process unnecessarily.

Question 2 [4 marks] (4-5 minutes)

Given the following information:

```
> whoami                 # what is my user name
reid
> ls -l
dr-xr-xr-x   2 reid          512 Feb 22 16:13 fruit/
> ls -l fruit
-rw-rw-rw-   1 chen           10 Feb 22 16:13 apple
-rw-rw-rw-   1 reid           50 Feb 22 16:13 banana
d--x--x--x   2 reid          512 Feb 22 16:58 citrus/

# citrus contains a file called orange that is executable by all
```

Which of the following commands will succeed? Circle YES if the command will succeed and NO if the command will produce an error due to improper permissions. Assume none of the commands have been aliased.

`rm fruit/banana`	YES	[✓ NO]	No write permissions on fruit.
`cat fruit/apple`	[✓ YES]	NO	Readable by all.
`ls fruit/citrus`	YES	[✓ NO]	No read permission on citrus.
`fruit/citrus/orange`	[✓ YES]	NO	Execute permission allows search

Question 3 [4 marks] (5 minutes)

What is the output of the following set of Bourne shell commands, given that the current working directory contains the following? If the output of the command would display the contents of a file, say that, and be sure to state which file or files would be displayed. (' is a forward quote, and ` is backquote.)

```
> ls
file1.ps   file2.ps   file1.pdf  file2.pdf
```

```
    prog=cat
a)  echo "$prog *.ps"  cat *.ps

b)  echo $prog *.ps    cat file1.ps file2.ps

c)  echo '$prog *.ps'  $prog *.ps

d)  echo `$prog *.ps`  displays contents of file1.ps and file2.ps
```

Question 4 [5 marks] (5-6 minutes)

```
1  int main()
2  {
3      int pid1, pid2;

4      printf("start\n");

5      pid1 = fork();
6      if(pid1 == 0) {
7         printf("hi\n");
8         pid2 = fork();
9         printf("hi\n");
10     } else {
11        printf("ho\n");
12     }
13 }
```

a. [1] How many processes are created when this program is run? Include the process that first calls fork(). 3

b. [2] How many times are each of the strings in the above code printed? Fill in the blanks.

"start" = 1 "hi" = 3 "ho" = 1

c. [1] Sometimes when this program is run at the command line the shell prompt appears before all of the output is displayed. Briefly explain why this may happen.

The parent may finish before the child. When the program is run in the foreground, the prompt appears when the parent terminates. If the program is run in the background, the prompt returns immediately, even before the parent has finished.

d. [1] What line of code could you add to the program to guarantee that all output would appear before the shell prompt appears? Also, give the line number after which your line of code should be inserted.

Put `wait()` after lines 5, 10, 11 or 12. To be completely correct, we should also put a `wait()` after line 8 or line 9.

Question 5 [6 marks] (12 minutes)

An experimenter frequently runs the following set of programs:

- `makegraph` takes a file name as an argument and writes to stdout. It is used to transform the a data set from the file into a format that can be interpreted by `jgraph`.
- `jgraph` reads from stdin and uses the input to render a postscript graph. It writes its output to stdout.

Write a Bourne shell script that runs `makegraph` and then `jgraph` (whose input is the output of `makegraph`) on all files in the current directory. The final output of `jgraph` should be sent to a file with the same name as the data file, but with the suffix "`.ps`". For example, if the data file file was named `out.1.30` then the final output of `jgraph` will be sent to a file named `out.1.30.ps`. You may assume that you do not need to give the path for any of the programs. You may create extra files.

```
for file in *
do
    makegraph $file > $file.tmp
    jgraph < $file.tmp > $file.ps
done

OR

for file in *
do
    makegraph $file | jgraph > $file.ps
done
```

Question 6 [8 marks] (15 minutes)

Complete the Perl program started on the next page. The program takes two arguments: a word to search for in the "Subject" lines of a news archive, and the filename of the file containing a news archive. Each line of output is composed of a subject line containing the given word followed by the contents of the "From" line for the same message. The output must not contain the label of of each line (i.e., "`~From`" or "`~Subject`").
For example, suppose the file "NASAnews" contains a messages with headers that look like

```
~Xref:
~From: Julie Payette <payette@nasa.gov>
~Subject: International Space Station
```

If we run the program with the two arguments "Space NASAnews", the output would be:

```
International Space Station Julie Payette <payette@nasa.gov>
```

NOTE: you do not need to pattern match on all types of email address lines, and you may not assume that the "From" line appears before the "Subject" line.

The characters that need to be escaped with a backslash in Perl pattern matching are

```
\  |  (  )  [  {  ^  $  *  +  ?  .
```

```
#!/usr/bin/perl -w
use strict;

my $word =  $ARGV[0]     ;
my $filename =  $ARGV[1] ;
my $line;

my $count = 0;
my $subject;
my @subjects;
my @froms;

open(NEWS, "<$filename") || die "Couldn't open $filename\n";

while($line = <NEWS>) {
    chomp($line);
    if($line =~ /^~Xref/) {
       $count++;

    }

  if( $line =~ /^Subject:(.*)$/) {
    $subjects[$count] = $1;
    print "$count subject\n";
  }
  if( $line =~ /^From:(.*)$/) {
    $froms[$count] = $1;
    print "$count from\n";
  }
}

for(my $i = 1; $i < @subjects; $i++) {
  if( $subjects[$i] =~ /$word/) {
    print "$subjects[$i] $froms[$i]\n";
  }
}
```