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. Budget your time carefully, so that you will have enough time for the last two questions.

Name: SOLUTIONS **Section:** _____

Student #: _____ **TA:** _____

1.	_____ /	7
2.	_____ /	6
3.	_____ /	5
4.	_____ /	8
5.	_____ /	10
TOTAL:	_____ /	36

Question 1 [7 marks] (8 minutes)

a. [1] State one of the three ideas that define the Unix philosophy.

One of the following three answers:

- Write programs that do one thing and do it well.
- Write programs to work together.
- Write programs that handle text streams, because that is a universal interface.

b. [1] Give the absolute path to where the password information is stored on a Unix system.

`/etc/passwd`

c. [1] What is the difference between a process and a program?

A process is a running instance of a program

d. [1] What is `errno` and how is it used?

errno is a global variable used to record the error conditions of system calls.

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

```
myProg >!  aFile
```

Standard output is redirected to aFile. The ! means that aFile will be overwritten even if the "noclobber" environment variable is set.

f. [1] Explain the difference between the following two measurements of the size of the directory `maildir`:

```
1> ls -s                # size units are in kilobytes
4 maildir/

2> du -s maildir        # size units are in kilobytes
272     maildir
```

ls -s measure the size of the directory file maildir
du -s measures the sum of the sizes of the contents of maildir

g. [1] Write a Unix command or pipeline of commands that gives a count of the number of lines on which "csc209" appears in all files in the current directory. Do not use "cat".

`grep csc209 * | wc -l`
Too many students want to do `"cat * | grep"` or `"ls * | grep"`. It is redundant and requires starting another process.

Question 2 [6 marks] (6-7 minutes)

What is the output of the following set of Bourne shell commands, given that the current working directory contains the following? (' is a forward quote, and ` is backquote.)

```
-rw-r--r--    1 karen           4 Feb 25 12:46 .hidden
-rw-r--r--    1 karen           2 Feb 23 11:58 a
-rw-r--r--    1 karen        1000 Feb 23 11:58 b
-rwxr--r--    1 karen          91 Feb 25 12:44 x.c
```

a) `echo "ls *"` = `ls *`

b) `echo ‘ls *‘` = `a b x.c`

c) `echo *` = `a b x.c`

d) `echo ls *` = `ls a b x.c`

e) `echo 'ls *'` = `ls *`

f)
```
var='ls *'
echo $var    = ls a b x.c
```

Question 3 [5 marks] (5-6 minutes)

Given two versions of the following C program:

Version 1

```
int main()
{
    int pid;
    printf("process\n");

    pid = fork();

    if( pid == 0 ) {
        printf("child\n");
    } else {
        printf("parent\n");
    }
    printf("end\n");
}
```

Version 2

```
int main()
{
    int pid;
    printf("process\n");

    pid = fork();

    if( pid == 0 ) {
        printf("child\n");
    } else {
        wait();         /* DIFFERENT LINE*/
        printf("parent\n");
    }
    printf("end\n");
}
```

a. [1] How many times will "end" be printed in each version of the program?

Version 1: 2 Version 2: 2

b. [2] Is the order of the output in Version 1 guaranteed to be displayed in the same order every time? Circle the correct answer. If **yes**, give the output of the program in the correct order. If **no**, give two different correct orderings of the output.

```
         __          process     process   (other orderings are possible)
YES     |NO|         child       parent
         --          end         child
                     parent      end
                     end         end
```

c. [2] Is the order of the output in Version 2 guaranteed to be displayed in the same order every time? Circle the correct answer. If **yes**, give the output of the program in the correct order. If **no**, give two different correct orderings of the output.

```
 ___             process
|YES|     NO     child
 ---             end
                 parent
                 end
```

Question 4 [8 marks] (15 minutes)

One way to combine several files into a single file is to use `cat`. However, in the new file, it can be difficult to tell where one file ends and the next begins. Write a Bourne shell program that will print to `stdout` all files ending in ".h" and ".c". Before each file is printed, one line containing the name of the file, and another line containing "`------`" (several dashes) should be printed. After each file is printed, a blank line should be printed.

```
for file in *.[ch]
do
   echo $file
   echo "------"
   cat $file
   echo "
done
```

Question 5 [10 marks] (15 minutes)

Complete the Perl program below that will merge two files: an account list containing user names and real names, and a class list containing student numbers and real names. Your program will write to standard output one line for each student containing the user name, student number, and real name. The order of the lines of output does not matter. The two files will be given as arguments to the program.

Example input and output:

Account list	Class list	Output
a209reid Karen L. Reid	123465289 Karen L. Reid	a209reid 123465289 Karen L. Reid
a228hort Diane Horton	872562182 Diane Horton	a228hort 872562182 Diane Horton
fpitt Francois Pitt	111222333 Francois Pitt	fpitt 111222333 Francois Pitt

a. [2] What assumptions do you need to make about the structure and the contents of the input files to ensure your program works correctly? (Assumptions about whitespace do not count.) Hint: do part b first.

The primary assumption is that the same names must appear in both files. If a name exists in the Account list then it must also exist in the Class list and vice versa. It was okay to assume that the files were sorted by name.

b. [8] Complete the program. Use the back of this page if you need more space.

```
#!/local/bin/perl -w
# Usage: merge <account file> <class list>
use strict;

my $accounts =  $ARGV[0]      ;  # get the arguments
my $classlist =  $ARGV[1]     ;
my $line;

# more variables
my %accts;
my %nums;


open ACCT, "<$accounts" || die "Couldn't open $accounts\n";
while($line = <ACCT> ) {
  chomp($line);
  if($line =~ /^(\w+)\s+(.*)\s*$/ ) {
    my $name = $2;
    my $acct = $1;
    $accts{$name} = $acct;


  }

}
close ACCT;
open CLASS, "<$classlist" || die "Couldn't open $classlist\n";
while($line = <CLASS> ) {
  chomp($line);
  if($line =~ /^(\d+)\s+(.*)\s*$/ ){
    my $name = $2;
    my $num = $1;
    $nums{$name} = $num;


  }
}
close CLASS;

foreach my $key (keys %nums)                  {
    print "$accts{$key} $nums{$key} $key\n";

}
```