Duration: **50 minutes**
Aids Allowed: **1 - 8.5x11 sheet**

Student Number: _____

Last Name: SOLUTION

First Name: _____

TA: _____ **Instructor:** Reid

---

*Do **not** turn this page until you have received the signal to start.*
(In the meantime, please fill out the identification section above, and read the instructions below *carefully*.)

This midterm test consists of 4 questions on 6 pages (including this one). *When you receive the signal to start, please make sure that your copy of the test is complete.* Extra space was left for each of the programming questions. Please indicate clearly the part of your work that should be marked.

IMPORTANT: You do not need to include the "#!" line in Bourne shell programs you are asked to write.

MARKING GUIDE

1: _____/ 4

2: _____/ 6

3: _____/ 9

4: _____/ 9

TOTAL: _____/28

Good Luck!

Question 1. [4 MARKS]

The current working directory contains the following files and directory. Recall that `touch` creates an empty file

```
ls -l .
total 24
-rw-r--r--  1 reid  staff   39  8 Oct 00:29 file1
-rw-r--r--  1 reid  staff   39  8 Oct 00:29 file2
drwxr-xr-x  3 reid  staff  102  8 Oct 00:31 subdir/

ls -l subdir/
total 8
-rw-r--r--  1 reid  staff  42  8 Oct 00:31 file4
```

In each of the following cases, write the command to change the permissions of the appropriate file or directory so that the behavior matches the output. (The $ is the shell prompt.) If it is not necessary to change the permissions, then write, "No change".

Part (a) [1 MARK]

```
$ file1
/bin/sh: ./file1: Permission denied
$ cat file1
cat: file1: Permission denied
```

```
chmod a-r file1
```

Part (b) [1 MARK]

```
$ file2
-bash: ./file2: Permission denied
$ cat file2
#!/bin/sh
echo this is a shell program
```

```
No change
```

Part (c) [2 MARKS]

```
$ ls subdir
ls: subdir: Permission denied
$ touch subdir/file5
touch: subdir/file5: Permission denied
$ subdir/file4
This is a shell program
```

```
chmod a-rw subdir
chmod a+x subdir
chmod a+x subdir/file4
```

Question 2. [6 MARKS]

The program `du -k` displays the size of each file argument in kilobytes. If `du -k` is run with no arguments, it prints the size of the current working directory.

The current working directory contains three files with the following sizes:

- violin 60
- viola 2
- cello 20

So `du -k` would print "82 .", and `du violin` would print "60 violin"
Write the output of the following sets of commands (' is a single quote, and ‘ is a backquote)

`cmd=‘du -k‘` `echo "$cmd"`	`82 .`
`cmd="du -k"` `echo "$cmd"`	`du -k`
`cmd="du -k"` `echo '$cmd'`	`$cmd`
`cmd="du -k"` `echo $cmd *`	`du -k cello viola violin`
`cmd="du -k"` `echo ‘$cmd *‘`	`20 cello 2 viola 60 violin`
`echo *io*`	`viola violin`

Question 3. [9 MARKS]

Recall that the output of `ps aux` looks like this:

```
ange      8001  0.4  0.1  29600 21360 pts/15   S+   10:13   0:14 pine
root      8620  0.0  0.0   8160  2668 ?        Ss   10:33   0:00 sshd: pgries [p
pgries    8623  0.3  0.0   8444  1776 ?        S    10:33   0:07 sshd: pgries@pt
pgries    8624  0.0  0.0   4764  3108 pts/17   Ss+  10:33   0:00 -bash
```

Part (a) [1 MARK]

Construct a single pipeline of programs that I could use to produce a list of the users currently running processes on my machine, in alphabetical order. If a user is running 3 processes, then that user's id will appear 3 times.

ps aux — cut -f 1 -d " " — sort

Part (b) [8 MARKS]

Write a Bourne shell program prints out a line that gives the user id of the person running the most number of processes and the number of processes that user is running. You must use the pipeline you created, and you may **not** use any program that produces the answer in one statement (For example, using grep, uniq, or sort is not allowed.)

```
#!/bin/sh

#userlist=‘ps aux |cut -f 1 -d " " | sort‘
userlist=‘cat tmp‘

max=0
maxuser=''
sum=0
prev=''

for u in $userlist
do
    if [ "$prev" = "$u" ]
        then
                sum=‘expr $sum + 1‘
        else
                if [ $sum -gt $max ]
                then
                        max=$sum
                        maxuser=$prev
                fi
                sum=1
                prev=$u
        fi
done

echo Biggest user is $maxuser with $max processes
```

Question 4. [9 MARKS]

Suppose the current working directory contains a set of class files where each file contains the list of cdf ids for students belonging to that class. There is one cdf id per line in each file.

Part (a) [5 MARKS]

Write a Bourne shell program, `coursestaken`, that takes as its first argument the cdf id of a student. The rest of the arguments are the names of class list files. The program will print to standard output the names of the files that contain the cdf id. No other output will be printed. If the cdf id does not exist in any of the class list files, nothing will be printed.

```
#!/bin/sh

stu=$1
shift

for course in $*
do
        if grep $stu $course >& /dev/null
        then
                echo $course
        fi
done
```

Part (b) [4 MARKS]

Using the program you wrote in part a), write another program, `numtaken`, that takes a class list file name as an argument. For each cdf id in the file the program will print to standard output the *cdf id* and the *number of class lists* in the current working directory that cdf id is found in. No other output will be printed.

For example, we run `numtaken 209` and 209 contains

```
c4reidka
c5patchi
```

If c4reidka was found in 3 class list files in the current working directory, and c5patchin was found in 1 class list file, then the output would be:

```
c4reidka 3
c5patchi 1
```

```
#!/bin/sh

while read stu
do
        echo $stu `othercourses $stu * | wc -l`
done < $1
```