Duration: **50 minutes**
Aids Allowed: **None**

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 5 questions on 6 pages (including this one), plus the aid sheet. *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 or Perl programs you are asked to write. In C programs, you do not need to add "#include" lines, or do error checking unless the question requires it, or the program would not function correctly given valid input without error checking.

MARKING GUIDE

1: _____/ 5
2: _____/ 9
3: _____/ 9
4: _____/ 7
5: _____/10

TOTAL: _____/40

Good Luck!

Question 1. [5 MARKS]

Part (a) [1 MARK]

Suppose a shell program called `doit` is stored in a directory called `programs` that is a subdirectory of your home directory. Assume `programs` is not in your PATH variable.

If the current working directory is your home directory, write a command to run `doit` without changing directories. Do not use an absolute path.

```
programs/doit
```

Part (b) [2 MARKS]

Consider the following sequence of shell commands:

```
penguin% echo $PWD
/u/reid
penguin% pwd
/h/u1/reid
```

Do the two paths refer to the same directory? _____

What Unix file system mechanism explains the difference in the results? Explain briefly.

Yes, they refer to the same directory. /u/reid is a symbolic link to /h/u1/reid.

Part (c) [2 MARKS]

Without using temporary files, write a shell command that will print out the lines of a file called "Book" that contain both of the words "Bilbo" and "Frodo". (The words can be anywhere in the line and can appear in either order.)

```
grep Bilbo Book | grep Frodo
```

Question 2. [9 MARKS]

Part (a) [3 MARKS]

Given the following two hashes in Perl, complete the Perl code necessary to print out the name of each TA and the room they are in. If the TA does not have a room, then skip that TA. For example, one line of the output will be "Wei RW 110".

```
%sections = ("Wei", "tut0", "Yilan", "tut1", "Russ", "tut2",
         "Brad", "tut3", "Unknown", "tutx");

%rooms = ("tut0", "RW 110", "tut1", "SS1070", "tut2", "SS1072", "tut3", "SS 1074");

# Solution
foreach $ta (keys(%sections)) {
  if(defined($rooms{$sections{$ta}}) {
    print("$ta  $rooms{$sections{$ta}}\n");
  }
}
```

Part (b) [2 MARKS]

Write a Bourne shell program that prints the names of the files but not the directories in the current working directory. Do not use `ls`.

```
for file in *
do
    echo $file
done
```

Part (c) [4 MARKS]

Given that the current working directory contains the following files, and given the following variable definitions below, write the output produced by the echo commands. (' is a single quote, ` is a backquote.)

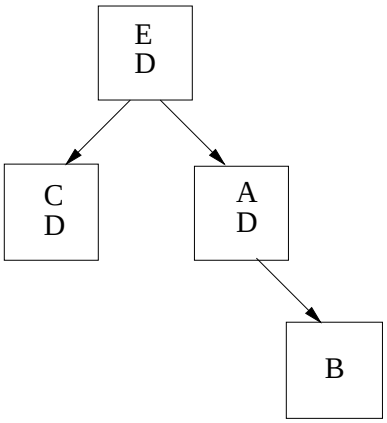
```
cat   p1   p2   pie
```

```
a="ls"
b="p?"
c="cat"

echo $a $b      ls p1 p2

echo "$a $b"    ls p?

echo '$a $b'    $a $b

echo `$a $b`    p1 p2
```

Question 3. [9 MARKS]

In the diagram below, each box represents a process. An arrow represents the creation of a child process. The contents of each box are the lines printed by each process. For example, the top box is the original process and prints "A" and "D" (in separate `printf` statements).

Write a C program that would lead to the set of processes illustrated below. Your program must contain only one `printf("D");` statement.




```
#include <stdio.h>

int main() {
    int j, i;
    if((i = fork()) == 0) {
        if( (j = fork()) == 0) {
            printf ("A\n");
        } else {
            printf ("B\n");
            exit();
        }
    } else {
        if( (j = fork()) == 0) {
            printf ("C\n");
        } else {
            printf("E\n");
        }
    }
    printf("D\n");
}
```

Question 4. [7 MARKS]

Write a C program that will print out the the directories from the user's PATH variable that could not be accessed. Your program should also print out a message that explains why the directory was not accessible. Assume that the following C function has already been defined for your use. (Do not write `getNextDir`!)

```
/* Takes as a parameter a colon separated list of directories. Each time
 * getNextDir is called, it returns the next directory in the list.
 * Returns NULL when there are no more list elements.
 */
char *getNextDir(const char *path_list)

int main(int argc, char *argv[])
{
    char *env = getenv("PATH");
    char *dirname;
    DIR *dp;
    struct dirent *entry;

    char *path = (char *)malloc(strlen(env) + 1);  /* Not necessary */
    path = strncpy(path, env, strlen(env) + 1);    /* for full marks */

    /* For each path element */
    while((dirname = getNextDir(path))!= NULL) {

    if((dp = opendir(dirname)) == NULL) {
        printf("%s\n", dirname);
        perror("opendir");
      }
      free(dirname);
    }
}
```

Question 5. [10 MARKS]

Write a Perl program that checks if records being sent to a data base are correctly formatted. The program reads from standard input, and each line of standard input corresponds to a single record. If a record is valid, it will be printed to a file called "clean.data". If a record is not correctly formatted, it will be printed to a file called "bad.data". (Hint: check the prototype sheet for print functions.)

A record contains exactly 3 fields separated by one or more spaces. The record formats are as follows:

- CDF username: The first character must be 'c' or 'g', the second character must be a digit, and the total length of the user name must not exceed 8 characters.
- Student number: Exactly 9 digits
- Mark: an integer or floating point number between 0 and 60.

```
open(CLEAN, ">clean.data") || die "Couldn't open clean.data\n";
open(BAD, ">bad.data") || die "Couldn't open bad.data\n";

my $line;
while($line = <STDIN>) {

  if($line =~ /^[cg]\d\w{1,6}\s+\d{9}\s+(\d+)\s*$/) {
    if($1 >= 0 && $1 <= 60) {
      print(CLEAN $line);
    } else {
      print(BAD $line);
    }
  } else {
    print(BAD $line);
  }
}
```