CSC209F Midterm (L0101)

Fall 1998

University of Toronto

Department of Computer Science

Date: November 6th, 1998
Time: 1:10 pm
Duration: 50 minutes

Notes:

1. This is a closed book test, no aids are allowed.
2. Print your name, student number, and cdf username in the space provided below.
3. There are a total of 50 marks.
4. All shell questions assume `csh`.
5. This test is worth 20% of your final course mark.
6. This test comprises 7 pages. Do not detach pages, and answer questions in the blank spaces provided. If you need more space, answer on the back of the pages (clearly identify which question in this case).
7. **Read all questions before starting. Not all questions are worth the same number of marks, so budget your time accordingly. Answer questions in any order you desire.**

Name:	**Marking Scheme**
Student Number:	
CDF Username:	

Marks:

Q1		Q5	
Q2		Q6	
Q3			
Q4			

Total

Q1: [15 marks]

1. Why are the commands `logout/exit` built into the shell?

   If they weren't, they would be child processes and couldn't (easily) terminate the parent.

2. Where should your alias statements be placed, in `.login` or `.cshrc`? Explain.

   `.cshrc` — since `.login` isn't sourced every time a new shell is created

3. (2 marks) What is the difference between a shell variable and an environment variable? (List at least 2 differences.)

   (1 mark) Shell vars can have multi-valued (array) properties
   (1 mark) Shell vars aren't visible to an `exec()`'d process while environment vars are

4. (2 marks) List two different ways in which a shell script can protect itself from unexpected behaviours due to aliased commands.

   (1 mark) `#!/bin/csh -f`
   (1 mark) preface commands with '\' to over-ride any aliases

5. How would you repeat your 2nd most recent command without retyping it?

   !-2

6. Consider the `.` and `..` special files. Is it correct to say that these files are system-created hard links to directories?

   Yes.

7. (2 marks) Joe wants to print out a numbered listing of the file "`fa`". He types "`cat -n fa > lpr`" but no printout appears. Why? What command would you suggest he use?

   (1 mark) he has redirected `stdout` into a file named "lpr"
   (1 mark) `cat -n fa | lpr`

8. Where does UNIX store the default shell to be used when a user logs in?

   `/etc/passwd`

9. (2 marks) Show an alias command that will cause `ls` to list hidden files (like `.`, `..`, `.login`, *etc.*) without explicitly typing a command line parameter each time. Show also how to execute `ls` without the parameter after the alias has been define (on a once-only basis).

   (1 mark) `alias ls 'ls -a'`
   (1 mark) `\ls`

10. (2 marks) Briefly explain the I/O redirection occurring in each of the following commands:
    ```
    cmd1 | cmd2
    cmd1 |& cmd2
    ```

    The first connects `stdout` from `cmd1` to `stdin` for `cmd2`. The second connects both `stdout` and stderr from `cmd1` to `stdin` for `cmd2`.

Q2: [4 marks]

Why is a hard link indistinguishable from the original file itself?

The "original file" is really just a hard link itself to the file's inode. If you add a new hard link, it will behave identically.

(2 marks) What happens if you `rm` a hard link?

The hard link is removed (1 mark), and if the file's (inode's) hard link count goes to 0, the file is also removed (1 mark).

Why is it not possible to have a hard link to a file in a different filesystem?

A hard link need to specify the inode of the linked file, and it is assumed that the inode is in the same filesystem (there is no mechanism for specifying an inode and a filesystem in the link).

Q3: [4 marks]

Write a code fragment which takes a string defined as

```
char *cmdLine ;
```

and processes it to put it in the correct format for `execvp(char *filename, char *argv[])`; You are to use `strtok(char *s, const char *delim)` to break the string into tokens. You can assume that there are no more than MAX_CMDLINE_ARGS tokens to be processed. Define whatever new variables you need.

```
int   i = 0 ;
char *argv[MAX_CMDLINE_ARGS+1];

memset(argv, 0, sizeof(argv));

argv[0] = strtok(cmdline, WHITESPACE);
while (argv[i] != (char *)NULL)
{
  i++ ;
  argv[i] = strtok(NULL, WHITESPACE);
}
```

Q4: [5 marks]

Consider the shell variable assignments:

```
set x = ( fred barney wilma betty )
set y = ( pebbles bam-bam )
set z = ( $x $y )
```

1. (1 mark) What is the output of "`echo $#z`"?

   6

2. (1 mark) What is the output of "`echo $?y`"?

   1

3. (1 mark) What is the output of "`set a = $x; echo $a`"?

   fred

   This shows the importance of parentheses around `set` assignments.

4. (2 marks) Show how the word "pebbles" can be removed from `y` using a simple assignment statement. The answer "`set y = bam-bam`"" is **not** acceptable in this case.

   ```
   set y = ( $y[2] )
   ```

   or

   ```
   set y[1] =
   set y[1] = ""
   ```

   These last two will be accepted, although they do not have the desired effect on $#y, namely reducing it to 1

Q5: [10 marks]

1. Write a shell script `newext` so that

   ```
   newext oldExt newExt
   ```

   searches for all the files in the current directory with extension `oldExt` and renames them to have the extension `newExt`. Your script should make sure a file with the new name doesn't already exist and give a warning if it does (in this case do not rename the file).

```
#!/bin/csh -f

if ( $#argv != 2 ) then
  echo Usage: $0 oldExt newExt
  exit
endif

set files = ( *.$1 )
if ( $#files == 0 ) then
  echo No files found ...
  exit
endif

foreach file ( $files )
  set newName = $file:r.$2
  if ( -e $newName ) then
    echo $newName already exists, $file not renamed
    continue
  else
    mv $file $newName
  endif
end
```

This is a complete, tested shell script. You do not need something this complete to get full marks.

Q6: [12 marks]

Write a C program that takes any number of file names as command line parameters. For each filename the program should create a child process which executes a UNIX utility "`foo`" on the named file. As each child process is created, increment a global variable to keep track of the number of child processes. Implement a signal handler that detects when a child exits by catching SIGCHLD. Your handler should print a statement as to whether the child terminated normally or not, and what exit status code was returned. It should then decrement the child process counter. The parent, after `fork()`-ing and `exec()`-ing all the child processes, will enter a `while()` loop which continually puts the parent to sleep for 1 second until the child process counter is 0. Recall that signal handlers are functions that have one `int` parameter and return `void`.

The following function prototypes may be useful:

```
execlp(char *filename, char *arg0, ... , (char *)0);
int wait(int *status);
int waitpid(int pid, int *status, int options);
void *malloc(long size);
void *realloc(long size);
char *strcpy(char *dest, char *src);
int   strcmp(char *s1, char *s2);
int   signal(int sig, void (*handler)(int));
WIFEXITED(status)
WEXITCODE(status)
WIFSIGNALED(status)
WTERMSIG(status)
WIFSTOPPED(status)
WSTOPSIG(status)
```

```
int childCount = 0 ; /* global var for child counting */

void SigChildHandler(int signo)
{
  int status, pid ;

  pid = wait(&status);
  if (WIFEXITED(status))
    printf("Child %d exited: return code = %d\n", pid,
           WEXITCODE(status));
  else
    printf("Child %d terminated abnormally.\n", pid);

  childCount-- ;
}

int main(int argc, char *argv[])
{
  int i ;

  signal(SIGCHLD, SigChildHandler); /* may want error checking here */
  for (i=1; i<argc; i++)
    switch (fork())
    {
      case 0:
        execlp("foo", "foo", argv[i], (char *)0);
        exit(i); /* need this in case execlp() fails   */
        break ;  /* this is here out of force of habit */
```

(Use this page for answer overflows)

```
      case -1:
        printf("fork() failure for %s.\n", argv[i]);
        break ;

      default:
        childCount++ ;
        break ;
    }

  while(childCount) sleep(1);
  return ( 0 );
}
```