PLEASE HAND IN

PLEASE HAND IN

UNIVERSITY OF TORONTO
Faculty of Arts and Science

APRIL EXAMINATIONS 2004

CSC 209H1 S
St. George Campus

Duration — 3 hours

Examination aids: none

Student Number: |_|_|_|_|_|_|_|_|_|_|_|

Last Name: _____

First Name: _____

Instructor: _____

---

*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.)

This examination consists of 10 questions on 16 pages (including this one). *When you receive the signal to start, please make sure that your copy of the examination is complete and fill in your student number on every page.* If you need more space for one of your solutions, use the last pages of the exam or the back of this page and *indicate clearly the part of your work that should be marked.*

1: _____/15
2: _____/ 8
3: _____/ 7
4: _____/ 9
5: _____/ 7
6: _____/ 7
7: _____/ 6
8: _____/ 9
9: _____/10
10: _____/14

TOTAL: _____/92

Good Luck!

Question 1. [15 MARKS]

Part (a) [8 MARKS]

Circle the correct answer below.

TRUE FALSE The statement, `unsigned int y = x >> 16` gives `y` the value of the upper 16 bits of `x` (assuming `x` is an unsigned int).

TRUE FALSE I can only execute a program that is in my current working directory or in one of the directories found the `PATH` variable.

TRUE FALSE Sockets are not a good candidate for sending messages between processes on the same machine because the overhead is too high.

TRUE FALSE It is important to free memory that you allocate using `malloc` before you exit a program because otherwise that memory will no longer be available for other programs to use.

TRUE FALSE The function `strncpy(dest, src, n)` will work correctly even if the null-termination character is not found in the first `n` bytes of `src` or `dest`

TRUE FALSE The following is legal C code:

```
int *p = malloc(10*sizeof(int));
p[3] = 10;
*p = 5;
```

TRUE FALSE Blocking a signal means that the signal is thrown away and can never be received.

TRUE FALSE We do not need to allocate memory for the pointer `ptr` when it is used in the following statement: `ptr = fgets(ptr, 10, stdin);`

Part (b) [7 MARKS]

The directory `adir` is a subdirectory of the current working directory. In `adir` are four files. Two files (`prog1` and `prog2`) are C object files, and two (`getone` and `getall`) are shell programs. Assume that the four programs are correct.

```
d-wx-wx-wx    adir/
---x--x--x    adir/prog1
-r-xr-xr-x    adir/prog2
---x--x--x    adir/getone
-r-xr-xr-x    adir/getall
```

For each of the following commands, circle TRUE if the command will run correctly and FALSE otherwise. Consider each command independently.

TRUE FALSE `ls adir`

TRUE FALSE `cd adir`

TRUE FALSE `adir/prog1`

TRUE FALSE `adir/prog2`

TRUE FALSE `adir/getone`

TRUE FALSE `adir/getall`

TRUE FALSE `rm adir/xx`

Question 2. [8 MARKS]

Part (a) [2 MARKS]

We cannot reliably use the `select` function to check when data is ready to read from a file pointer returned by `popen` even though it is possible to add the appropriate file descriptor to the set that `select` is waiting on. Briefly explain why not.

Part (b) [1 MARK]

Write a C statement that reads an integer from file descriptor `fd`. Include any necessary variable declarations.

Part (c) [2 MARKS]

Briefly explain how to terminate or kill a process that is running in the background.

Part (d) [3 MARKS]

In assignment 3, a child process wrote data to the parent process through a pipe. A number of students wrote code that called `wait` in the parent before the parent read from the pipe. Explain why this seemed to work.

If the parent waits before it reads from the child, explain what error might occur, and why the error occurs.

Question 3. [7 MARKS]

When a process terminates it sends the SIGCHLD signal to its parent. Normally, the parent ignores the signal until it calls wait. Your task is to write a C program that prints a message as soon as a child process terminates and the SIGCHLD signal is received.

If the child process terminates normally, the parent will print the message, "Child x terminated with y" where "x" is the process id of the child and y is its exit status. If the child terminated abnormally, then the parent will print the message, "Child x terminated" where "x" is the child's process id.

To make this program complete, the child and the parent need something to do. The child will call the function `playTime()` and the parent will call `makeDough()`. Do **not** write these two functions.

Question 4. [9 MARKS]

Part (a) [2 MARKS]

The buffer returned by `fgets` contains the newline character. Write a C code fragment to remove the newline character from `line`.

```
char line[MAXBUF];
fgets(line, MAXBUF, stdin);
```

Part (b) [7 MARKS]

Write a C program that implements a limited form of "grep". The first command line argument is the word to search for and the remaining command line arguments are the files to search. The program prints to standard output each line in the files that contains the word. If a file cannot be searched, the program prints "permission denied".

For example if the program was run with the arguments "foo file1 file2" then all of the lines in file1 and file2 that contain the word "foo" would be printed.

Question 5. [7 MARKS]

Write a program that creates 5 child processes, numbered 0 through 5. Each child process computes the sum of the integers $5 \times i$ through $5 \times i + 4$, where i is its index number. It then returns this value as the process exit status. The parent waits for all of its children and outputs the total of the exit statuses, which will be the sum of the integers from 0 to 24 inclusive.

Question 6. [7 MARKS]

Consider the following C program.

```
 1 int main() {
 2     printf("Start\n");

 3     if(fork() == 0) {
 4         printf("Child1 start\n");
 5         if((j = fork()) == 0) {
 6             printf("Child2");
 7             return 0 ;
 8         } else {
 9             printf("Child1 end\n");
10             return 0;
11         }
12     } else if (i > 0) {
13         printf("Parent\n");
14     }

15     printf("End\n");
16     return 0;
17 }
```

Part (a) [3 MARKS]

Circle the correct answer

TRUE FALSE It is possible for "Child1 end" to be printed before "Child2"

TRUE FALSE It is possible for "Child1 end" to be printed before "Child1start"

TRUE FALSE It is possible for "Parent" to be printed before "Child1 start"

TRUE FALSE It is possible for "End" to be printed before "Child1 start"

TRUE FALSE It is possible for "Child2" to be printed before "Child1 start"

TRUE FALSE It is possible for "End" to be printed more than once

Part (b) [2 MARKS]

Is it possible to insert a call to `wait(0)` that would guarantee that "Parent" would be printed after "Child1 end"? _____

If so, the call to `wait(0)` should be placed immediately following line number _____.

Part (c) [2 MARKS]

The program can be modified to guarantee that "Parent" will be printed after "Child 2", by adding one or more calls to `wait(0)`. After which line or lines should a call to `wait(0)` be added? (Full marks will be awarded only if the minimum number of `wait(0)` calls are added.)

Question 7. [6 MARKS]

Write a Bourne shell program that takes a path as an argument. It prints the number of path elements and "relative" if it is a relative path or "absolute" if it is an absolute path.

The man page descriptions of `dirname` and `basename` may help you:

```
SYNOPSIS      dirname NAME
DESCRIPTION   Print  NAME  with its trailing /component removed; if NAME
              contains no /'s, output '.' (meaning  the  current directory).

SYNOPSIS      basename NAME
DESCRIPTION   Print  NAME with any leading directory components removed.
```

Question 8. [9 MARKS]

Part (a) [5 MARKS]

The file "classlist" contains a list of student ids. Write a Bourne shell program that creates a subdirectory for each student in `/u/csc209/marks/a3` and copies the file whose name is given as a command line argument to each student's directory. Do not use `cd`.

Part (b) [2 MARKS]

To run the program in part (a), is it necessary for the file passed in as an argument to be in the same directory as the program?

_____

If yes, explain how to run the program if the file given as an argument is in a different directory. If no, explain why not.

Part (c) [2 MARKS]

To run the program in part (a) is it necessary for the file class list to be in the current working directory?

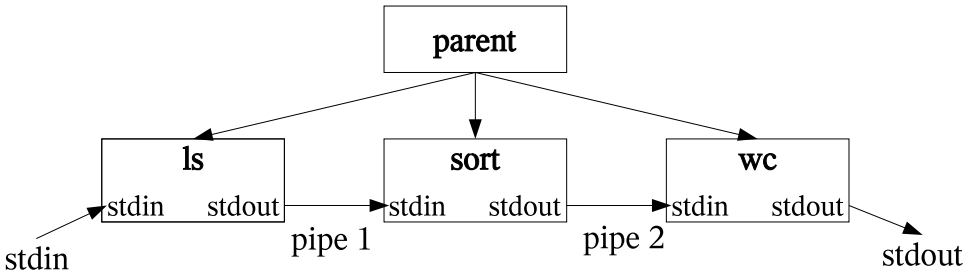
_____

If yes, explain how to run the program if "classlist" in a different directory. If no, explain why not.

Question 9. [10 MARKS]

Pipes are frequently used in Unix to connect the standard output from one process to the standard input of the next. Write a C program that takes 2 or more program names as command line arguments, constructs pipes, and executes the programs so that the output of one process becomes the input of the next. Full marks will only be awarded if the pipes are closed correctly.

For example, if the program is run with the arguments "ls sort wc", the parent creates three processes to execute the programs and sets up two pipes.



This page is provided for rough work and any answers that didn't fit.

Question 10. [14 MARKS]

Part (a) [6 MARKS]

One problem that may occur with sockets is that a read on a socket may block indefinitely. This is undesireable if there are other connections that the process could be handling. Name and describe three techniques that may be used to solve this problem.

Technique 1: _____

Technique 2: _____

Technique 3: _____

Part (b) [8 MARKS]

A port scanner is a program that tries to find out which ports have servers actively listening on them. A simple way to do this is to iterate over a set of ports, trying to establish a connection on each port.

Complete the C program on the next page to implement a port scanner that is looking for a web server listening on ports in the range 8000 to 10000. It attempts to determine whether a server listening on a port is a web server by sending a "GET" request, and checking if a response begins with "HTTP". (Note that the "GET" request message is already set up for you.) You do not need to handle the problem described in part (a).

The program will print the following messages in the appropriate circumstances. In each case "x" is the port number.

- Port x answered with HTTP
- Port x is not a web server
- Port x is not listening

```
#define BLOCK_SIZE 128

int main(int argc, char* argv[])
{
    int i, soc;
    char buf[BLOCK_SIZE];
    struct hostent *hp;
    struct sockaddr_in peer;

    strcpy(buf, "GET /index.html HTTP/1.0\r\n");
    peer.sin_family = AF_INET;

    hp = gethostbyname(argv[1]);
    if ( hp == NULL ) {
        fprintf(stderr, "%s: %s unknown host\n",argv[0], argv[1]);
        exit(1);
    }

    peer.sin_addr = *((struct in_addr *)hp->h_addr);
    soc = socket(AF_INET, SOCK_STREAM, 0);
```

This page is provided for rough work and any answers that didn't fit.

Total Marks = 92