CSI 402 – Spring 2012

Programming Assignment V

Administrative Information

- **Deadline:** 11 PM, Sunday, May 6, 2012. **There is no (two-day) grace period for this assignment.**
- Two parts, but only *one* `makefile`.
- The program Part (a) may be in a single C source file. The program for Part (b) must have two or more C source files.
- The files for both parts (C source files, header files and the `makefile`) must be submitted together using the `turnin-csi402` command.
- README file

  `~csi402/public/prog5/prog5.README`

  will be available by 10 PM on Tuesday, Apr. 24, 2012.
- **The `README` file will contain additional specifications for the `makefile`.**

Administrative Information (continued)

- **Weightage:** 10%
- **Total Points:** 100
  - **Part (a):** 35 points (Correctness: 30, Str. & doc: 5).
  - **Part (b):** 65 points (Correctness: 55, Str. & doc: 10).

Caution

This is also a challenging project. You are strongly advised to start working on the project right away.

Details Regarding Part (a)

- **Hidden files:** Files whose names start with `'.'`.
- **Goal:** To print information about hidden files.
- **Command line:**

  % `p5a` *pathname*

- Program must go through the files in the directory specified by *pathname*.
- For each hidden file, except `"."` and `".."`, program must print to `stdout`, the name of the file, the file size (in bytes) and the date (month, day and year) of last modification.
- You should *not* use the `ftw` system call.

Details Regarding Part (a) (continued)

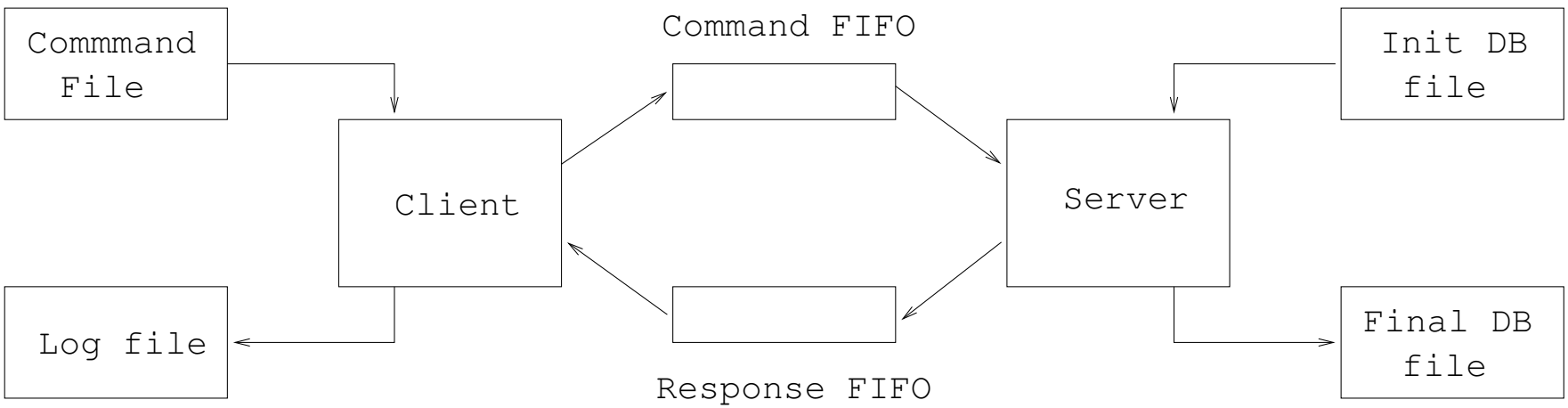
Suggestion: The following material from of the text by Haviland et al. will be useful.

- Pages 53 through 57 of Section 3.3: In particular, you must understand the specifications of the library function `stat` and the components of the `stat` structure (pages 53–54).
- Pages 67 through 71 of Section 4.4: In particular, you must understand the specifications of the functions such as `opendir`, `closedir`, `chdir` and `readdir`.
- Page 318 of Section 12.4 (**Time**): This section discusses how the time values stored in the `stat` structure can be converted into conventional date/time specifications.

Details Regarding Part (b)

Goal: To implement a simple client-server system using **named pipes** (**FIFOs**).

Structure of the System:



Details Regarding Part (b)

Suggestion: Start by studying Chapters 5 and 7 of [HGS]. (Lecture slides and handouts for Lectures 13 and 15 will also be useful.)

Important Notes:

- The server and client are **separate** programs.
- The corresponding executables must be named `p5b_server` and `p5b_client` respectively.

Unix Command Line:

% `p5b_server` *initdbfile* *finaldbfile* *cmdfile* *logfile*

Important Note

The server should start the client using `fork` and (an appropriate version of) `exec` system calls.

Example: Initial Database File

```
5
Hanks,Tom        2  311  403
Baker,Norma      3  311  402  403
Allen,Woody      1  404
Roberts,Julia    1  310
Pitt,Brad        0
6
310  3   TH--11:45-1:05
311  3   MW--12:30-1:25
402  3   TH--1:15-2:35
403  3   TH--10:15-11:35
404  3   MWF--1:40-2:35
445  3   T--4:15-7:15
```

Example: Command File

```
addc  Hanks,Tom      404
addc  Martin,Steve   310
drpc  Baker,Norma    311
wdrw  Allen,Woody
wdrw  Allen,Tim
tcre  Roberts,Julia
newc  424  3   MWF--9:15-10:10
csch  310  MWF--1:25-2:20
ccre  402  4
gsch  445
gsch  426
gcre  446
gcre  402
```

The Corresponding Log File

```
 0  addc  1
 1  addc  1
 2  drpc  1
 3  wdrw  1
 4  wdrw  0
 5  tcre  1   3
 6  newc  1
 7  csch  1
 8  ccre  1
 9  gsch  1   T--4:15-7:15
10  gsch  0   Error
11  gcre  0   -1
12  gcre  1   4
```

The Final Database File

```
6
Hanks,Tom        3  311  403  404
Baker,Norma      2  402  403
Allen,Woody      0
Roberts,Julia    1  310
Pitt,Brad        0
Martin,Steve     1  310
7
310   3    MWF--1:25-2:20
311   3    MW--12:30-1:25
402   4    TH--1:15-2:35
403   3    TH--10:15-11:35
404   3    MWF--1:40-2:35
445   3    T--4:15-7:15
424   3    MWF--9:15-10:10
```

Additional Remarks Regarding Part (b)

Outlines for server and client:

- See page 6 of handout.

Assumptions for Part (b):

- Max. no. of students in the database = 100.
- Max. no. of courses in the database = 100.
- Max. no. of courses for each student = 10.

Additional Clarifications Regarding Commands:

- The `exit` command is *not* part of the command file.
- The command `newc` must fail if adding the course increases the number of courses in the database to 101.

Additional Clarifications ... (continued)

- The command `addc` must fail if *any* of the following conditions hold.
  - Adding the course will increase the number of courses in which the student is registered to 11.
  - Adding the course will increase the number of students in the database to 101.
  - The student is already registered for the specified course.
- The `drpc` command must fail if the student is not registered for the specified course.
- The `wdrw` command must fail if the student is not registered for any course.

Errors to be detected: See handout.

Recommendations

- Follow the suggested outline for the server and client programs.
- Use `stdio` library functions (e.g. `fopen`, `fscanf`) for database, command and log files. System calls (e.g. `open`, `read`, `write`) are needed for FIFOs.
- Use simple data structures to store the student and course information.
- Define a suitable `struct` for sending a command to the server and another `struct` for receiving a reply from the server. Read from and write to FIFOs using these `structs`.
- Get the `exit` command and one other command to work first; then add one new command at a time.