HOMEWORK ASSIGNMENTS

CS-450 & CS-550: Operating Systems

Fall 2008 Semester

Textbooks and *JMU* Facilities for these Assignments:

Textbooks:

1. TANENBAUM, ANDREW S. (2008). *Modern Operating Systems. Third Edition.* Upper Saddle River, NJ: Prentice-Hall. QA76.76.063 T359 2008; 005.4'3—dc21; ISBN: 0-13-601919-6, 978-0-13-601919-0.

2. SOBELL, MARK G. (2005). *A Practical Guide to LINUX Commands, Editors, and Shell Programming.* Upper Saddle River, NJ: Prentice Hall Professional Technical Reference. QA76.76.063 S59483 2005; 005.4'46—dc22; 2005050051; ISBN: 0131478230 (pbk. : alk. paper).

LINUX Computing Facilities:

An account has been opened for you on the ***Linux*** computers in the lab in ISAT/CS Room 248. The Username for your account on these machines is the same as your JMU Electronic ID, and your password is the same as your JMU system password. NOTE that when you change your JMU network password, the change will automatically be carried over to the ***Linux*** facility. You should use the lab computers to familiarize yourself thoroughly with ***Linux,*** and in particular to practice and to familiarize yourself with **all** of the operations described in your ***Linux*** reading assignments.

(1) Please note that the common feature of ALL of the assignments pertaining to *Linux* is that I expect you to become familiar with the *Linux* Command-Line Interface (CLI). You may practice the various *Linux* commands either on your own computer, if you have *Linux* loaded on it, or on one of the computers in the CS Department's *Linux* Laboratory (ISAT/CS Room 248), or on the CS Department's web server for students (*stu.cs.jmu.edu*).

(2) **Logging in:** On all of the *Linux* machines in the CS Department's ***Linux*** laboratory (ISAT/CS Room 248) the login screen is a Graphical User Interface, or GUI. The login screen for the GUI has a brownish red background pattern, and is prominently marked, "RED HAT ENTERPRISE LINUX 5." To use the machine, you must first log into the GUI.

   a. Type your Username, **all in lower case,** in the appropriately marked white box.
   b. Press either <ENTER> or <TAB>.

c. The label on the white box will change from "Username" to "Password."
d. Type in your password, taking care to distinguish lower-case from upper-case,
e. Press <ENTER>.
f. You should then see a screen with several icons on the left side, stacked in a vertical column. The lowermost icon is labeled "Trash", and the second one from the top is labeled as your Username's home.
g. Once you have logged into the GUI, you have at least three ways to reach the **Command-Line Interface (CLI).** You can either:
   i. Press <CTRL><ALT>F1. This gives you a black screen with white text characters.
   ii. Right-click anywhere on the desktop background. This brings up a pop-up menu. In the pop-up menu, click on **"Open Terminal."** A new window will then appear on the screen, bearing a CLI in the form of black text on a white background.
   iii. In the menu bar at the top of the screen, click on **"Applications/Accessories/Terminal."**

You should try **all three** of these methods to see how they differ from each other. Once you have familiarized yourself with all of them, then you are free to select whichever one you prefer for regular use.

Once you reach the CLI, you may be prompted to log in **again**. Go ahead and do that, BUT when you are done working, be sure to remember to log out one time for each time that you are logged in. This point will be addressed again, with further details, in item (4) below.

(3) **Saving Files Beyond the End of the Session:** When you are logged in to any of the ***Linux*** machines in Room 248, you can save any files that you like onto the hard disk. HOWEVER, please be aware that around midnight each evening, the hard disk of every machine in Room 248 is wiped clean. **Thus, if you want to retain any files saved to the hard disk, you must copy them from the hard disk to some other medium before the nightly erasure of the hard disk occurs.**

(4) **Logging OUT of the PC in the *Linux* Laboratory:** To log out completely from the PC in the ***Linux*** laboratory, **in addition to logging out of the CLI, you must ALSO log out of the machine's GUI-based desktop environment.** To do this, after logging out of the CLI, activate "System" on the menu bar at the top of the screen. This will open a pull-down menu. From the pull-down menu, select **"Logout *YourUserName*",** which is the third item from the bottom out of nine total items on the menu.

(5) **Using the CS department's *Linux* Server:** In addition to your account on all of the individual ***Linux*** PCs in Room 248, you also have an account on the CS Department's ***Linux*** server. The Domain Name for the department's ***Linux*** server is: ***stu.cs.jmu.edu*** You can get there either from one of the ***Linux*** PCs in Room 248, or from any ***Windoze*** PC on the JMU network, via the Secure Socket Header **(*ssh*)** protocol. From a ***Linux*** PC, type: ***"ssh stu.cs.jmu.edu",*** and then log in with your JMU EID and password. The ***ssh*** protocol is, indeed, as the name suggests, secure. **DO NOT EVER log in over the network to the departmental *Linux* server using any protocol that is NOT secure.**

NOTE that if you use a machine in the *Linux* laboratory to log in to the department's ***Linux*** server, then when you are done working, **you will have to log out a total of three times. FIRST,** log out from the ***Linux* server. THEN** log out from the **CLI on the local machine.** And **FINALLY,** log out from the **GUI on the local machine.**

(6) **Programming Language for the Programming Assignments for this Course:** The standard programming language for the ***GNU-Linux*** operating system environment is **not** ***C++,*** but rather plain old vanilla ***C.***

Therefore, **to receive credit** for your assignments **you must write them all in *C*, without utilizing any of the object-oriented features of *C++*.** Note that the prohibition on object-oriented features extends to library routines as well. The file with your source code should bear a name of the form: ***ProgramName.c*** and **not *ProgramName.cpp*** or any of the other filename extensions used for *C++* programs, and it must be compilable with the GNU *C* compiler, which is invoked via the command ***gcc*.**

(7) **Development of the Programs for Your Assignments:** You are free to familiarize yourself with the ***Linux*** command-line environment on any machine of your choice. However, for those programs that I assign you to write and hand in, I will expect you to do all of the following: **write, compile, test, and debug your program on the CS department's *Linux* server for students, *stu.cs.jmu.edu*** This means that you are **not** to write, compile, and debug your program on some other machine, not even on the PCs in the ***Linux*** laboratory, and then just copy it over to the department's server. Even though your program may compile and run correctly on some other machine, there are sometimes rather subtle effect that may render it unrunnable on the department's ***Linux*** server. In grading your assignment, your source code may be recompiled and run or rerun on the department's ***Linux*** server. Your grade will be determined based solely upon how the program behaves during the grading procedure. You will not have an opportunity to redo it or to demonstrate proper operation on some other machine. Therefore, if you take a shortcut and do your development and testing on another computer and then just copy it over to the department's ***Linux*** server, that might cost you heavily in your grade for the project. Note that you are free to access the department's ***Linux*** server from any location of your choice on the planet, so long as you access it via a secure communications protocol.

(8) **Importance of Adhering to the Specified File-Naming Conventions:** In conjunction with the programming assignments, you will be given instructions on how you are to name your various files. **Be sure to adhere punctiliously to the instructions given for naming your files.** This includes but is not limited to **capitalization, hyphens** embedded in the file name, and **file extensions.** Please note that failure to adhere precisely to the instructions for the naming of files can result in your losing a major part or maybe even all of the credit for your assignment.

Assignment 1:

READINGS :

- *Modern Operating Systems. Third Edition***:** Preface (pages xxiv-xxvi) and Chapter 1: *Introduction* (pages 1 through 79).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming***:** Preface (pages XXVII-XXXVII); Chapter 1: *Welcome to Linux* (pages 5-17); and Chapter 2: *Getting Started* (pages 19-39).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming***:** Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands covered in the assigned reading from Sobell.

DELIVERABLE:

- *Modern Operating Systems. Third Edition***:** Problems 7, 10, 18, 19, and 23 from Chapter 1, pages 79-82.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming***:** Exercises 1, 6, 7, and 10 from Chapter 1, page 17; also, exercises 1, 3, 8 and 9 from Chapter 2, pages 39-40.

Assignment 2:

READINGS

- *Modern Operating Systems. Third Edition*: Chapter 2: *Processes and Threads*, Sections 2.1-2.3(pages 83-145).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 3: *Command Line Utilities* (pages 41-72);

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands covered in the assigned reading from Sobell.

DELIVERABLE:

- *Modern Operating Systems. Third Edition*: Exercises 4, 5, 10, 14, 20, and 24 from Chapter 2, pages 170-174.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises 2, 3, and 13 from Chapter 3, pages 72-73.

Assignment 3:

READINGS:

- *Modern Operating Systems. Third Edition*: Chapter 2: *Processes and Threads*, Sections 2.41-2.7(pages 145-170).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 4: *The Linux Filesystem* (pages 75-103).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands covered in the assigned reading from Sobell.

DELIVERABLE:

- *Modern Operating Systems. Third Edition*: Problems 32, 33, 35, 37, and 38 from Chapter 2, pages 170-174.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises 1, 2, 7, 11, 15, and 16 from Chapter 4, pages 103-106.

Assignment 4:

READINGS:

- *Modern Operating Systems. Third Edition*: Chapter 3: *Memory Management*, Sections 3.1 – 3.4 (pages 175-216).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 5: *The Shell* (pages 107-135).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands covered in the assigned reading from Sobell.

DELIVERABLE:

- *Modern Operating Systems. Third Edition*: Problems 4, 9, 13, 15, and-18 from Chapter 3, pages 248-254.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises 3-6, 9, 11, and 15 from Chapter 5, pages 134-136.

Assignment 5:

READINGS:

- *Modern Operating Systems. Third Edition*: Chapter 3: *Memory Management*, Sections 3.5 – 3.7 (pages 216-247).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 6: *The* **vim** *Editor* (pages 139-192).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands and ***vim*** operations covered in the assigned reading from Sobell.

DELIVERABLE:

- *Modern Operating Systems. Third Edition*: Problems 28, 31, 33, and 37 from Chapter 3, pages 248-254.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises 3, 4, 5, 7, and 8 from Chapter 6, pages 193-194.

Assignment 6:

READINGS:

- *Modern Operating Systems. Third Edition*: Chapter 4: *File Systems* Sections 4.1-4.3 (pages 255-291).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 8: *The Bourne Again Shell* (pages 255-295 — about half of the chapter).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands covered in the assigned reading from Sobell.

DELIVERABLE:

- *Modern Operating Systems. Third Edition*: Problems 5, 11, and 15 from Chapter 4, pages 325-328.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises 2, 3, and 4 from Chapter 8, pages 334-337.

Assignment 7:

READINGS:

- *Modern Operating Systems. Third Edition*: Chapter 5, *Input/Output,* Sections 5.1-5.3 (pages 329-360).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 8: *The Bourne Again Shell* (pages 295-334 — the remaining half of the chapter).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Log on to a ***Linux*** machine, and review the ***Linux*** commands covered in all the readings assigned so far from Sobell.

DELIVERABLE:

- *Modern Operating Systems*: Problems 4, 5, 9, and 11 from Chapter 5, pages 427-432.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises10 and 11 from Chapter 8, pages 334-337.

Assignment 8:

READINGS:

- *Modern Operating Systems. Third Edition*: Chapter 5. *Input/Output*, sections 5.4-5.5 (pages 360-394 **only**).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 10. *Programming Tools* (pages 387-430).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands covered in the assigned reading from Sobell.

DELIVERABLE I:

- *Modern Operating Systems*: Problems 15, 21, 22, and 24 from Chapter 5, pages 427-432. Please pay especially close attention to Problem 24.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises 1, 3, 5, and 8 from Chapter 10, pages 431-433.

DELIVERABLE II:

- Write a **`bash`** Command Procedure (known in *UNIX-speak* as a "Shell Script") that does each of the following:
  1. takes a list of login names as its arguments;
  2. displays the current working directory;
  3. displays today's date and the current time;
  4. displays the login names in the argument list in order as they appear on the list, each on a separate line;
  5. displays on the same line as the login name the UID and the full name of the user who owns the login, if the login name is valid (*i.e.,* if the login name corresponds to an entry in the file **`/etc/passwd`**);
  6. displays on the same line as the login name a message that reads: "NO SUCH LOGIN", if an appropriate entry does not appear in the **`/etc/passwd`** file;
  7. summarizes the results at the end, *i.e.,* displays the total number of logins that it checked, and indicates how many were found to be valid and how many were not valid.

- Your shell script should be thoroughly documented by means of a liberal sprinkling of literate comments.
- Naming convention for the file containing your shell script: **Script1-*YourLastName–FirstInitial***
- Please note especially carefully the details of the file naming, *viz.*:
    i. The first letter only of **"Script1"** is capitalized. Remaining letters are lower-case.
    ii. The first letter only of ***"YourLastName"*** is capitalized. Remaining letters are lower case.
    iii. Your ***"FirstInitial"*** is capitalized.
    iv. The field connections are hyphens, not underscores.

- Set the protection bits for your file so that you yourself have complete access, but Group and Other have no access.
- Copy the file into the following directory on the CS department's ***Linux*** server:
    /cs/shr/OperatingSystems
- **Check that your files were successfully copied into the target directory.** To accomplish this, you need to issue each of the following commands:

```
$ ls -l /cs/shr/OperatingSystems/Script1-YourLastName–FirstInitial
```

RESULT of invoking this command should look something like this**:**

```
-rwx------ 1   9999 csmajor 5432 Nov 31 04:15 Script1-YourLastName–FirstInitial
```

When you check for the presence of your file in the directory, be sure not only to note the presence of the file name in the directory listing, but also to note the file size and the protection bits. It has happened more than once in the past that some student deposited in the shared directory files whose names met the specification, but that had zero content, or accesses that were inappropriate.

Please note that you **must** issue the "*ls -l* " command precisely in the manner shown, in order to obtain the result shown. If you were instead to attempt to obtain a general listing of the directory contents of **/cs/shr/OperatingSystems** by invoking the command:

```
$ ls -l /cs/shr/OperatingSystems
```

RESULT of invoking this command will look something like this**:**

```
ls: /cs/shr/OperatingSystems: Permission denied
```

Assignment 9:

READINGS:

- *Modern Operating Systems. Third Edition***:** Chapter 6. *Deadlocks*, (pages 433-463).
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming***:** Chapter 10. *Programming Tools* (pages 387-430).

PRACTICUM:

- *A Practical Guide to LINUX Commands, Editors, and Shell Programming***:** Log on to a ***Linux*** machine, and try out each of the ***Linux*** commands covered in the assigned reading from Sobell.

DELIVERABLE

- *Modern Operating Systems. Third Edition*: Problems 17, 18, 20 and 29 from Chapter 6, pages 463-466.
- *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Exercises 2, 8, and 9 from Chapter 10, pages 431-433

Assignment 10:

READINGS:

- *Modern Operating Systems. Third Edition*: Chapter 9. *Security,* sections-9.1 – 9.5 (pages 611-659).

DELIVERABLE : I

- *Modern Operating Systems. Third Edition*: Problems 1, 7, 17, and 18 from Chapter 9, pages 713-718.

DELIVERABLE II

- The first purpose of this assignment is to acquaint you with the ***C*** programming language, in case you have not previously acquired any experience with it. In case you are a newbie to programming in ***C,*** I give you below some useful references that are high in quality but cost-free.
- The second purpose is to also acquaint you with the ***Linux*** *C* compiler and to give you some experience in working in the ***Linux*** programming environment, which you have previously read about in Sobell's *A Practical Guide to LINUX Commands, Editors, and Shell Programming*: Chapter 10. *Programming Tools* (pages 387-430). The writing of the program itself is purposely intended **not** to be a significant challenge.
- Write a program in *C* on the CS department's ***Linux*** system for students, ***stu.cs.jmu.edu***
- Your program should take as input the contents of a file containing ASCII **printing** characters, possibly including characters of the following three types:
  - Alphabetic characters, including both upper-case and lower-case;
  - Numeric characters;
  - Special characters[1]
- Your program should output, for each of the three types of characters, a statement indicating the total number of characters of that type present in the input file. It should also state how many of each individual character were present in the file. Your output should look something like this partial example:

> This program, "Count-Characters," was written by Joe (or Jane) Student. It reads in an ASCII text file, counts the number of instances of each printing character present in the file, and reports on the file content.
>
> NUMERIC CHARACTERS:
> The input file contained a total of 57 numeric characters.
>
> There were 4 instances of the numeral '0'.
> There were 3 instances of the numeral '1'.
> There were 4 instances of the numeral '2'.
> There were 0 instances of the numeral '3'.
> There were 9 instances of the numeral '4'.

[1] Please note that the term "Special Characters" refers to all characters that print, but that do not fall into the categories of alphabetic or numeric. Excluded are almost all of the Control Codes, *e.g.,* SOH (Start Of Heading or ^A), STX (Start of Text or ^B), ETX (End of Text or ^C), End of Transmission (EOT or ^D), Enquiry (ENQ or ^E), Acknowledge (ACK or ^F), *etc.*

```
There were 2 instances of the numeral '5'.
There were 12 instances of the numeral '6'.
There was 1 instance of the numeral '7'.
There were 14 instances of the numeral '8'.
There were 8 instances of the numeral '9'.
    etc.
```

- Please note the following points about the sample output that you must follow in order to get full credit for your hard work in producing this program:
    i. The output identifies you as the perpetrator who wrote the program.
    ii. There is a brief description of the function of the program.
    iii. The output is thoroughly organized and also very neat.
    iv. The wording is very thorough, and is totally lacking in abbreviations, slang, and jargon.
- Naming convention for the file containing your source-code program:
    Program2-*YourLastName–FirstInitial*.c
- You will also submit your compiled program, for which the naming convention is:
    Program2-*YourLastName–FirstInitial*.out
- You will also produce an input file for use in testing your program. For this file, the naming convention is
    Program2-*YourLastName–FirstInitial*.input
- Please note especially carefully the details of the file naming, *viz.*:
    i. The first letter only of **"Program2"** is capitalized. Remaining letters are lower-case.
    ii. The first letter only of ***"YourLastName"*** is capitalized. Remaining letters are lower case.
    iii. Your ***"FirstInitial"*** is capitalized.
    iv. The field connectors are hyphens, <u>not</u> underscores.

- Set the protection bits for all three of your files so that you yourself have complete access, but Group and Other have no access.
- Copy the files into the following directory on the CS department's ***Linux*** server:
    /cs/shr/OperatingSystems
- **<u>Check that your files were successfully copied into the target directory</u>.** To accomplish this, you need to issue each of the following commands:

```
$ ls -l /cs/shr/OperatingSystems/Pprogram2-YourLastName–FirstInitial.*
```

RESULT of invoking this command should look something like this for each file**:**

```
-rwx------ 1   9999 csmajor 5432 Nov 31 04:15 Program2-YourLastName–FirstInitial
```

When you check for the presence of your files in the directory, be sure not only to note the presence of the expected file names in the directory listing, but also to note the file size and the protection bits. It has happened more than once in the past that some student deposited in the shared directory files whose names met the specification, but that had zero content, or whose access bits were inappropriate.

Please note that you **<u>must</u>** issue the "*ls -l* " command precisely in the manner shown, in order to obtain the result shown. If you were instead to attempt to obtain a general listing of the directory contents of **/cs/shr/OperatingSystems** by invoking the command:

```
$ ls -l /cs/shr/OperatingSystems
```

RESULT of invoking this command will look something like this**:**

```
ls: /cs/shr/OperatingSystems: Permission denied
```

Useful Internet Resources for Programmers As Yet Inexperienced in C:

1. ***C** Programming Language.* URL: http://en.wikipedia.org/wiki/C_(programming_language)
2. *How **C** Programming Works.* URL: http://www.howstuffworks.com/c.htm and http://computer.howstuffworks.com/c.htm/printable
3. *Introduction to **C** Programming* (University of Leicester course, material available on-line). URL: http://www.le.ac.uk/cc/tutorials/c/
4. *The C Book. Second Edition.* By Mike Banahan, Declan Brady, and Mark Doran. Originally published by Addison-Wesley in 1991 and no longer in print, but now available without charge on the Internet. Based upon the earlier ANSI standard for C, and therefore not up-to-date, but the differences between the older standard and the current standard are small. URL: http://publications.gbdirect.co.uk/c_book/
5. ***C** Programming* (Experimental College of the University of Washington in Seattle; on-line course materials). URL: http://www.eskimo.com/~scs/cclass/
6. ***UNIX** System Calls and Subroutines Using **C*** (University of Cardiff-Wales on-line course materials. Includes general material on the ***C*** programming language in addition to the extensive coverage of System Calls and Subroutines)**.** URL: http://www.cs.cf.ac.uk/Dave/C/
7. *Programming in C. A Tutorial.* By Brian Kernighan (1974). An historic document, produced prior to the first edition of their well-known textbook, and therefore not up-to-date in accordance with the current ANSI standard which is adhered to by modern C compilers. Nevertheless, this is a superb introduction to the features of the C programming language. URL: http://www.lysator.liu.se/c/bwk-tutor.html
8. *ASCII Character Set and its Encoding.* URL: http://www.robelle.com/smugbook/ascii.html
9. *The New* (*i.e.*, the latest) *C Standard*: *An Economic and Cultural Commentary* (2008) Warning: Before you decide to send this document to the printer, know that it is 1,615 pages long.). URL: http://www.coding-guidelines.com/cbook/cbook1_1.pdf
10. *ANSI Draft Standard for C* (07 September 2007). NOTE that this version is "only" 552 pages long. URL: http://www.open-std.org/JTC1/SC22/WG14/www/docs/n1256.pdf

Assignment 11:

READINGS:

- *Modern Operating Systems. Third Edition***:** Chapter 10. *Case Study 1*: *LINUX*, sections 10.1 – 10.4 (pages 719-771).

DELIVERABLE :

- *Modern Operating Systems. Third Edition***:** Exercises 5, 7, 13, and 18 from Chapter 10, pages 808-812.

Optional Extra-Credit Programming Opportunity:

No cost to your grade if you choose not to do it, or if you attempt to do it but it doesn't work.

You may, if you wish, carry out for extra credit the programming project described here. The project entails the extraction from the working ***Linux*** system of information regarding the processor and the other hardware installed in the system, as well as states and conditions of various goings-on within the kernel of the OS.

You have already attained some familiarity with the ***Linux*** command :**ps** which requests the display of the **p**rocessor **s**tatus. This command is summarized in Chapter 8, page 293, Sobell's *A Practical Guide to Linux Commands, Editors, and Shell Programming.* It is described more fully on pages 746-749 in the command reference section of the book. There is another command called **top**, which is described on pages 798-800 in the command reference section, that also requests that certain information about the current state of the machine.

How do these two utilities, **ps** and **top**, obtain the information that they need to carry out your commands to them to display information? Every ***UNIX*** variant contains a pseudo-file-system or virtual file system named **/proc** This consists of a virtual file system; that is,: **/proc** is resident in main memory but can be accessed through the directory tree, just like a file, so as to reveal certain information about the system dynamics.

In addition to Sobell's coverage of the utilities **ps**, and **top,** there are also several internet sites that you can access to obtain information on the **/proc** virtual file system: These include:

1. WARREN, TREVOR (2000). *Exploring* **/proc.** This source provides an excellent overview of the **/proc** virtual file system. URL: http://www.freeos.com/articles/.2879/ [This brief article provides an excellent introduction to the subject.]
2. ***NGUYEN, BINH*** (2004). ***Linux*** *Filesystem Hierarchy.* Section 1.14: **§ 1.14 /proc** Published under the auspices of the ***Linux*** Documentary episode. URL: http://tldp.org/LDP/Linux-Filesystem-Hierarchy/html/index.html [Very thorough coverage.]
3. COMPUTER TECHNOLOGY DOCUMENTATION PROJECT (2000+). *Linux System Configuration and the /proc filesystem.* URL: http://comptechdoc.org/os/linux/howlinuxworks/linux_hlproc.html Rationale and history of the Computer Technology Documentation Project: http://comptechdoc.org/articles/whyctdp.html
4. **man** page for **proc.** This appears in section 5 of the manual.

DELIVERABLE:

- Write a program in the *C* programming language which accepts two integer parameters, reads information stored within the ***Linux*** kernel, and prints out to the screen the following information regarding : the CS department's ***Linux*** server for students, ***stu.cs.mu.edu*** :

  Information About the Hardware

  - The number of processors installed on the server.

- The manufacturer, model name, and speed of the processor.
- The number of CPU cores present in the processor.
- The on-processor cache size
- The number of address bits in the physical memory address within the processor.
- The number of address bits in the virtual memory address within the processor.
- The total amount of Main Memory installed on the server.
- The number of character devices installed.
- The number of block devices installed.
- The names and sizes of the various disk partitions, including the identification of which partition is reserved for swapping..

Information About the Installed Operating System

- The version designation of the ***Linux*** kernel that is installed.
- The date and time when the system was last booted.
- The time elapsed since the last boot.
- The memory addresses assigned to Memory-Mapped I/O devices.
- I/O Ports.

System Dynamics

- The total amount of time that the processors have spent in User Mode since the last boot.
- The total amount of time that the processors have spent in Kernel Mode since the last boot.
- Tottal amount of processor Idle Time since the last boot.
- The number of processes created since the last boot.
- The number of context switches occurring since the last boot.
- The utilization of Main Memory, in each case expressed **both** as a total **and also** as a percentage of all of the installed Main Memory for:
  - Buffers.
  - Page Tables.
  - Cached
  - Designated for Swap and in use.
  - Designated for Swap but free.
  - Allocated and used by processes.
- Statistics on the operation of the disk partitions.
- The number and name(s) of the installed filesystems.
- Information regarding the interrupts serviced by the system. For each interrupt type:
  - The Interrupt Number assigned.
  - The name of the associated device.
  - The total number of interrupts that occurred and were serviced by all of the CPUs.

- The required information should be printed out **in the order specified.**
- The name of your program will be: **Program3.** The two parameters with which you will call it are, in order: the sampling interval in seconds, and the total sampling time in minutes.
- The full file specification for your source-code and object-code files is:

  Program3-*YourLastName–FirstInitial*.c

  Program3-*YourLastName-FirstInitial*r.out

- I will not specify a detailed format for your program output. **However,** I **will** expect your output to be **neat,** and I will also expect it to be **clearly labeled.** Avoid quirky abbreviations, *e.g.*, "**numcxs**".

The plain-English-language term "**Number of Context Switches**" is so much clearer and more expressive, and is also self-explanatory.

- You will also run your program, and redirect its output from `stdio` into a file, which you will name:
  Program3-*YourLastName–FirstInitial*.txt
- Set the protection bits for all three files so that you have complete access, but group and other have no access.
- Copy all three files into **/cs/shr/OperatingSystems**
- Check that your files are all present in **/cs/shr/OperatingSystems**, that the protection bits of the copies in **/cs/shr/OperatingSystems** are appropriately set, and that the copied files are of the correct size.