Assignment for 3rd Year 1stSemester Students
IT/PC/B/S/313

Assignment 1A: (6 Marks) [CO1]
Familiarity with User and System level OS Commands

Run the commands and validate the output
Example of Output Validation

$ pwd [Unix commands are Case Sensitive]
/home/it13/it1340

1) who:-It displays the information about all the users who have logged into the system currently
2) whoami:- It displays Current username, Terminal number, date and time at which user logged into the system
3) pwd:- It displays current working directory
4) date:- It displays system date and time
5) ls - It lists the files and directories stored in the current directory. To list the files in a directory use the following syntax: $ls dirname
6) mkdir – It is used to create directories by using the command: $mkdir dirname
7) clear- It clears the screen
8) cd - It is used to change the current working directory to any other directory specified
9) cd.. -This command is used to come out from the current working directory.
10) rmdir - Directories can be deleted using the **rmdir** command - $rmdir dirname
11) cat – It displays the contents of a file - $cat filename
12) cp - It is used to copy a file - $ cp source_file destination_file
13) mv- It is used to change the name of a file - $ mv old_file new_file
14) rm – It is used to delete an existing file - $ rm filename
15) stat- It is used to display file or file system status - $ stat filename
16) stty – Change and print terminal line settings. Its option **– "stty –a"** prints all current settings in human readable form
17) tty – It prints the filename of the terminal connected to standard input.
18) uname –It prints system information
19) umask – It specifies user file creation mask, implying which of the 3 permissions are to be denied to the owner,group and others.
20) find – It searches for files in a directory hierarchy
21) sort – It sorts the lines of text files
22) ps - It displays information about the current processes.
23) chmod 777 file1 - gives full permission to owner, group and others
24) chmod o-w file1 - Removes write permission for others.

Assignment 1B: (4 Marks) [CO1]
Program to get and set environment variables using system calls

Problem Definition

Program to GET and SET the Environment variable and to know use of getenv and setenv system calls

UNIX ENVIRONMENT VARIABLES

Variables are a way of passing information from the shell to programs when you run them. Programs look "in the environment" for particular variables and if they are found will use the values stored. Some are set by the system, others by you, yet others by the shell, or any program that loads another program. Standard UNIX variables are split into two categories, environment variables and shell variables. In broad terms, shell variables apply only to the current instance of the shell and are used to set short-term working conditions; environment variables have a farther reaching significance, and those set at login are valid for the duration of the session. By convention, environment variables have UPPER CASE and shell variables have lower case names.

Display the following environment variables using getenv call:

- USER (your login name)
- HOME (the path name of your home directory)
- HOST (the name of the computer you are using)
- ARCH (the architecture of the computer's processor)
- DISPLAY (the name of the computer screen to display X windows)
- PRINTER (the default printer to send print jobs)
- PATH (the directories the shell should search to find a command)

Syntax: Char *getenv(const char *name);
The getenv() function searches the environment list to find the Environment variable *name*, and returns a pointer to the corresponding *value* string.

Now, Set two new environment variables and display them.

Syntax: int setenv(const char * envname,const char * enval,int overwrite)
The setenv() function adds the variable *name* to the environment with the value *value*, if *name* does not already exist. If *name* does exist in the environment, then its value is changed to *value* if *overwrite* is nonzero; if *overwrite* is zero, then the value of *name* is not changed (and setenv() returns a success status).

Note: make sure you don't modify the already generated system's environment variable like HOME, HOST etc.