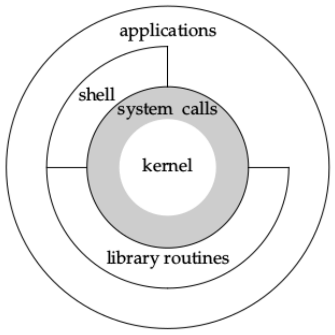
List of Lab Assignments

Lab 0(a) : Getting familiar with Linux Environment

In this lab we will gain the knowledge of a Linux based operating system (Process management, File system, Memory Mnagament, OS Design & Architecture) and to work in this environment with the help of commands and their actual implementations.



Architecture of the UNIX operating system

- UNIX, Linux and the GNU Project
- Users and Groups

```
#include "apue.h"
int main(void)
{
printf("uid = %d, gid = %d\n", getuid(), getgid());
exit(0);
}
```

- Processes
- Files
- Directory Layout, Pathnames and Symbolic Links
  - Relative and Absolute Pathnames
  - Symbolic Links
- Basic Commands
  - The Bash Shell
  - ls [with all the options]: Below is a sample C program how ls is implemented in OS.

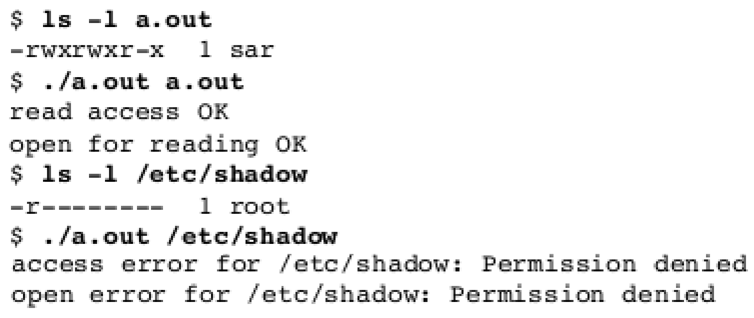
```
#include "apue.h"
#include <dirent.h>
int
main(int argc, char *argv[])
{
        DIR       *dp;
        struct dirent *dirp;
        if (argc != 2)
                err_quit("usage: ls directory_name");
        if ((dp = opendir(argv[1])) == NULL)
                err_sys("can't open %s", argv[1]);
        while ((dirp = readdir(dp)) != NULL)
```

```
                printf("%s\n", dirp->d_name);
        closedir(dp);
        exit(0);
}
```

- cp [with all the options]
- Essential: pwd, cd, rm, mv, mkdir, cat, less, file, fins, locate, chmod, gzip, gunzip, tar, df, head, tail, date, grep, kill
- Background and foreground jobs
- Choosing a suitable editor: gedit, vi, vim, nano, emac
- Sample Progrm to show the use of the access function

```
#include "apue.h"
#include <fcntl.h>
int
main(int argc, char *argv[])
{
if (argc != 2)
err_quit("usage: a.out <pathname>");
if (access(argv[1], R_OK) < 0)
err_ret("access error for %s", argv[1]);
else
printf("read access OK\n");
if (open(argv[1], O_RDONLY) < 0)
err_ret("open error for %s", argv[1]);
else
printf("open for reading OK\n");
exit(0);
}
```

Sample Output:



```
$ ls -l a.out
-rwxrwxr-x  1 sar
$ ./a.out a.out
read access OK
open for reading OK
$ ls -l /etc/shadow
-r--------  1 root
$ ./a.out /etc/shadow
access error for /etc/shadow: Permission denied
open error for /etc/shadow: Permission denied
```

Assignment :

1. Create a user in command window and provide the sudo access to it.

2. Access the manual of every command using 'man <command>' and execute the commands providing different types of arguments. And record the returned results.

3. Write a C-program to implement 'cat' command.

APPENDIX – I

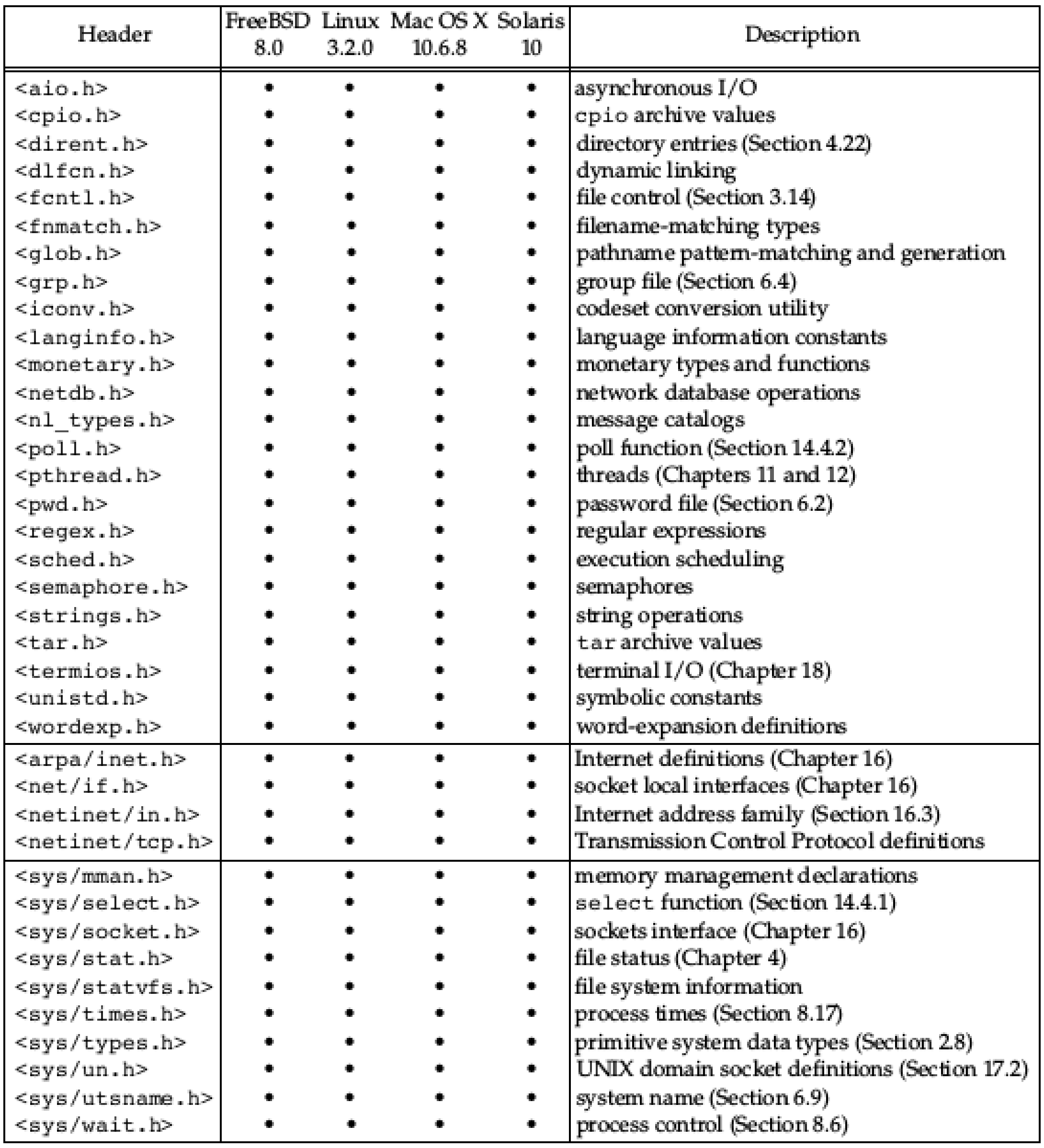
$$

In order to run the given examples you have to set up the environment first:

1. Download the zip file from: https://drive.google.com/file/d/1pABtdJnSnHSc4g9T64BteiT5yz_c_93a/view?usp=sharing
2. Extract the tar.gz file and you will get a folder called ‘apue’ . Go to that folder in command window and run a command : *make*
3. Then compile the program (*.c file) using this format:

gcc demo.c -o demo -I /path-to-this-folder/apue/include/ -L /path-to-this-folder/apue/lib/ -lapue

$$



Header	FreeBSD 8.0	Linux 3.2.0	Mac OS X 10.6.8	Solaris 10	Description
`<aio.h>`	•	•	•	•	asynchronous I/O
`<cpio.h>`	•	•	•	•	cpio archive values
`<dirent.h>`	•	•	•	•	directory entries (Section 4.22)
`<dlfcn.h>`	•	•	•	•	dynamic linking
`<fcntl.h>`	•	•	•	•	file control (Section 3.14)
`<fnmatch.h>`	•	•	•	•	filename-matching types
`<glob.h>`	•	•	•	•	pathname pattern-matching and generation
`<grp.h>`	•	•	•	•	group file (Section 6.4)
`<iconv.h>`	•	•	•	•	codeset conversion utility
`<langinfo.h>`	•	•	•	•	language information constants
`<monetary.h>`	•	•	•	•	monetary types and functions
`<netdb.h>`	•	•	•	•	network database operations
`<nl_types.h>`	•	•	•	•	message catalogs
`<poll.h>`	•	•	•	•	poll function (Section 14.4.2)
`<pthread.h>`	•	•	•	•	threads (Chapters 11 and 12)
`<pwd.h>`	•	•	•	•	password file (Section 6.2)
`<regex.h>`	•	•	•	•	regular expressions
`<sched.h>`	•	•	•	•	execution scheduling
`<semaphore.h>`	•	•	•	•	semaphores
`<strings.h>`	•	•	•	•	string operations
`<tar.h>`	•	•	•	•	tar archive values
`<termios.h>`	•	•	•	•	terminal I/O (Chapter 18)
`<unistd.h>`	•	•	•	•	symbolic constants
`<wordexp.h>`	•	•	•	•	word-expansion definitions
`<arpa/inet.h>`	•	•	•	•	Internet definitions (Chapter 16)
`<net/if.h>`	•	•	•	•	socket local interfaces (Chapter 16)
`<netinet/in.h>`	•	•	•	•	Internet address family (Section 16.3)
`<netinet/tcp.h>`	•	•	•	•	Transmission Control Protocol definitions
`<sys/mman.h>`	•	•	•	•	memory management declarations
`<sys/select.h>`	•	•	•	•	select function (Section 14.4.1)
`<sys/socket.h>`	•	•	•	•	sockets interface (Chapter 16)
`<sys/stat.h>`	•	•	•	•	file status (Chapter 4)
`<sys/statvfs.h>`	•	•	•	•	file system information
`<sys/times.h>`	•	•	•	•	process times (Section 8.17)
`<sys/types.h>`	•	•	•	•	primitive system data types (Section 2.8)
`<sys/un.h>`	•	•	•	•	UNIX domain socket definitions (Section 17.2)
`<sys/utsname.h>`	•	•	•	•	system name (Section 6.9)
`<sys/wait.h>`	•	•	•	•	process control (Section 8.6)