1 OPERATING SYSTEMS LABORATORY X - Files and Directories I

Examples&Exercises:

- Compile and run the code.
- Analyze the code and output.

1. Try the following commands;
   - ls –i
   - stat *
   - stat /dev/sda1
   - istat inode_number /dev/sda1 (if *istat* command is available)
2. A code example for for printing out structure members of files; code54.c
   - Give the command; code54 *
   - Modify the code to print out other structure members of files as

```
% st_mode (the type and mode of the file)
% st_ino
% st_dev
% st_rdev
% st_nlink
% st_uid
% st_gid
% st_size
% st_atime
% st_mtime
% st_ctime
```

3. **File Types**; *stat, fstat*, and *lstat* Functions; code48.c (do not forget to download also the file, apue.h)
   - Given a pathname, the *stat* function returns a structure of information about the named file. The *fstat* function obtains information about the file that is already open on the descriptor (file descriptor). The *lstat* function is similar to *stat*, but when the named file is a symbolic link, *lstat* returns information about the symbolic link, not the file referenced by the symbolic link.

- Study the following command in detail;

  ```
  $ man 2 stat
  ```

- The type of a file is encoded in the *st_mode* member of the *stat* structure.
- We can determine the file type with the macros in $< sys/stat.h >$ (S_IS***()).
- Compile the code and execute as the following;

  ```
  ./code48 /etc/passwd /etc /dev/initctl /dev/log /dev/tty \
  /dev/sda1 /dev/cdrom
  ```

- Study the output.
- Modify the code such that
  - it will be able to print out the information (present status, values, etc.) about all the fields in the *stat* structure (man 2 stat; see *struct stat*) not only the field *protection*.

4. **File Access**; *chmod* and *fchmod* Functions; code49.c

   - These two functions allow us to change the file access permissions for an existing file. The *chmod* function operates on the specified file, whereas the *fchmod* function operates on a file that has already been opened.
   - To change the permission bits of a file, the effective user ID of the process must be equal to the owner ID of the file, or the process must have superuser permissions (we will not discuss the effective user ID and superuser concepts).
   - The mode constants for *chmod* functions, from $< sys/stat.h >$ are S_IRWXU, S_IRUSR, S_IWUSR, S_IXUSR, S_IRWXG, S_IRGRP, S_IWGRP, S_IXGRP, S_IRWXO, S_IROTH, S_IWOTH, S_IXOTH.
   - They are self-explanatory, but as an example S_IRWXU stands for read, write, and execute by user (owner).
   - Compile the code
   - Before execution create two files as the following;

     ```
     -rw------- 1 ozdogan ozdogan 0 May  8 02:27 foo
     -rw-rw-rw- 1 ozdogan ozdogan 0 May  8 02:27 bar
     ```

     by using *touch* and *chmod* commands.
   - Execute the code as

```
$./code49
```

- Observe how the final state of the two files are changed after running the program.
- Modify the code to use some other mode constants, meanwhile make a search for the S_ISGID.
- Note that the time and date listed by the *ls* command did not change after we ran the program.By default, the *ls -l* lists the time when the contents of the file were last modified.

5. **Reading Directories**; File tree walk code50.c (do not forget to download also the files, code51.c, code52.c)

   - Directories can be read by anyone who has access permission to read the directory. (But only the kernel can write to a directory, to preserve file system sanity)
   - We'll use these directory routines (opendir, readdir, etc.) to write a program that traverses a file hierarchy. The goal is to produce the count of the various types of files.
   - The program takes a single argument the starting *pathname* and recursively descends the hierarchy from that point.
   - Apply the following commands;

     ```
     $ gcc -c code52.c
     $ gcc -c code51.c
     $ gcc -o code50 code50.c code51.o code52.o
     $ ./code50 .
     $ ./code50 ..
     ```

     also try with different paths.
   - You can also use this program to see the file profile in your system by executing with root privileges and with the pathname "/".
   - Also see

     ```
     $ man nftw
     ```