## Programming Exercise H - Files and Directories

### Assignment Objectives

* Practice using the getcwd, chdir, opendir, readdir, and closedir functions for handling directories and directory contents
* Practice using the stat function for obtaining file and directory attributes
* Practice using the ctime function for converting numeric time data into an easily readable text format
* Practice using predefined POSIX macros to extract the permissions and type information from the mode attribute of a file or directory

### Assignment Summary

In this assignment, you shall develop a C program that displays specific status information, in a columnar fashion with single line spacing, for all of the entries in a designated UNIX/LINUX directory. See the **sample-runs.txt** file for sample runs that illustrate both the error messages produced and the format of the displayed information. Also see the section below entitled **Contents of each Column** for more details.

The entry names shall come from either the directory specified on the command line when the program is executed, or from the current working directory if no entry is given on the command line. In other words, the user shall NOT be prompted to enter any information. Here are two sample command line invocations for the program:

a.out

a.out  /etc

### Contents of Each Column

* **Type** - one of the character phrases shown in the right column or "Unknown type" if none of the Boolean macros is true; the phrase is determined in your program by checking the return value from the respective Boolean macro in the left column. Print the Type phrase left justified.

|  |  |
| --- | --- |
| **Boolean Macro** | **Type** |
| S\_ISBLK(mode\_t mode) | Block special |
| S\_ISCHR(mode\_t mode) | Character special |
| S\_ISDIR(mode\_t mode) | Directory |
| S\_ISFIFO(mode\_t mode) | Pipe or FIFO |
| S\_ISLNK(mode\_t mode) | Symbolic link |
| S\_ISREG(mode\_t mode) | Regular file |
| S\_ISSOCK(mode\_t mode) | Socket |

* **Size** - integer value for the number of bytes in a file. Print the size value right justified.
* **Access** - user, group, and world permissions obtained from "bitwise and" operations on the mode field, and displayed in the same manner as the approach used by the UNIX ls command (e.g., rwx--x--x)
* **Status changed** - the month, date, and time information for the **time of last file status change**
* **Name** - the physical entry name obtained when reading the directory contents. Print the name left justified.

### Design and Implementation Constraints

* Download the **ls-driver-skeleton.c** file and rename it to **ls-driver.c**
* Use a modular design consisting of specific user-defined functions to formulate and display the various columns of information required on each entry output line
* Declare no global variables
* Print all output to Standard Out
* Use no tab characters in the program output
* In the algorithm for determining the permissions of a file or directory, use the constants listed in Chapter 4 of the Robbins slides
* Have your program respond to any error conditions detected (i.e., returned) by the POSIX built-in functions. Do this by calling the perror() function and then exiting the program.
* Except for error messages, the actual output of the program may use any of the Standard C output functions such as printf() and putchar()