```
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab05$ ./a.out test
34058472182250602 - HelloWorld.c - File Type: Regular
10133099162019517 - Journal - File Type: Regular
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab05$ ./a.out test
34058472182250602 - HelloWorld.c - File Type: Regular
10133099162019517 - Journal - File Type: Regular
62768919806487985 - test1 - File Type: Directory
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab05$
```

```c
/***************************************
 *
 *             lab5.c
 *
 *  Starting point for laboratory 5 solution
 *  The single parameter to this progra is
 *  the path to the directory to be listed.
 ***************************************/
//Lab05
 //Adam Green, 100653971
#define _GNU_SOURCE
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <sys/types.h>
#include <dirent.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/syscall.h>
#include <fcntl.h>


/*
 *  This procedure prints the contents of a directory.
 *  If a directory entry is another directory it calls
 *  itself recursively the list the contents of that
 *  directory.
 *
 *  The parameters to this procedure are:
 *    dir - the directory stream for the directory
 *    indent - the level of the directory in the listing
 *    base - the filename of the directory
```

```
 */

void dumpDir(DIR *dir, int indent, char *base) {
    struct dirent *entry;   // the current directory entry
    char *name;     // the name of the entry
    int type;       // the type of the directory entry
    DIR *newdir;        // directroy stream for recursive listing
    char *dirname;      // full name of sub-directory
    int len;
    int i;


    entry = readdir(dir);
    while(entry != NULL) {
        name = entry->d_name;
        type = entry->d_type;
        if(name[0] != '.') {  // skip filenames that start with .
            for(i=0; i<indent; i++)
                printf("%s", " ");
            printf("%ld - %s - File Type: ",entry->d_ino, name);
            if (entry->d_type == DT_REG) {
                printf("Regular\n");
            } else if (entry->d_type == DT_DIR) {
                printf("Directory\n");
            } else if (entry->d_type == DT_FIFO) {
                printf("FIFO\n");
            } else if (entry->d_type == DT_SOCK) {
                printf("Socket\n");
            } else if (entry->d_type == DT_LNK) {
                printf("Symlink\n");
            } else if (entry->d_type == DT_BLK) {
                printf("Block dev\n");
            } else if (entry->d_type == DT_CHR) {
                printf("Char dev\n");
            }

                if(type == DT_DIR) {  // recursive directory listing
                len = strlen(base) + strlen(name) + 2;
                dirname = (char*) malloc(len);
                strcpy(dirname, base);
```

```
                strcat(dirname, "/");
                strcat(dirname, name);
                newdir = opendir(dirname);
                dumpDir(newdir, indent+2, dirname);
                closedir(newdir);
                free(dirname);
            }
        }
        entry = readdir(dir);
    }
}

int main(int argc, char **argv) {
    DIR *dir;

    if(argc != 2) {
        printf("usage: lba5 directory\n");
        exit(1);
    }

    dir = opendir(argv[1]);
    if(dir == NULL) {
        printf("can't open directory: %s\n", argv[1]);
        exit(1);
    }
    dumpDir(dir,0,argv[1]);
    closedir(dir);
}
```