1. vi was used for the entire homework.

2. mv xyz vader

3. cp vader galaxy

4. ls /proc | grep -c -v [^0-9]

5.
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdint.h>

// Includes for stat() sys call
#include <sys/types.h>
#include <sys/stat.h>
#include <unistd.h>

// Function prototypes
int get_stat(char *path, struct stat *buf);
void get_mode(struct stat *buf);

/*
 * Main function takes passed filename and prints the file type and mode
 * Note: Only works with files found in the same directory as this
 * program is found.
 */
int main(int argc, char *argv[])
{
        struct stat buffer;
        char path[256] = "/home/pi/ece331/hw02/";
        int err;

        // Some error checking
        if (argc < 2) {
                printf("No file name was entered, please enter a filename\n\n");
                return -1;
        } else if (argc > 2) {
                printf("Too many arguements, please enter a single filename\n\n");
                return -1;
        }
        if ((strlen(path) + strlen(argv[1])) > 255) {
                printf("Filename too long");
                return -1;
        }

        // Concatenate filename to path
        strcat(path, argv[1]);
        printf("%s\n", path);

        err = get_stat(path, &buffer);
        if (err < 0) {
                return -1;
        }

        get_mode(&buffer);

        return 0;
}

// Get a stat structure using the specified path
int get_stat(char *path, struct stat *buf)
{
        int err;
        err = stat(path, buf);
        if (err < 0) {
                printf("Error using stat().\n");
                return -1;
        }

        return 0;
}

/*
 * Determine the filetype and print appropriate message
 * using POSIX macors. Also display mode field becuase
 * it contains more information such as file permissions
 */
void get_mode(struct stat *buf)
{
        int mode_var = buf->st_mode;

        if (S_ISREG(mode_var)) {
                printf("File type: Regular File.\n");
                printf("File mode: %d\n", mode_var);
        } else if (S_ISDIR(mode_var)) {
                printf("File type: Directory.\n");
                printf("File mode: %d\n", mode_var);
        } else if (S_ISCHR(mode_var)) {
                printf("File type: Character device.\n");
                printf("File mode: %d\n", mode_var);
        } else if (S_ISBLK(mode_var)) {
                printf("File type: Block Device.\n");
                printf("File mode: %d\n", mode_var);
        } else if (S_ISFIFO(mode_var)) {
                printf("File type: FIFO (named pipe).\n");
                printf("File mode: %d\n", mode_var);
        } else if (S_ISLNK(mode_var)) {
                printf("File type: Symbolic link.\n");
                printf("File mode: %d\n", mode_var);
        } else if (S_ISSOCK(mode_var)) {
                printf("File type: Socket.\n");
                printf("File mode: %d\n", mode_var);
        } else {
                printf("File type can not be determined.\n");
                printf("File mode: %d\n", mode_var);
        }
}
```

6.

```
#include <stdio.h>
#include <string.h>

// Main functions prints length of passed string
int main (int argc, char *argv[])
{
        if (argc < 2) {
                printf("Please enter a string.\n");
                return -1;
        } else if (argc > 2) {
                printf("String includes spaces, please pass as single arguement.\n");
                return -1;
        }

        printf("String length: %d\n", strlen(argv[1]));
        return 0;
}
```

7.

8a. The /sbin directory contains mostly executables and links to executables owned by root. It is essentially full of administrative tools.

8b. /usr/share requires a directory containing static data i.e. data that doesn't need to be modified. If only using a single file it should be placed in the /usr/share/misc subdirectory.

9.
```
enscript --header='$n %D $C|$%|Tyler Punch'
       hw02.txt -o temp | ps2pdf temp Punch-Tyler-ECE331-HW02.pdf
```