File Execises

Exercize 1

Write a C program to list al files of the current directory.

Solution:

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

int main()
{
    DIR *folder;
    struct dirent *entry;
    int files = 0;

    folder = opendir(".");
    if(folder == NULL)
    {
        perror("Unable to read directory");
        return(1);
    }

    while( (entry=readdir(folder)) )
    {
        files++;
        printf("File %3d: %s\n",
                files,
                entry->d_name
              );
    }

    closedir(folder);

    return(0);
}
```

Exercize 2

Write a C program to list all files and subdirectories of a path provided by command-line argument. The program must list recursively all subdirectories.

Solution:

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
```

```
#include <string.h>
#include <errno.h>
#include <dirent.h>
#include <limits.h>

void list_dir(const char * dir_name) {
  DIR * d;

  /* Open the directory specified by "dir_name". */

  d = opendir(dir_name);

  /* Check it was opened. */
  if (!d) {
    fprintf(stderr, "Cannot open directory '%s': %s\n",
      dir_name, strerror(errno));
    exit(EXIT_FAILURE);
  }
  while (1) {
    struct dirent * entry;
    const char * d_name;

    /* "Readdir" gets subsequent entries from "d". */
    entry = readdir(d);
    if (!entry) {
      /* There are no more entries in this directory, so break
         out of the while loop. */
      break;
    }
    d_name = entry -> d_name;
    /* Print the name of the file and directory. */
    printf("%s/%s\n", dir_name, d_name);

    if (entry -> d_type & DT_DIR) {

      /* Check that the directory is not "d" or d's parent. */

      if (strcmp(d_name, "..") != 0 &&
        strcmp(d_name, ".") != 0) {
        int path_length;
        char path[PATH_MAX];

        path_length = snprintf(path, PATH_MAX,
          "%s/%s", dir_name, d_name);
        printf("%s\n", path);
        if (path_length >= PATH_MAX) {
          fprintf(stderr, "Path length has got too long.\n");
          exit(EXIT_FAILURE);
        }
        /* Recursively call "list_dir" with the new path. */
        list_dir(path);
      }
    }
  }
  /* After going through all the entries, close the directory. */
  if (closedir(d)) {
    fprintf(stderr, "Could not close '%s': %s\n",
      dir_name, strerror(errno));
    exit(EXIT_FAILURE);
  }
}

int main(int argc, char * argv[]) {
  if (argc != 2) {
    fprintf(stderr, "Usage: pgrm <pathname>\n");
    exit(1);
  }
  list_dir(argv[1]);
  return 0;
}
```