**Experient:14-Construct a C program to organise the file using a single level directory.**

**Aim:**

The aim of this program is to demonstrate how to organize files using a **single-level directory structure** in C. A single-level directory is a simple file structure in which all files are stored directly in one directory, with no subdirectories. This program will create a directory, create some files within it, and then list the files contained in that directory.

**Procedure:**

1. **Create a Directory**: The program will create a directory using mkdir() system call.
2. **Create Files in the Directory**: We will create files inside the directory using file operations like fopen().
3. **List Files in the Directory**: The program will use opendir() and readdir() system calls to list the contents of the directory.
4. **Error Handling**: Proper error handling will be implemented to ensure the creation of the directory and files, and listing of files occurs successfully.

**Steps Involved:**

1. **Create a Directory**: Use the mkdir() function to create a directory where files will be stored.
2. **Create Files**: Use fopen() to create files inside the directory.
3. **List Files**: Use opendir() and readdir() to list the contents of the directory.
4. **Clean Up**: Close files and directories properly.

**C Program to Organize Files Using a Single-Level Directory:**

c

Copy code

#include <stdio.h>

#include <stdlib.h>

#include <dirent.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

#define DIRECTORY\_NAME "single\_level\_directory"

// Function to create a directory

void create\_directory(const char \*dir\_name) {

if (mkdir(dir\_name, 0755) == -1) {

perror("Error creating directory");

exit(EXIT\_FAILURE);

} else {

printf("Directory '%s' created successfully.\n", dir\_name);

}

}

// Function to create a file inside the directory

void create\_file(const char \*dir\_name, const char \*file\_name) {

char path[256];

FILE \*file;

// Construct the full path for the file

snprintf(path, sizeof(path), "%s/%s", dir\_name, file\_name);

// Create and open the file for writing

file = fopen(path, "w");

if (file == NULL) {

perror("Error creating file");

exit(EXIT\_FAILURE);

}

fprintf(file, "This is file: %s\n", file\_name);

fclose(file);

printf("File '%s' created successfully in directory '%s'.\n", file\_name, dir\_name);

}

// Function to list the files in the directory

void list\_files(const char \*dir\_name) {

DIR \*dir;

struct dirent \*entry;

// Open the directory

dir = opendir(dir\_name);

if (dir == NULL) {

perror("Error opening directory");

exit(EXIT\_FAILURE);

}

printf("\nListing files in directory '%s':\n", dir\_name);

// Read and list the files in the directory

while ((entry = readdir(dir)) != NULL) {

if (entry->d\_type == DT\_REG) {

printf("%s\n", entry->d\_name);

}

}

closedir(dir);

}

int main() {

// Create a directory

create\_directory(DIRECTORY\_NAME);

// Create files in the directory

create\_file(DIRECTORY\_NAME, "file1.txt");

create\_file(DIRECTORY\_NAME, "file2.txt");

create\_file(DIRECTORY\_NAME, "file3.txt");

// List the files in the directory

list\_files(DIRECTORY\_NAME);

return 0;

}

Output:

A screenshot of a computer code

Description automatically generated