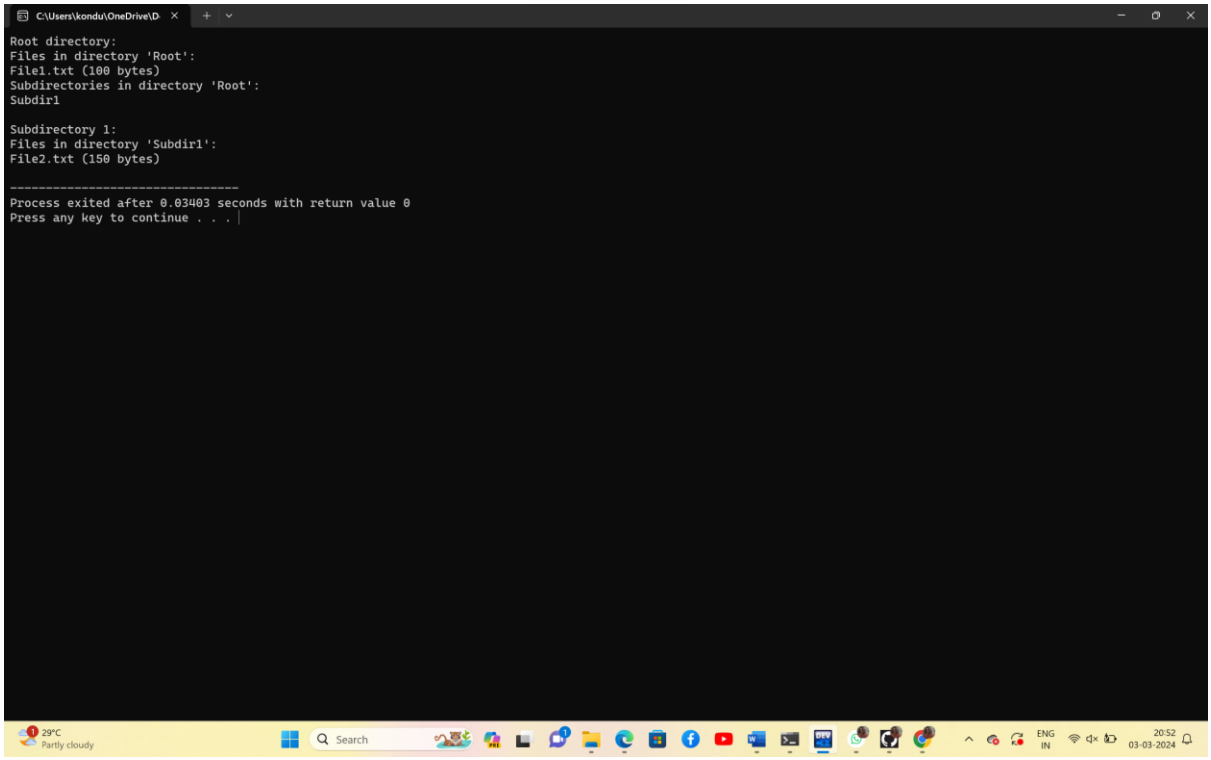
```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_FILES_PER_DIR 100
#define MAX_DIRS 100
#define MAX_FILENAME_LENGTH 50
#define MAX_DIR_NAME_LENGTH 50

typedef struct {
    char name[MAX_FILENAME_LENGTH];
    int size;
} File;

typedef struct Directory {
    char name[MAX_DIR_NAME_LENGTH];
    File files[MAX_FILES_PER_DIR];
    int fileCount;
    struct Directory* subdirs[MAX_DIRS];
    int subdirCount;
} Directory;

Directory root;

void initializeRootDirectory() {
    strcpy(root.name, "Root");
    root.fileCount = 0;
    root.subdirCount = 0;
}

Directory* createDirectory(const char* name) {
```

```
    if (root.subdirCount < MAX_DIRS) {
        Directory* newDir = (Directory*)malloc(sizeof(Directory));
        strcpy(newDir->name, name);
        newDir->fileCount = 0;
        newDir->subdirCount = 0;
        root.subdirs[root.subdirCount++] = newDir;
        return newDir;
    } else {
        printf("Cannot create directory. Maximum directories reached.\n");
        return NULL;
    }
}

void createFile(Directory* dir, const char* name, int size) {
    if (dir->fileCount < MAX_FILES_PER_DIR) {
        strcpy(dir->files[dir->fileCount].name, name);
        dir->files[dir->fileCount].size = size;
        dir->fileCount++;
    } else {
        printf("Cannot create file. Maximum files per directory reached.\n");
    }
}

void listFiles(Directory* dir) {
    printf("Files in directory '%s':\n", dir->name);
    for (int i = 0; i < dir->fileCount; i++) {
        printf("%s (%d bytes)\n", dir->files[i].name, dir->files[i].size);
    }
}

void listDirectories(Directory* dir) {
```

```
    printf("Subdirectories in directory '%s':\n", dir->name);
    for (int i = 0; i < dir->subdirCount; i++) {
        printf("%s\n", dir->subdirs[i]->name);
    }
}

void deleteFile(Directory* dir, const char* name) {
    for (int i = 0; i < dir->fileCount; i++) {
        if (strcmp(dir->files[i].name, name) == 0) {
            for (int j = i; j < dir->fileCount - 1; j++) {
                strcpy(dir->files[j].name, dir->files[j + 1].name);
                dir->files[j].size = dir->files[j + 1].size;
            }
            dir->fileCount--;
            return;
        }
    }
    printf("File '%s' not found in directory '%s'.\n", name, dir->name);
}

void deleteDirectory(Directory* parentDir, const char* name) {
    for (int i = 0; i < parentDir->subdirCount; i++) {
        if (strcmp(parentDir->subdirs[i]->name, name) == 0) {
            free(parentDir->subdirs[i]);
            for (int j = i; j < parentDir->subdirCount - 1; j++) {
                parentDir->subdirs[j] = parentDir->subdirs[j + 1];
            }
            parentDir->subdirCount--;
            return;
        }
    }
```

```
    printf("Directory '%s' not found in directory '%s'.\n", name, parentDir->name);
}

Directory* findDirectory(Directory* dir, const char* name) {
    for (int i = 0; i < dir->subdirCount; i++) {
        if (strcmp(dir->subdirs[i]->name, name) == 0) {
            return dir->subdirs[i];
        }
    }
    return NULL;
}

int main() {
    initializeRootDirectory();
    Directory* subdir1 = createDirectory("Subdir1");
    createFile(&root, "File1.txt", 100);
    createFile(subdir1, "File2.txt", 150);

    printf("Root directory:\n");
    listFiles(&root);
    listDirectories(&root);

    printf("\nSubdirectory 1:\n");
    listFiles(subdir1);

    return 0;
}
```


C:\Users\kondu\OneDrive\D
Root directory:
Files in directory 'Root':
File1.txt (100 bytes)
Subdirectories in directory 'Root':
Subdir1
Subdirectory 1:
Files in directory 'Subdir1':
File2.txt (150 bytes)

Process exited after 0.03403 seconds with return value 0
Press any key to continue . . .
29°C
Partly cloudy
Search
ENG
IN
20:52
03-03-2024