

14. Construct a C program to organise the file using a single level directory.

Aim:

To construct a C program that organizes files using a single-level directory. The program will simulate basic file operations such as creating, displaying, and deleting files within the directory.

Algorithm:

1. **Create a Directory:** Simulate creating a directory to hold files.
2. **Add Files:** Simulate adding files to the directory.
3. **Display Files:** Display all the files currently in the directory.
4. **Delete Files:** Allow deletion of specific files from the directory.
5. **Search Files:** Allow the user to search for a specific file by name.

Procedure:

1. Define a structure for representing a file with its name and status (if it's in the directory).
2. Implement functions to create a file, delete a file, display all files, and search for a specific file.
3. Use an array to simulate the directory and store file information.
4. Implement a menu-driven interface to allow users to interact with the directory.

CODE:

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

#define MAX_FILES 100
#define MAX_NAME_LENGTH 50

typedef struct {
    char name[MAX_NAME_LENGTH];
    int isOccupied; // 1 if occupied, 0 if free
} File;
```

```

void addFile(File directory[], int *fileCount) {
    if (*fileCount >= MAX_FILES) {
        printf("Directory is full. Cannot add more files.\n");
        return;
    }

    char fileName[MAX_NAME_LENGTH];
    printf("Enter the name of the file to add: ");
    scanf("%s", fileName);

    // Check if file already exists
    for (int i = 0; i < *fileCount; i++) {
        if (directory[i].isOccupied && strcmp(directory[i].name, fileName) == 0) {
            printf("File already exists.\n");
            return;
        }
    }

    // Add new file
    strcpy(directory[*fileCount].name, fileName);
    directory[*fileCount].isOccupied = 1;
    (*fileCount)++;
    printf("File '%s' added successfully.\n", fileName);
}

void searchFile(File directory[], int fileCount) {
    char fileName[MAX_NAME_LENGTH];
    printf("Enter the name of the file to search: ");
    scanf("%s", fileName);

    for (int i = 0; i < fileCount; i++) {
        if (directory[i].isOccupied && strcmp(directory[i].name, fileName) == 0) {
            printf("File '%s' found in the directory.\n", fileName);
            return;
        }
    }
    printf("File '%s' not found.\n", fileName);
}

void deleteFile(File directory[], int *fileCount) {
    char fileName[MAX_NAME_LENGTH];

```

```

printf("Enter the name of the file to delete: ");
scanf("%s", fileName);

for (int i = 0; i < *fileCount; i++) {
    if (directory[i].isOccupied && strcmp(directory[i].name, fileName) == 0) {
        directory[i].isOccupied = 0;
        printf("File '%s' deleted successfully.\n", fileName);
        return;
    }
}
printf("File '%s' not found.\n", fileName);
}

void listFiles(File directory[], int fileCount) {
    printf("\nFiles in the directory:\n");
    int empty = 1;

    for (int i = 0; i < fileCount; i++) {
        if (directory[i].isOccupied) {
            printf("%d. %s\n", i + 1, directory[i].name);
            empty = 0;
        }
    }

    if (empty) {
        printf("No files in the directory.\n");
    }
}

int main() {
    File directory[MAX_FILES] = {0};
    int fileCount = 0;
    int choice;

    while (1) {
        printf("\nSingle-Level Directory Management System\n");
        printf("1. Add File\n");
        printf("2. Search File\n");
        printf("3. Delete File\n");
        printf("4. List Files\n");
        printf("5. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);
    }
}

```

```

switch (choice) {
    case 1:
        addFile(directory, &fileCount);
        break;
    case 2:
        searchFile(directory, fileCount);
        break;
    case 3:
        deleteFile(directory, &fileCount);
        break;
    case 4:
        listFiles(directory, fileCount);
        break;
    case 5:
        printf("Exiting the program.\n");
        return 0;
    default:
        printf("Invalid choice. Please try again.\n");
}
}
}

```

OUTPUT:

The screenshot shows the OnlineGDB web interface. On the left is a sidebar with navigation links: 'Create New Project', 'My Projects', 'Classroom' (with a 'new' badge), 'Learn Programming', 'Programming Questions', 'Upgrade', and 'Logout'. The main area displays the execution of a C program named 'main.c'. The program's output is shown in a black terminal window with white text. The output reads: 'Single-Level Directory Management System', followed by a numbered list: '1. Add File', '2. Search File', '3. Delete File', '4. List Files', '5. Exit'. Below the list is the prompt 'Enter your choice:' followed by a cursor. The top of the interface has a toolbar with buttons for 'Run', 'Debug', 'Stop', 'Share', 'Save', 'Beautify', and a user profile icon.