

**Name:** REVI THIMMA REDDY

**Reg-No:** 192325025

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

**Aim:**

To implement file organization using a single-level directory in C, where all files reside in a single directory and are managed efficiently.

**Algorithm:**

1. Start the program.
2. Initialize an array to store file names and a counter for the number of files.
3. Present a menu to the user with options:
  - Create a new file.
  - Delete an existing file.
  - Search for a file.
  - Display all files.
  - Exit.
4. Based on the user's choice, perform the respective operation:
  - **Create:** Check for duplicates, then add a file if the name is unique.
  - **Delete:** Search for the file and remove it from the list.
  - **Search:** Check if the file exists in the list.
  - **Display:** Print all file names.
5. Repeat until the user exits.
6. End the program.

**Procedure:**

1. Define an array to hold file names.
2. Use loops and conditional statements to manage the files.
3. Perform operations based on user input, updating the array of file names accordingly.

**Code:**

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

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

```
#define MAX_FILES 100
```

```
#define MAX_NAME_LEN 50
```

```

int main() {

    char files[MAX_FILES][MAX_NAME_LEN];

    int file_count = 0;

    int choice;

    char filename[MAX_NAME_LEN];

    int i, found;

    do {

        printf("\n1. Create File\n2. Delete File\n3. Search File\n4. Display Files\n5. Exit\nEnter
your choice: ");

        scanf("%d", &choice);

        switch (choice) {

            case 1:

                if (file_count == MAX_FILES) {

                    printf("Directory is full.\n");

                    break;

                }

                printf("Enter file name to create: ");

                scanf("%s", filename);

                found = 0;

                for (i = 0; i < file_count; i++) {

                    if (strcmp(files[i], filename) == 0) {

```

```
        found = 1;

        break;

    }

}

if (found) {

    printf("File already exists.\n");

} else {

    strcpy(files[file_count++], filename);

    printf("File created successfully.\n");

}

break;
```

case 2:

```
printf("Enter file name to delete: ");

scanf("%s", filename);

found = 0;

for (i = 0; i < file_count; i++) {

    if (strcmp(files[i], filename) == 0) {

        found = 1;

        for (int j = i; j < file_count - 1; j++) {

            strcpy(files[j], files[j + 1]);

        }

        file_count--;

    }

}
```

```
        printf("File deleted successfully.\n");

        break;

    }

}

if (!found) {

    printf("File not found.\n");

}

break;
```

case 3:

```
printf("Enter file name to search: ");

scanf("%s", filename);

found = 0;

for (i = 0; i < file_count; i++) {

    if (strcmp(files[i], filename) == 0) {

        found = 1;

        printf("File found.\n");

        break;

    }

}

if (!found) {

    printf("File not found.\n");

}
```

```
break;
```

```
case 4:
```

```
if (file_count == 0) {  
    printf("No files in the directory.\n");  
} else {  
    printf("Files in directory:\n");  
    for (i = 0; i < file_count; i++) {  
        printf("%s\n", files[i]);  
    }  
}  
  
break;
```

```
case 5:
```

```
printf("Exiting...\n");  
  
break;
```

```
default:
```

```
    printf("Invalid choice.\n");  
}  
  
} while (choice != 5);
```

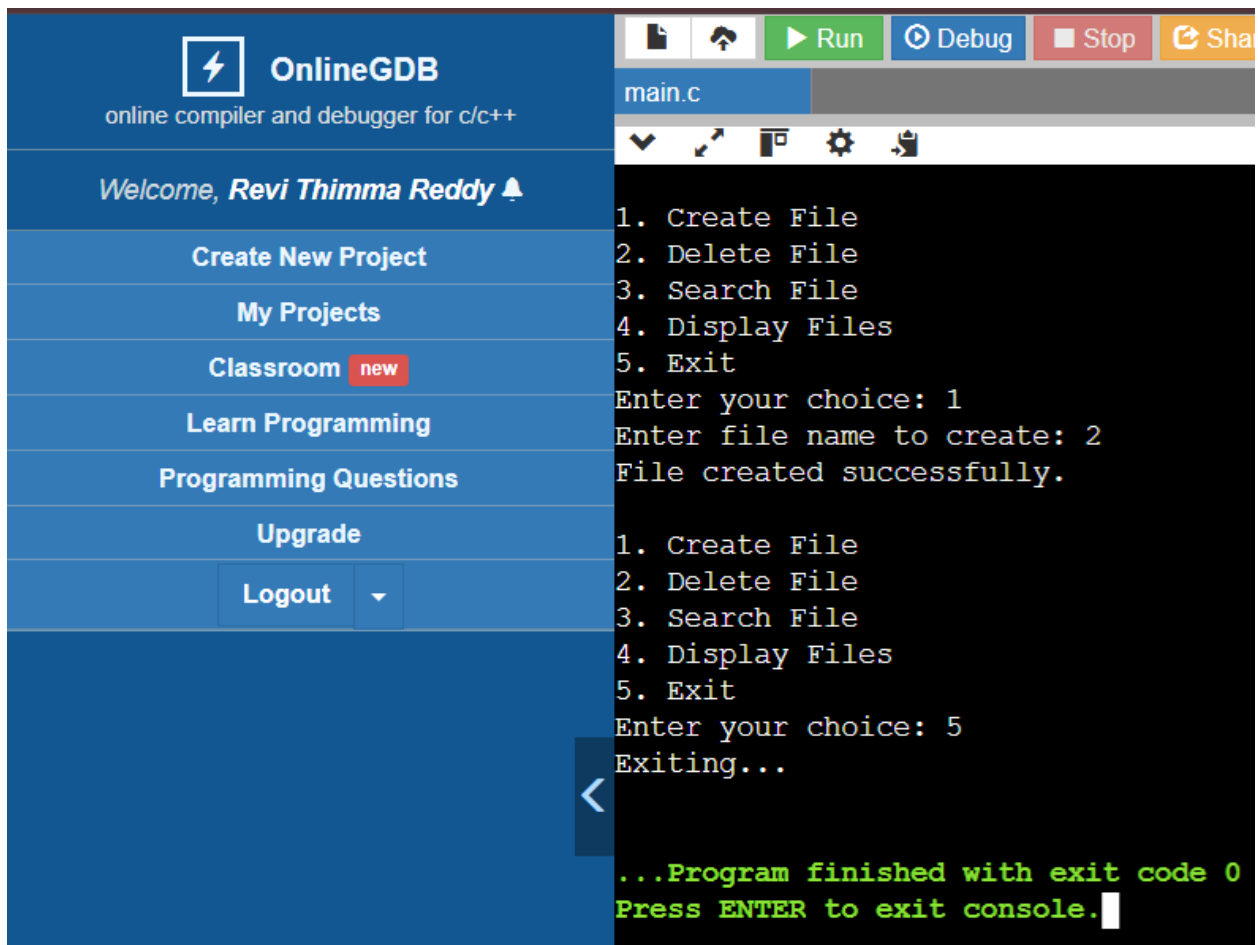
```
return 0;
```

}

### Result:

The program successfully implements a single-level directory. It allows creating, deleting, searching, and displaying files in a directory, ensuring no duplicate file names exist.

### Output:



The screenshot displays the OnlineGDB web interface. On the left is a navigation sidebar with links: 'Create New Project', 'My Projects', 'Classroom' (with a 'new' badge), 'Learn Programming', 'Programming Questions', 'Upgrade', and a 'Logout' button. The main area shows a C program named 'main.c' with the following code:

```
1. Create File
2. Delete File
3. Search File
4. Display Files
5. Exit
Enter your choice: 1
Enter file name to create: 2
File created successfully.

1. Create File
2. Delete File
3. Search File
4. Display Files
5. Exit
Enter your choice: 5
Exiting...

...Program finished with exit code 0
Press ENTER to exit console.
```

The interface includes a top toolbar with icons for file operations and buttons for 'Run', 'Debug', 'Stop', and 'Share'. The output is shown in a terminal window with a dark background and green text for the final status message.