```
Script started on 2019-03-13 13:22:00+0530
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ cat
SingleLevel.c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef struct files {
    char fname[10];
}File;
File* root[50];
int fcount = 0;
File* new_file(char s[])
{
    File* n = (File*)malloc(sizeof(File));
    strcpy(n->fname, s);
    return n;
}
int search_file(char s[])
{
    int flag = 0;
    for(int i = 0; i < fcount; i++)
    {
        if(root[i] != NULL) {
            if(strcmp(root[i]->fname,s) == 0) {
                flag = 1;
                break;
            }
        }
    }
    return flag;
}
void insert_file(char s[])
{
    if(search_file(s) == 1) {
        printf("File %s already exists!\n", s);
        return;
    }
    root[fcount] = new_file(s);
    fcount++;
    printf("Created!\n");
}
void display(File* d[])
{
    printf("Contents of root:\n");
    if(fcount == 0) {
        printf("Empty!\n");
        return;
    }
    for(int i = 0; i < fcount; i++) {
        if(root[i] != NULL) {
            printf("%s\t", root[i]->fname);
        }
    }
    printf("\n");
}
int main()
{
    int c;
    while(1)
    {
```

```
        printf("1. New File\n");
        printf("2. Display all files\n");
        printf("3. Exit\n");
        printf("Enter choice: ");
        scanf("%d",&c);
        if(c==1)
        {
            char s[50];
            printf("Enter file name: ");
            scanf("%s",s);
            insert_file(s);
        }
        else if(c==2)
        {
            display(root);
        }
        else {
            break;
        }
    }
}
```

```
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ gcc
SingleLevel.c
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$
./a.out
1. New File
2. Display all files
3. Exit
Enter choice: 1
Enter file name: file1.txt
Created!
1. New File
2. Display all files
3. Exit
Enter choice: 1
Enter file name: file1.txt
File file1.txt already exists!
1. New File
2. Display all files
3. Exit
Enter choice: 1
Enter file name: file2.txt
Created!
1. New File
2. Display all files
3. Exit
Enter choice: 1
Enter file name: fil3# #e.txt
Created!
1. New File
2. Display all files
3. Exit
Enter choice: 1
Enter file name: file4.txt
Created!
1. New File
2. Display all files
```

```
3. Exit
Enter choice: 1
Enter file name: file5,# #.txt
Created!
1. New File
2. Display all files
3. Exit
Enter choice: 2
Contents of root:
file1.txt   file2.txt   file.txt    file4.txt   file5.txt
1. New File
2. Display all files
3. Exit
Enter choice: 3
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ cat
TwoLevel.c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef struct files {
    char fname[10];
}File;
typedef struct directory {
    char fname[10];
    int c;
    File* l[5];
}Directory;
typedef struct unit {
    int d;
    void *p;
}Unit;
Unit root[50];
int count = 0;
File* new_file(char s[])
{
    File* n = (File*)malloc(sizeof(File));
    strcpy(n->fname, s);
    return n;
}
Directory* new_dir(char s[])
{
    Directory* n = (Directory*)malloc(sizeof(Directory));
    strcpy(n->fname, s);
    n->c = 0;
    for(int i = 0; i < 5; i++) n->l[i] = NULL;
    return n;
}
int search_file(char s[])
{
    int flag = 0;
    for(int i = 0; i < count; i++)
    {
        if(root[i].p != NULL) {
            if(strcmp(((File*)(root[i].p))->fname,s) == 0 && root[i].d == 0) {
                flag = 1;
                break;
            }
        }
    }
    return flag;
```

```
}
Directory* search_dir(char s[])
{
    Directory* flag = NULL;
    for(int i = 0; i < count; i++)
    {
        if(root[i].p != NULL) {
            if(strcmp(((Directory*)(root[i].p))->fname,s) == 0 && root[i].d ==
1) {
                flag = ((Directory*)(root[i].p));
                break;
            }
        }
    }
    return flag;
}
void insert_file(char s[])
{
    if(count >= 50) {
        printf("Full!\n");
        return;
    }
    if(search_file(s) == 1) {
        printf("File %s already exists!\n", s);
        return;
    }
    if(search_dir(s) != NULL) {
        printf("Directory named %s already exists!\n", s);
        return;
    }
    root[count].p = new_file(s);
    root[count].d = 0;
    count++;
    printf("Created!\n");
}
void insert_file_dir(Directory* d, char s[])
{
    int i, pos;
    if(d->c >= 5) {
        printf("Directory full!\n");
        return;
    }
    for(i = 0; i < 5; i++)
    {
        if(d->l[i] != NULL) {
            if(strcmp(d->l[i]->fname, s)==0) {
                printf("File already exists!\n");
                return;
            }
        }
        else {
            pos = i;
            i = 5;
        }
    }
    d->l[pos] = new_file(s);
    d->c = d->c + 1;
    printf("Created!\n");
}
void insert_dir(char s[])
{
    if(count >= 50) {
```

```
        printf("Full!\n");
        return;
    }
    if(search_dir(s) != NULL) {
        printf("Directory %s already exists!\n", s);
        return;
    }
    if(search_file(s) == 1) {
        printf("File named %s already exists!\n", s);
        return;
    }
    root[count].p = new_dir(s);
    root[count].d = 1;
    count++;
    printf("Created!\n");
}
void display(Unit d[])
{
    printf("Contents of root:\n");
    if(count == 0) {
        printf("Empty!\n");
        return;
    }
    int ch = 0;
    printf("Files:\n");
    for(int i = 0; i < count; i++) {
        if(root[i].p != NULL) {
            if(root[i].d == 0) {
                printf("%s ",((File*)(root[i].p))->fname); ch++;
            }
        }
    }
    if(ch == 0) printf("None!");
    printf("\nDirectories:\n");
    ch = 0;
    int dc = 0;
    for(int i = 0; i < count; i++) {
        if(root[i].p != NULL) {
            if(root[i].d == 1) {
                ch++;
                printf("%s ",((Directory*)(root[i].p))->fname);
            }
        }
    }
    if(ch == 0) printf("None!");
    printf("\n");
    ch = 0;
    for(int i = 0; i < count; i++) {
        if(root[i].p != NULL) {
            if(root[i].d == 1) {
                ch++;
                printf("Contents of %s:\n",((Directory*)(root[i].p))->fname);
                dc = 0;
                for(int j = 0; j < 5; j++)
                    if(((Directory*)(root[i].p))->l[j] != NULL) {
                        printf("%s ", ((Directory*)(root[i].p))->l[j]->fname);
                        dc++;
                    }
                if(dc == 0) printf("None!");
                printf("\n");
            }
        }
```

```
    }
    printf("\n");
}
int main()
{
    int c;
    while(1)
    {
        printf("1. New File\n");
        printf("2. New Directory\n");
        printf("3. Display all files\n");
        printf("4. Exit\n");
        printf("Enter choice: ");
        scanf("%d",&c);
        if(c==1)
        {
            char d[10], s[10];
            printf("Enter root to create file in the root directory.\nEnter
root/directory to create file in the sub-directory.\nEnter directory: ");
            scanf("%s",d);
            printf("Enter file name: ");
            scanf("%s", s);
            if(strcmp(d,"root")!=0)
            {
                char* n = strtok(d, "/");
                n = strtok(NULL, "/");
                Directory* dir = search_dir(n);
                if(dir != NULL) {
                    insert_file_dir(dir, s);
                }
                else printf("No such directory!\n");
            }
            else if(strcmp(d,"root")==0) {
                insert_file(s);
            }
        }
        else if(c==2)
        {
            char d[10];
            printf("Enter directory name: ");
            scanf("%s", d);
            insert_dir(d);
        }
        else if(c==3)
        {
            display(root);
        }
        else
        {
            break;
        }
    }
}
```

```
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ gcc
TwoLevel.c
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$
```

```
./a.out
1. New File
2. New Directory
3. Display all files
4. Exit
Enter choice: 2
Enter directory name: D1
Created!
1. New File
2. New Directory
3. Display all files
4. Exit
Enter choice: 1
Enter root to create file in the root directory.
Enter root/directory to create file in the sub-directory.
Enter directory: root
Enter file name: file1.txt
Created!
1. New File
2. New Directory
3. Display all files
4. Exit
Enter choice: 1
Enter root to create file in the root directory.
Enter root/directory to create file in the sub-directory.
Enter directory: root/D1
Enter file name: file2.txt
Created!
1. New File
2. New Directory
3. Display all files
4. Exit
Enter choice: 3
Contents of root:
Files:
file1.txt
Directories:
D1
Contents of D1:
file2.txt

1. New File
2. New Directory
3. Display all files
4. Exit
Enter choice: 4
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ cat
Tree.c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
typedef struct files {
    char fname[10];
}File;
typedef struct directory {
    char dname[10];
    struct directory *d1,*d2,*d3;
    File *f1,*f2;
}Directory;
Directory *root = NULL;
```

```
void insert_directory(char s[])
{
    Directory* temp=root;
    char *t = strtok(s,"/");
    t = strtok(NULL,"/");
    while(t != NULL) {
        if(temp->d1 != NULL && strcmp(t,temp->d1->dname)==0)
        {
            temp = temp->d1;
        }
        else if(temp->d2 != NULL && strcmp(t,temp->d2->dname)==0)
        {
            temp=temp->d2;
        }
        else if(temp->d3 != NULL && strcmp(t,temp->d3->dname)==0)
        {
            temp=temp->d3;
        }
        t = strtok(NULL,"/");
    }
    if(t==NULL)
    {
        if(temp->d1==NULL||temp->d2==NULL||temp->d3==NULL)
        {
            char d[10];
            printf("Enter the directory name: ");
            scanf("%s",d);
            Directory* newdir = (Directory*)malloc(sizeof(Directory));
            newdir->d1=NULL;
            newdir->d2=NULL;
            newdir->d3=NULL;
            newdir->f1=NULL;
            newdir->f2=NULL;
            strcpy(newdir->dname,d);
            if(temp->d1 == NULL)
            {
                temp->d1 = newdir;
            }
            else if(temp->d2 == NULL && strcmp(d,temp->d1->dname)!=0)
            {
                temp->d2 = newdir;
            }
            else if(strcmp(d,temp->d1->dname) != 0 && strcmp(d,temp->d2->dname)!
=0)
            {
                temp->d3 = newdir;
            }
            else if(strcmp(d,temp->d1->dname) == 0 || strcmp(d,temp->d2-
>dname)==0)
                printf("Duplicate directories not allowed!\n");
        }
        else printf("Directory limit exceeded!\n");
    }
}
void insert_file(char s[])
{
    Directory* temp=root;
    char temp1[100];
    strcpy(temp1,s);
    char *t = strtok(s,"/");
    t = strtok(NULL,"/");
    while(t != NULL) {
```

```
        if(temp->d1 != NULL && strcmp(t,temp->d1->dname)==0)
        {
            temp = temp->d1;
        }
        else if(temp->d2 != NULL && strcmp(t,temp->d2->dname)==0)
        {
            temp = temp->d2;
        }
        else if(temp->d3 != NULL && strcmp(t,temp->d3->dname)==0)
        {
            temp = temp->d3;
        }
        t = strtok(NULL,"/");
    }
    if(t == NULL)
    {
        if(temp->f1 == NULL || temp->f2 == NULL)
        {
            char d[10];
            printf("Enter the file name: ");
            scanf("%s",d);
            File *newfile = (File*)malloc(sizeof(File));
            strcpy(newfile->fname,d);
            if(temp->f1 == NULL)
            {
                temp->f1=newfile;
            }
            else if(temp->f2 == NULL)
            {
                temp->f2=newfile;
            }
        }
        else
            printf("File limit exceeded!");
    }
}
File* get_file_pointer(char s[])
{
    char *t = strtok(s,"/");
    char *g;
    Directory *temp = root;
    while(t != NULL) {
        if(temp->d1 != NULL && strcmp(t,temp->d1->dname)==0)
        {
            temp = temp->d1;
        }
        else if(temp->d2 != NULL && strcmp(t,temp->d2->dname)==0)
        {
            temp = temp->d2;
        }
        else if(temp->d3 != NULL && strcmp(t,temp->d3->dname)==0)
        {
            temp=temp->d3;
        }
        g = t;
        t = strtok(NULL,"/");
        if(t==NULL)
        {
            if(strcmp(temp->f1->fname,g)==0)
                return temp->f1;
            else if(strcmp(temp->f2->fname,g)==0)
                return temp->f2;
```

```
            else
            {
                printf("No such file!\n");
                return NULL;
            }
        }
    }
    return NULL;
}
Directory* get_directory_pointer(char s[])
{
    char *t = strtok(s,"/");
    char *g;
    Directory *temp = root;
    while(t != NULL){
        if(temp->d1 != NULL && strcmp(t,temp->d1->dname)==0)
        {
            temp = temp->d1;
        }
        else if(temp->d2 != NULL && strcmp(t,temp->d2->dname)==0)
        {
            temp = temp->d2;
        }
        else if(temp->d3 != NULL && strcmp(t,temp->d3->dname)==0)
        {
            temp = temp->d3;
        }
        g = t;
        t = strtok(NULL,"/");
        if(t == NULL)
        {
            return temp;
        }
    }
    return NULL;
}
void display_file(File* f, char s[])
{
    printf("%s\t\t%s\n",f->fname,s);
}
void display(Directory* r, char s[])
{
    if(r!=NULL)
    {
        strcat(s,r->dname);
        strcat(s,"/");
        if(r->f1 != NULL)
        {
            display_file(r->f1,s);
        }
        if(r->f2!=NULL)
        {
            display_file(r->f2,s);
        }
        if(r->d1 != NULL) {
            char s1[50];
            strcpy(s1, s);
            display(r->d1,s1);
        }
        if(r->d2 != NULL) {
            char s1[50];
            strcpy(s1, s);
```

```
            display(r->d2,s1);
        }
        if(r->d3 != NULL) {
            char s1[50];
            strcpy(s1, s);
            display(r->d3,s1);
        }
    }
}
int main()
{
    root = (Directory*)malloc(sizeof(Directory));
    strcpy(root->dname,"root");
    root->d1=NULL;
    root->d2=NULL;
    root->d3=NULL;
    root->f1=NULL;
    root->f2=NULL;
    int c;
    while(1)
    {
        printf("1. Insert a Directory\n");
        printf("2. Insert a File\n");
        printf("3. Display all files\n");
        printf("4. Exit\n");
        printf("Enter choice: ");
        scanf("%d",&c);
        if(c==1)
        {
            char s[50];
            printf("Path format:\n");
            printf("root/ (or) root - to insert in root\n");
            printf("root/directory/ - to insert into directory in root\n");
            printf("Enter the path: ");
            scanf("%s",s);
            insert_directory(s);
        }
        else if(c==2)
        {
            char s[50];
            printf("Path format:\n");
            printf("root/ (or) root - to insert file in root\n");
            printf("root/directory/ - to insert file into directory in root\n");
            printf("Enter the path: ");
            scanf("%s",s);
            insert_file(s);
        }
        else if(c==3)
        {
            char s[400];
            strcpy(s,"");
            printf("File\t\tPath\n");
            display(root,s);
        }
        else {
            break;
        }
    }
}
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ gcc
```

```
Tree.c
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$
./a.out
1. Insert a Directory
2. Insert a File
3. Display all files
4. Exit
Enter choice: 1
Path format:
root/ (or) root - to insert in root
root/directory/ - to insert into directory in root
Enter the path: root
Enter the directory name: D1
1. Insert a Directory
2. Insert a File
3. Display all files
4. Exit
Enter choice: 1
Path format:
root/ (or) root - to insert in root
root/directory/ - to insert into directory in root
Enter the path: root
Enter the directory name: D2
1. Insert a Directory
2. Insert a File
3. Display all files
4. Exit
Enter choice: 1
Path format:
root/ (or) root - to insert in root
root/directory/ - to insert into directory in root
Enter the path: root
Enter the directory name: D3
1. Insert a Directory
2. Insert a File
3. Display all files
4. Exit
Enter choice: 1
Path format:
root/ (or) root - to insert in root
root/directory/ - to insert into directory in root
Enter the path: root
Directory limit exceeded!
1. Insert a Directory
2. Insert a File
3. Display all files
4. Exit
Enter choice: 1
Path format:
root/ (or) root - to insert in root
root/directory/ - to insert into directory in root
Enter the path: root/D1/
Enter the directory name: file2.txt
1. Insert a Directory
2. Insert a File
3. Display all files
4. Exit
Enter choice: 3
File        Path
1. Insert a Directory
```

```
2. Insert a File
3. Display all files
4. Exit
Enter choice: 4
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ cat
DAG.c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>
struct Files {
  char fname[10];
  int reference;
};
typedef struct Files file;
struct Directory {
  char dname[10];
  struct Directory *d1,*d2,*d3;
  file *f1,*f2;
};
typedef struct Directory directory;

directory *root=NULL;

void insert_directory(char s[])
{
  directory* temp=root;
  char *t=strtok(s,"/");
  t=strtok(NULL,"/");
  while(t!=NULL){
    if(temp->d1!=NULL&&strcmp(t,temp->d1->dname)==0)
      {
      temp=temp->d1;
      }
    else if(temp->d2!=NULL&&strcmp(t,temp->d2->dname)==0)
      {
      temp=temp->d2;
      }
    else if(temp->d3!=NULL&&strcmp(t,temp->d3->dname)==0)
      {
      temp=temp->d3;
      }
    t= strtok(NULL,"/");
  }
    if(t==NULL)
      {
      if(temp->d1==NULL||temp->d2==NULL||temp->d3==NULL)
        {
          char d[10];
          printf("Enter the directory name: ");
          scanf("%s",d);
          directory* newdir = (directory*)malloc(sizeof(directory));
          strcpy(newdir->dname,d);
          newdir->d1=NULL;
            newdir->d2=NULL;
            newdir->d3=NULL;
            newdir->f1=NULL;
            newdir->f2=NULL;
          if(temp->d1==NULL)
            {
            temp->d1=newdir;
```

```
            }
          else if(temp->d2==NULL&& strcmp(d,temp->d1->dname)!=0)
            {
            temp->d2=newdir;
            }
          else if( strcmp(d,temp->d1->dname)!=0&& strcmp(d,temp->d2->dname)!=0)
            {
            temp->d3=newdir;
            }
          else if( strcmp(d,temp->d1->dname)==0|| strcmp(d,temp->d2->dname)==0)
            printf("Duplicate directories not allowed: Invalid Entry\n");

          }
      else
        printf("Directory limit Exceeded\n");
      }// close of if(t==NULL)

}//close of function

void insert_file(char s[])
{
  directory* temp=root;
  char temp1[100];
  strcpy(temp1,s);
  char *t=strtok(s,"/");
  t=strtok(NULL,"/");
  while(t!=NULL){
    if(temp->d1!=NULL&&strcmp(t,temp->d1->dname)==0)
      {
      temp=temp->d1;
      }
    else if(temp->d2!=NULL&&strcmp(t,temp->d2->dname)==0)
      {
      temp=temp->d2;
      }
    else if(temp->d3!=NULL&&strcmp(t,temp->d3->dname)==0)
      {
      temp=temp->d3;
      }
    t= strtok(NULL,"/");
  }
    if(t==NULL)
      {
      if(temp->f1==NULL||temp->f2==NULL)
        {
          char d[10];
          printf("Enter the file name: ");
          scanf("%s",d);
          file *newfile = (file*)malloc(sizeof(file));
          strcpy(newfile->fname,d);
            printf("Enter the file reference number: ");
            scanf("%d",&(newfile->reference));
          if(temp->f1==NULL)
            {
            temp->f1=newfile;
            }
          else if(temp->f2==NULL)
            {
            temp->f2=newfile;
            }
        }
      else
```

```
        printf("Directory file limit Exceeded");
      }// close of if(t==NULL)

}//close of function

file* get_file_pointer(char s[])
{
  char *t= strtok(s,"/");
  char *g;
  directory *temp=root;
  while(t!=NULL){
    if(temp->d1!=NULL&&strcmp(t,temp->d1->dname)==0)          //
      {                                                      //
      temp=temp->d1;                                        //
      }                                                      //
    else if(temp->d2!=NULL&&strcmp(t,temp->d2->dname)==0)     //    if "file"
name part of the string is reached then all the if an else id would fail
      {                                                      //
      temp=temp->d2;                                        //
      }                                                      //
    else if(temp->d3!=NULL&&strcmp(t,temp->d3->dname)==0)     //
      {
      temp=temp->d3;
      }
    g=t;//g will store the last parsed part of the string . This will the the
file name.
    t= strtok(NULL,"/");
    if(t==NULL)
      {
      if(strcmp(temp->f1->fname,g)==0)
        return temp->f1;
      else if(strcmp(temp->f2->fname,g)==0)
        return temp->f2;
      else
        {
          printf("ERROR: NO SUCH FILE FOUND\n");
          return NULL;
        }
      }//close of if(t==NULL)
  }//close of while
}// close of function.




directory* get_directory_pointer(char s[])
{
  char *t= strtok(s,"/");
  char *g;
  directory *temp=root;
  while(t!=NULL){
    if(temp->d1!=NULL&&strcmp(t,temp->d1->dname)==0)          //
      {                                                      //
      temp=temp->d1;                                        //
      }                                                      //
    else if(temp->d2!=NULL&&strcmp(t,temp->d2->dname)==0)     //
      {                                                      //
      temp=temp->d2;                                        //
      }                                                      //
    else if(temp->d3!=NULL&&strcmp(t,temp->d3->dname)==0)     //
      {
```

```
      temp=temp->d3;
      }
    g=t;
    t= strtok(NULL,"/");
    if(t==NULL)
      {
      return temp;
      }//close of if(t==NULL)
  }//close of while
}// close of function.

void create_link(char s1[],char s2[])
{
  file* f1= get_file_pointer(s1);
  char a[300];
  directory* d2=get_directory_pointer(s2);
  if(f1!=NULL){
    if(d2->f1==NULL)
      {
      d2->f1=f1;
      }
    else if(d2->f2==NULL)
      {
      d2->f2=f1;

      }
    else
      {
      printf("Not enough space in the directory to make the link.\n");
      }
  }
}
void display_file(file* f,char s[])
{
  printf("%s\t\t%s\t\t%d\n",f->fname,s,f->reference);
}

void display(directory* r,char s[])   // this is a simple n-ary tree traversal
routine
{
  if(r!=NULL)
    {
      strcat(s,r->dname);
      strcat(s,"/");
      if(r->f1!=NULL)
      {
        display_file(r->f1,s);
      }
      if(r->f2!=NULL)
      {
        display_file(r->f2,s);
      }
      display(r->d1,s);
      display(r->d2,s);
      display(r->d3,s);
    }
}

int main()
{
  root= (directory* )malloc(sizeof(directory));
  strcpy(root->dname,"root");
```

```
  root->d1=NULL;
  root->d2=NULL;
  root->d3=NULL;
  root->f1=NULL;
  root->f2=NULL;
  int c;
  while(1)
    {
      printf("1. Insert a Directory.\n");
      printf("2. Insert a File.\n");
      printf("3. Create another link to the file.\n");
      printf("4. Display all the file.\n");
      printf("5. Exit.\n");
      printf("Enter your choice: ");
      scanf("%d",&c);
      if(c==1)
      {
        printf("=============================================\n");
        printf("   Insert a Directory to the Tree.           \n");
        printf("=============================================\n");
        char s[50];
        printf("Path format:- \n");
        printf("root/ (or) root --------> to insert a directory in root\n");
        printf("root/directory1/------> to insert into the directory1 in root\
n");
        printf("Note: You need to create the directory1 before you do operation
on it.\n\n");
        printf("Enter the path: ");
        scanf("%s",s);
        insert_directory(s);
      }
      else if(c==2)
      {
        printf("=============================================\n");
        printf("   Insert a File to the Tree.           \n");
        printf("=============================================\n");
        char s[50];
        printf("Path format:- \n");
        printf("root/ (or) root --------> to insert a file in root\n");
        printf("root/directory1/------> to insert file into the directory1 in
root\n");
        printf("Note: You need to create the directory1 before you do operation
on it.\n\n");
        printf("Enter the path: ");
        scanf("%s",s);
        insert_file(s);
      }
      else if(c==3)
      {
        printf("=============================================\n");
        printf("  Create another link to the File              \n");
        printf("=============================================\n");
        char s1[50];
        char s2[50];
        printf("Enter the complete path of the file(including the file name):
");
        scanf("%s",s1);
        printf("Enter the path of the directory with which you want to create
link: ");
        scanf("%s",s2);
        create_link(s1,s2);
```

```
      }
      else if(c==4)
      {
        char s[400];
        strcpy(s,"/");
        printf("FILE\t\tPATH\n");
        display(root,s);
      }
      else{
      break;
      }
    }

}
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$ gcc
DAG.c
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$
./a.out
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 1
================================================
   Insert a Directory to the Tree.
================================================
Path format:-
root/ (or) root --------> to insert a directory in root
root/directory1/------> to insert into the directory1 in root
Note: You need to create the directory1 before you do operation on it.

Enter the path: root
Enter the directory name: D1
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 1
================================================
   Insert a Directory to the Tree.
================================================
Path format:-
root/ (or) root --------> to insert a directory in root
root/directory1/------> to insert into the directory1 in root
Note: You need to create the directory1 before you do operation on it.

Enter the path: root/D1/
Enter the directory name: D@# #2
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 2
================================================
   Insert a File to the Tree.
```

```
==============================================
Path format:-
root/ (or) root --------> to insert a file in root
root/directory1/------> to insert file into the directory1 in root
Note: You need to create the directory1 before you do operation on it.

Enter the path: root
Enter the file name: file1.txt
Enter the file reference number: 1000
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 2
==============================================
   Insert a File to the Tree.
==============================================
Path format:-
root/ (or) root --------> to insert a file in root
root/directory1/------> to insert file into the directory1 in root
Note: You need to create the directory1 before you do operation on it.

Enter the path: fi# ## #root/D1/
Enter the file name: file2.txt
Enter the file reference number: 2000
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 2
==============================================
   Insert a File to the Tree.
==============================================
Path format:-
root/ (or) root --------> to insert a file in root
root/directory1/------> to insert file into the directory1 in root
Note: You need to create the directory1 before you do operation on it.

Enter the path: root/D1/
Enter the file name: file3.txt
Enter the file reference number: 3000
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 1
==============================================
   Insert a Directory to the Tree.
==============================================
Path format:-
root/ (or) root --------> to insert a directory in root
root/directory1/------> to insert into the directory1 in root
Note: You need to create the directory1 before you do operation on it.

Enter the path: root/D1
Enter the directory name: D2
Duplicate directories not allowed: Invalid Entry
1. Insert a Directory.
2. Insert a File.
```

```
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 2
==============================================
   Insert a File to the Tree.
==============================================
Path format:-
root/ (or) root --------> to insert a file in root
root/directory1/------> to insert file into the directory1 in root
Note: You need to create the directory1 before you do operation on it.

Enter the path: root/D1/D2/
Enter the file name: file4.ts# #xt
Enter the file reference number: 4000
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 4
FILE        PATH
file1.txt         /root/
file2.txt         /root/D1/
file3.txt         /root/D1/
file4.txt         /root/D1/D2/
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 3
==============================================
  Create another link to the File
==============================================
Enter the complete path of the file(including the file name): root/D1/D4#
#2/file4.txt/# #
Enter the path of the directory with which you want to create link: root
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 3
==============================================
  Create another link to the File
==============================================
Enter the complete path of the file(including the file name): root/D1/file2.txt
Enter the path of the directory with which you want to create link: root
Not enough space in the directory to make the link.
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 4
FILE        PATH
file1.txt         /root/
file4.txt         /root/
file2.txt         /root/D1/
file3.txt         /root/D1/
file4.txt         /root/D1/D2/
```

```
1. Insert a Directory.
2. Insert a File.
3. Create another link to the file.
4. Display all the file.
5. Exit.
Enter your choice: 5
#]0;praveen@praveen-Lenovo-ideapad-520-15IKB: ~/Opearting systems/15.File
Organisation techniques##[01;32mpraveen@praveen-Lenovo-ideapad-520-
15IKB#[00m:#[01;34m~/Opearting systems/15.File Organisation techniques#[00m$
exit
exit

Script done on 2019-03-13 13:36:56+0530
```