Ex. No.: 12
Date:

File Organization Technique- Single and Two level directory

AIM:

To implement File Organization Structures in C are

a. Single Level Directory
b. Two-Level Directory
c. Hierarchical Directory Structure
d. Directed Acyclic Graph Structure

a. Single Level

Directory

ALGORITHM

1. Start
2. Declare the number, names and size of the directories and file names.
3. Get the values for the declared variables.
4. Display the files that are available in the directories.
5. Stop.

PROGRAM:

```
#include<stdio.h>
#include<stdlib.h>
#include<graphics.h>
void main() {
    int gd=DETECT, gm, count, i, j, mid, ur_x;
    char fname[10][20];
    int graph(&gd, &gm, "c:\\tc\\bgi");
    clear device();

    Set br color(Green);
    puts("Enter the no: of files");
```

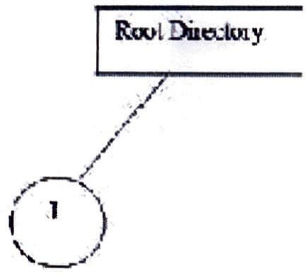
```
scanf("%d", &count);
for(i=0; i<count; i++){
    cleardevice();
    setbkcolor("Green");
    printf("Enter the file %d name", i+1);
    scanf("%s", fname[i]);
    setfillstyle(1, MAGNETA);
    mid= 640/count; ur_x= mid/3;
    bar3d(270,100,370,150,0,0);
    settextstyle(2,0,4);
    settextjustify(1,1);
    outtextxy(320,125," Root Directory");
    sdcolor(BLUE);
    for(j=0; j<=i; ur_x+=mid){
        lines(320,150,ur_x-250);
        fillellipse(ur_x,250,30,30);
        outtextxy(ur_x, 250, fname[j]);
    }
}
```

OUTPUT:

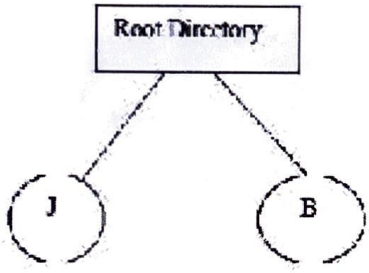
Enter the Number of files

2

Enter the file1 J



Enter the file2 B



b. Two-level directory Structure

ALGORITHM:

1. Start
2. Declare the number, names and size of the directories and subdirectories and file names.
3. Get the values for the declared variables.
4. Display the files that are available in the directories and subdirectories.
5. Stop.

PROGRAM:

```
#include <stdio.h>
#include<graphic.h>
struct tree_element {
    char name[20];
    int x, y, ftype, nx, inc, level; structure_element*
        link[5]; typedef struct tree_element node;
void main() {
    int gd= DETECT, gm; node* root
    root = NULL; clrscr();
    create (&root, 0, "null", 0, 630, 320);
    clrscr();
    initgraph (& gd, &gm, "C:\\tc\\bgi");
    display (root);
    getch();
    closegraph();
}
create (node** root, int lev, char* dname, int lx,
        int nx, int x) {
```

```
int i, gap;
if (*root == NULL){
    (*root) = (node*) malloc (sizeof (node));
    printf ("enter name of dir/ file
(under %s): ", dname); fflush(stdin);
    gets ((*root) -> name);
    if (lev == 0 || lev == 1)
    (*root) -> ftype = 1;
else
    (*root) -> ftype = 2;
    (*root) -> level = lev;
    (*root) -> y = 50 + lev * 50;
    (*root) -> x = x;
    (*root) -> lx = lx;
    (*root) -> rx = rx;
    for (i=0; i<5; i++)
    (*root) -> link[i] = NULL;
    if ((*root -> ftype == 1){
        if (lev == 0 || lev == 1)
        {
        if ((*root -> level == 0)
        printf ("How many users");
        else
```

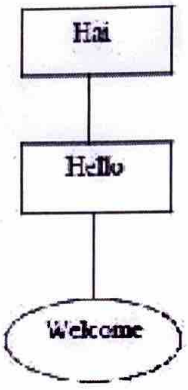
```
printf ("How many files");
printf ("(for %s): ", (*root) -> name);
scanf ("%d", &(*root) -> nc);
}
else (*root) -> nc = 0;
if ((*root) -> nc == 0)
gap = rx - lx;
else
gap = (rx - lx) / (*root) -> nc;
for (i=0; i < (*root) -> nc; i++)
create (&((*root) -> link[i]), lev+1,
(*root) -> name, lx + gap * i, lx + gap * i
+ gap, lx + gap * i + gap/2);
}
else
(*root) -> nc = 0;
}
}
```

```
display (node* root) {
int i;
settextstyle (2,0,4);
settextjustify(1,1);
setfillstyle (1, BLUE);
setcolor(14);
if (root != NULL){
for (i=0; i < root -> nc; i++)
```

```
{
    line (root->x, root->y, root->link[i]->x,
        root->link[i]->y);
}
if (root->type==1) bar3d (root->x-20,
 root->y-10, root->x+20, root->y+10, 0, 0);
else
fill ellipse (root->x, root->y, 20, 20);
outtextxy (root->x, root->y, root->name);
for(i=0; i<root->nc; i++)
{
    display (root->link[i]);
    }
}
}
```

Sample Output:

Enter the name of dir/file(under null): Hai
How many users(for Hai):1
Enter name of dir/file(under Hai):Hello
How many files(for Hello):1
Enter name of dir/file(under Hello):welcome



Result:

Thus the code implementation file structure is executed successfully.