

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 <graphics.h>
#include <stdlib.h>
#include <string.h>

struct tree-element {
    char name[20];
    int x, y, ttype, lx, rx, nc, level;
    struct tree-element *link[5];
}

typedef struct tree-element node;
void create [node **root, int lev, char *dname,
            int lx, int rx, int x);
void display (node *root);
void main() {
    int gd = DETECT gm;
    node * root = NULL;
    clrscr();
```



```
create(&root, 0, "null", 0, 630, 320);
```

```
clrscr();
```

```
init_graph(&gd, &gm, "C:\\TurboC3\\BGI");
```

```
display(root);
```

```
getch();
```

```
closegraph();
```

```
}
```

```
void create(node** root, int lev, char *dname, int lx,  
int rx, int x) {
```

```
int i, gap;
```

```
if (*root == NULL) {
```

```
*root = (node*) malloc (sizeof (node));
```

```
printf("Enter the name of dir file (under %s);",  
dname);
```

```
fflush(stdin);
```

```
gets((*root) -> name);
```

```
(*root) -> ftype = (lev != 1) ? 1 : 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) {
```



```

printf ((lev == 0) ? "How many users (for %.s) : " :
        "How many files (for %.s) : ", (*root) → name);
scanf ("%d", &((*root) → nc));

```

```

}

```

```

else {

```

```

    (*root) → nc = 0;

```

```

}

```

```

gap ((*root) → nc == 0) ? (rx - lx) : (rx - lx) /
    (*root) → nc;

```

```

for (i = 0; i < (*root) → nc; i++) {
    create (&((*root) → link[i]), la + 1, (*root) → name,

```

```

        lx + gap * i, lx + gap * i + gap,
        lx + gap * i + gap / 2); }

```

```

}

```

```

void display (node *root) {

```

```

    int i;

```

```

    setttextjustify (1, 1);

```

```

    settextstyle (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 → ftype == 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);

outtext xy (root → x, root → y, root → name);

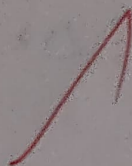
for (i=0; i < root → nc, i++) {

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~~ technique to implement file organization - single and two level directory has been executed successfully.

[Signature]