# Lab-5: FILE ALLOCATION TECHNIQUE

1. **Outcomes:** After this lab, learner able to:
   1.1. Use some most common linux commands
2. **Prepare:**
   2.1. Read most commands linux
3. **Facilities**
   3.1. Computer with Ubuntu operating system
4. **Duration:** 4 hours
5. **Sequences**
   5.1. **Programming-1: INDEXED ALLOCATION**
   - ✓ **AIM:** To write a C program to implement File Allocation concept using the technique indexed allocation Technique..
   - ✓ **ALGORITHM:**
     - o Step 1: Start the Program
     - o Step 2:Obtain the required data through char and int datatypes.
     - o Step 3:Enter the filename,index block.
     - o Step 4: Print the file name index loop.
     - o Step 5:Fill is allocated to the unused index blocks
     - o Step 6: This is allocated to the unused linked allocation.
     - o Step 7: Stop the execution.
   - ✓ **PROGRAM CODING**

```
#include<stdio.h>
void main()
{
char a[10];
int i,ib,cib[10];
printf("\n enter the file name:");
scanf("%s",a);
printf("\n index block:");
scanf("%d",&ib);
for(i=1;i<=5;i++)
{
printf("\n enter the child of index block %d:",i);
scanf("%d",&cib[i]);
}
printf("\n the list of files\t index block\n");
printf("%s\t\t %d",a,ib);
printf("\n the above file utiltization index block of child blocks followin\t");
printf("\n");
for(i=1;i<=5;i++)
{
printf("%d\t\t",cib[i]);
}
printf("\n");
}
```

- ✓ **OUTPUT:**

```
Enter the name:Testing
Index block:19
Enter the child of index block 1:9
Enter the child of index block 2:16
Enter the child of index block 3:1
Enter the child of index block 4:10
Enter the child of index block 5:25
The list of files index block
Testing 19
The above file utilization index block of child blocks following:
9   16       1         10       25
```

- ✓

5.2. **Programming-2**: **LINKED ALLOCATION**

- ✓ **AIM:**To write a C program to implement File Allocation concept using the technique Linked
  List Technique..
- ✓ **ALGORITHM:**
  - o Step 1: Start the Program
  - o Step 2:Obtain the required data through char and int datatypes.
  - o Step 3:Enter the filename,starting block ending block.
  - o Step 4: Print the free block using loop.
  - o Step 5:"for" loop is created to print the file utilization of linked type of entered type .
  - o Step 6: This is allocated to the unused linked allocation.
  - o Step 7: Stop the execution
- ✓ **PROGRAM CODING**

```
#include<stdio.h>
void main()
{
char a[10];
int i,sb,eb,fb1[10];
printf("\n enter the file name:");
scanf("%s",a);
printf("\n Enter the starting block:");
scanf("%d",&sb);
printf("Enter the ending Block:");
scanf("%d",&eb);
for(i=0;i<5;i++)
{
printf("Enter the free block %d",i+1);
scanf("%d",&fb1[i]);
}
printf("\n File name \t Starting block \t Ending block \n");
printf("%s \t\t %d\t\t %d",a,sb,eb);
printf("\n %s File Utilization of Linked type of following blocks:",a);
printf("\n %d->",sb);
for(i=0;i<5;i++)
{
printf("%d->",fb1[i]);
}
printf("%d\n",eb);
}
```

- ✓ **OUTPUT:**

```
Enter the filename:binary
Enter the starting block:19
Enter the ending block:25
Enter the free block:1:12
Enter the free block:2:34
Enter the free block:3:21
Enter the free block:4:18
Enter the free block:5:35
File name              starting block         endingblock
Binary                 19                     25
Binary file utilization of linked type of the following
blocks: 19        12      34      21      18      35      25
```

- ✓

## 5.3. Programming-3: SINGLE LEVEL DIRECTORY

- ✓ **AIM**: To write a C program to implement File Organization concept using the technique Single level directory.
- ✓ **ALGORITHM:**
  - o Step 1: Start the Program
  - o Step 2:Obtain the required data through char and int datatypes.
  - o Step 3:Enter the filename,index block.
  - o Step 4: Print the file name index loop.
  - o Step 5:Fill is allocated to the unused index blocks
  - o Step 6: This is allocated to the unused linked allocation.
  - o Step 7: Stop the execution
- ✓ **PROGRAM CODING**

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<graphics.h>
void main()
{
int gd=DETECT,gm,count,i,j,mid,cir_x;
char fname[10][20];
clrscr();
initgraph(&gd,&gm,"c:\tc\bgi");
cleardevice();
setbkcolor(GREEN);
puts("Enter no of files do u have?");
scanf("%d",&count);
for(i=0;i<count;i++)
{
cleardevice();
setbkcolor(GREEN);
printf("Enter file %d name",i+1);
scanf("%s",fname[i]);
setfillstyle(1,MAGENTA);
mid=640/count;
cir_x=mid/3;
bar3d(270,100,370,150,0,0);
settextstyle(2,0,4);
settextjustify(1,1);
outtextxy(320,125,"Root
Directory"); setcolor(BLUE);
for(j=0;j<=i;j++,cir_x+=mid)
{
line(320,150,cir_x,250);
fillellipse(cir_x,250,30,30);
outtextxy(cir_x,250,fname[j]);
}
getch();
}
}
```

✓

## 5.4. Programming-4: FILE ORGANIZATION TECHNIQUES : TWO LEVEL

- ✓ **AIM:** To write a C program to implement File Organization concept using the technique two
  level directory
- ✓ **ALGORITHM:**
  - Step 1: Start the Program
  - Step 2: Obtain the required data through char and in datatypes.
  - Step 3: Enter the filename, index block.
  - Step 4: Print the file name index loop.
  - Step 5: File is allocated to the unused index blocks
  - Step 6: This is allocated to the unused linked allocation.
  - Step 7: Stop the execution
- ✓ **PROGRAM CODING**

```
#include<stdio.h>
#include<graphics.h>
struct tree_element
{
char name[20];
int x,y,ftype,lx,rx,nc,level;
struct tree_element *link[5];
};
```

```
typedef truct tree_element
node; void main()
{
int gd=DETECT,gm;
node *root;
root=NULL;
clrscr();
create(&root,0,"null",0,639,320);
clrscr();
initgraph(&gd,&gm,"c:\tc\bgi");
display(root);
getch();
closegraph();
}
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 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("hoe 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+ga
p/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->ftype==1) bar3d(root->x-20,root->y-10,root->x+20,roo>y+10,0,0);
else
fillellipse(root->x,root->y,20,20);
outtextxy(root->x,root->y,root->name);
for(i=0;i<root->nc;i++)
{
display(root->link[i]);
}
}
}
```

✓

**5.5.**