**PROGRAM:**

#include<stdio.h>

int main()

{

int gd=DETECT,gm,count,i, j,mid, cir\_x;

char fname[10][20];

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(2,9,4);

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]);

}

}

}

**OUTPUT:**

Enter no of files do u have? 3

Enter file 1 name: 1.c

ROOT

1

Enter file 2 name: 2.c

ROOT

1

2

Enter file 3 name: 3.c

ROOT

1

3

2

**PROGRAM:**

#include<stdio.h>

#include<graphics.h>

struct tree\_element

{

char name[20];

int x, , ,ftype, lx, rx, nc, level;

struct tree\_element \*link[5];

};

Typedef struct tree\_element node;

int main()

{

Int gd=DETECT, gm;

Node \*root;

Root=NULL;

create(&root,0,”null”,0,639,3320);

clrscr():

initgraph(&gd,&gm,”c;\\tc\\bgi”);

display(root);

closegraph();

}

create(node \*\*root,int lev, char \*danme, 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=1ev;

(\*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->ftype==1)

Bar3d(root->x-20,root->y-10,root->x+20,root->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]);

}

}

}

**OUTPUT**

Enter name of dir/file(under null):sld

How many users(forsld):2

Enter name of dir/file(under sld);tld

How many files(for tld): 2

Enter name of dir/file(under tld): hir

Enter name of dir/file(under tld): dag

Enter name of dir/file(under sld): bin

How many files(for bin): 2

Enter name of dir/file(under bin): exe

Enter name of dir/file(under bin): obj

sld

bin

tld

exe

hir

obj

dag

**PROGRAM**

#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 struct tree\_element node;

int main()

{

Int gd=DETECT, gm;

Node \*root;

Root=NULL:

Create(&root,0,”root”,0,639,320);

Initgraph(&gd,&gm,”c:\\tc\\BGI”);

Display(root);

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);

Printf(“enter 1 for Dir/2 for file:”);

Scanf(“%d”,&(\*root)->ftype);

(\*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(“No of sub directories/files(for %s);”,(\*root)->name);

Scanf(“%d”,&(\*root)->nc;

If((\*root)->nc==0)

Gap=rx-lx;

Else

Gap=(rx-ls)/(\*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->ftype==1)

Bar3d(root->x-20,root->y-10,root->x+20,root->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]);

}

}

}

**OUTPUT**

Enter name of dir/file(under root): root

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for root) : 2

Enter name of dir/file(under root) : user 1

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for user 1) : 1

Enter name of dir/file(under user 1) : subdir

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for subdir) : 2

Enter name of dir/file(under user 1) : java

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for java) : 0

Enter name of dir/file(under subdir) : vb

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for vb) : 0

Enter name of dir/file(under root) : user2

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for user2) : 2

Enter name of dir/file(under root) : a

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under user2) : subdir2

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for subdir2) : 2

Enter name of dir/file(under subdir2) : ppl

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for ppl) : 2

Enter name of dir/file(under ppl) : b

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under ppl) : c

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under subdir) :ai

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for ai) : 2

Enter name of dir/file(under ai) :d

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under ai) :e

Enter 1 for Dir/ 2 for file : 2

root

User2

User1

a

Subdir2

Subdir1

ai

ppl

vb

java

d

c

b

e

**PROGRAM**

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

#include<string.h>

Struct tree\_element

{

Char name[20];

Int x, y, ftype, ls, rs, nc, level;

Struct tree\_element \*link[5];

};

Typedef struct tree\_element node;

Typedef struct

{

Char from[20],to[20];

}link;

Link l[10];

Int nofl;

Node \*root;

Void main()

{

Int gd=DETECT, gm;

Root=NULL;

Clrscr();

Create(&root,0,”root”,0,639,320);

Read\_links();

Clrscr();

Initgraph(&gd,&gm,”c:\\tc\\BGI”);

Draw\_link\_lines();

Display(root);

Getch();

Closegraph();

}

Read\_links()

{

Int I;

Printf(“how many links”);

Scanf(“%d”, &nofl;

For(i=0;i<nofl;i++)

{

Printf(“File/dir : “);

Fflush(stdin);

Gets(l[i].from);

Printf(“user name : “);

Fflush(stdin);

Gets(l[i].to);

}

}

Draw\_link\_lines()

{

Int I, x1, y1, x2, y2;

For(i=0;i<nofl;i++)

{

Search(root,l[i].from,&x1,&y1);

Search(root,l[i].to,&x2,&y2);

Setcolor(LIGHTGREEN);

Setlinestyle(3,0,1);

Line(x1,y1.x2.y2);

Setcolor(YELLOW);

Setlinestyle(0,0,1);

}

}

Search(noe \*root,char \*s,int \*x,int \*y)

{

Int I;

If(root!=NULL)

{

If(strcmpi(root->name,s)==0)

{

\*x=root->x;

\*y=root->y;

Return;

}

Else

{

For(i=0;i<root->nc;i++)

Search(root->link[i],s,x,y);

}

}

}

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);

Printf(“enter 1 for Dir/2 for file:”);

Scanf(“%d”,&(\*root)->ftype);

(\*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(“No of sub directories/files(for %s);”,(\*root)->name);

Scanf(“%d”,&(\*root)->nc;

If((\*root)->nc==0)

Gap=rx-lx;

Else

Gap=(rx-ls)/(\*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->ftype==1)

Bar3d(root->x-20,root->y-10,root->x+20,root->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]);

}

} }

**OUTPUT**

Enter name of dir/file(under root): root

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for root) : 2

Enter name of dir/file(under root) : user 1

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for user 1) : 2

Enter name of dir/file(under user 1) : vb

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for vb) : 2

Enter name of dir/file(under vb) : a

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under vb) : b

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under user1) : c

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under root) : user2

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for user2) : 1

Enter name of dir/file(under user2) : java

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for java) : 2

Enter name of dir/file(under java) : d

Enter 1 for Dir/ 2 for file : 2

Enter name of dir/file(under java) : html

Enter 1 for Dir/ 2 for file : 1

No of subdirectories /files (for html) : 0

How many links :2

File/dir : B

User name:user2

File/dir:html

User name:user1

**PROGRAM**

#include<stdio.h>  
int max[100][100];  
int alloc[100][100];  
int need[100][100];  
int avail[100];  
int n,r;  
void input();  
void show();  
void cal();  
int main()  
{  
int i,j;  
printf("\*\*\*\*\*\*\*\*\*\* Banker's Algo \*\*\*\*\*\*\*\*\*\*\*\*\n");  
input();  
show();  
cal();  
return 0;  
}  
void input()  
{  
int i,j;  
printf("Enter the no of Processes\t");  
scanf("%d",&n);  
printf("Enter the no of resources instances\t");  
scanf("%d",&r);  
printf("Enter the Max Matrix\n");  
for(i=0;i<n;i++)  
{  
for(j=0;j<r;j++)  
{  
scanf("%d",&max[i][j]);  
}  
}  
printf("Enter the Allocation Matrix\n");  
for(i=0;i<n;i++)  
{  
for(j=0;j<r;j++)  
{  
scanf("%d",&alloc[i][j]);  
}  
}  
printf("Enter the available Resources\n");  
for(j=0;j<r;j++)  
{  
scanf("%d",&avail[j]);  
}  
}  
void show()  
{  
int i,j;  
printf("Process\t Allocation\t Max\t Available\t");  
for(i=0;i<n;i++)  
{  
printf("\nP%d\t   ",i+1);  
for(j=0;j<r;j++)  
{  
printf("%d ",alloc[i][j]);  
}  
printf("\t");  
for(j=0;j<r;j++)  
{  
printf("%d ",max[i][j]);  
}  
printf("\t");  
if(i==0)  
{  
for(j=0;j<r;j++)  
printf("%d ",avail[j]);  
}  
}  
}  
void cal()  
{  
int finish[100],temp,need[100][100],flag=1,k,c1=0;  
int safe[100];  
int i,j;  
for(i=0;i<n;i++)  
{  
finish[i]=0;  
}  
//find need matrix  
for(i=0;i<n;i++)  
{  
for(j=0;j<r;j++)  
{  
need[i][j]=max[i][j]-alloc[i][j];  
}  
}  
printf("\n");  
while(flag)  
{  
flag=0;  
   for(i=0;i<n;i++)  
{  
int c=0;  
for(j=0;j<r;j++)  
{  
if((finish[i]==0)&&(need[i][j]<=avail[j]))  
{  
     c++;  
if(c==r)  
{  
for(k=0;k<r;k++)  
{  
avail[k]+=alloc[i][j];  
finish[i]=1;  
flag=1;  
       }  
   printf("P%d->",i);  
   if(finish[i]==1)  
{  
   i=n;  
   }  
       }  
}  
}  
   }  
 }  
 for(i=0;i<n;i++)  
 {  
if(finish[i]==1)  
{  
c1++;  
}  
else  
{  
printf("P%d->",i);  
}  
}  
if(c1==n)  
{  
printf("\n The system is in safe state");  
}  
else  
{  
printf("\n Process are in dead lock");  
printf("\n System is in unsafe state");  
}  
}

**OUTPUT**

\*\*\*\*\*\*\*\*\*\* Banker's Algo \*\*\*\*\*\*\*\*\*\*\*\*

Enter the no of Proseesses 5

Enter the no of resources instances 3

Enter the max matric

7 5 3

3 2 2

9 0 2

2 2 2

4 3 3

Enter the allocation matrix

1. 1 0
2. 0 0
3. 0 2
4. 1 1
5. 0 2

Enter the available resources

1. 3 2

Process Allocation Max Available

P1 0 1 0 7 5 3 3 3 2

P2 2 0 0 3 2 2

P3 3 0 2 9 0 2

P4 2 1 1 2 2 2

P5 0 0 2 4 3 3

P1->P3->P4->P2->P0

The system is in safe state