```
/*
*********************************************************
Author:Pkhighfile
*********************************************************

PROGRAM IN C++ TO CREATES A 3-DIMENSIONAL SOLID OBJECT,
USING ROTATIONAL SWEEP  REPRESENTATION METHOD
OR TRANSLATIONAL SWEEP  REPRESENTATION METHOD

*/

#include"3dframe.cpp"
#include<iostream.h>
void circleMidpoint(int ,int ,int,int );
void circlePlotpoint(int ,int ,int ,int,int ) ;

void main()
{
    int gd=DETECT,gm;
    initgraph(&gd,&gm,"c:\\tc\\bgi");
    int x_center=0,y_center=0,z_center=0,radius=700;
    DRAW3DFRAME();
    cleardevice();
    outtextxy(0,0,"Circle before Sweep");
    circleMidpoint(x_center,y_center,z_center,radius);
    getch();
    outtextxy(0,50,"Now look at the Translational Sweep");
    getch();
    for(int i=0;i<=1000;i=i+1)
    {
       translate_point(x_center,y_center,z_center,0,0,1);
       x_center=P1[0][0];
       y_center=P1[0][1];
       z_center=P1[0][2];
       circleMidpoint(P1[0][0],P1[0][1],P1[0][2],radius);
    }
    getch();
    cleardevice();
    moveto(0,0);
    outtextxy(0,0,"Now look at the Rotational Sweep");
    getch();
    cleardevice();
    radius=200;
    for(float  j=0;j<=360;j=j+1)
    {
       rotate_point_X(x_center,y_center,z_center,j);
       x_center=P1[0][0];
       y_center=P1[0][1];
       z_center=P1[0][2];
       circleMidpoint(P1[0][0],P1[0][1],P1[0][2],radius);
    }
    getch();
    cleardevice();
    outtextxy(0,0,"Now look at another the Rotational Sweep");
    getch();
    cleardevice();
    radius=1500;
    for(  i=0;i<=300;i=i+1)
    {
       rotate_point_X(-x_center,-y_center,-z_center,-1);
       x_center=P1[0][0];
       y_center=P1[0][1];
       z_center=P1[0][2];
       circleMidpoint(P1[0][0],P1[0][1],P1[0][2],radius);
       rotate_point_Y(-x_center,-y_center,-z_center,-1);
       x_center=P1[0][0];
       y_center=P1[0][1];
       z_center=P1[0][2];
       circleMidpoint(P1[0][0],P1[0][1],P1[0][2],radius);
       rotate_point_Z(-x_center,-y_center,-z_center,-1);
       x_center=P1[0][0];
       y_center=P1[0][1];
       z_center=P1[0][2];
       circleMidpoint(P1[0][0],P1[0][1],P1[0][2],radius);
    }
    getch();
    cleardevice();
    getch();
    closegraph();
}

void circleMidpoint(int xcenter,int ycenter,int zcenter,int radius)
{
     int x=0;
     int y=radius;
     int p=1-radius;
     //plot for first set of points
     circlePlotpoint(xcenter,ycenter,zcenter,x,y);
     while(x<y)
     {
     x=x+1;
     if(p<0)
          p += 2*x + 1;
     else
     {
          y=y-1;
          p +=2*(x-y)+1;
     }
     circlePlotpoint(xcenter,ycenter,zcenter,x,y);
     }
}

void circlePlotpoint(int xcenter,int ycenter,int zcenter,int x,int y)
{
     int arr[4];
     putxyz(xcenter+x,ycenter+y,zcenter,arr,RED);
     putxyz(xcenter-x,ycenter+y,zcenter,arr,LIGHTRED);
     putxyz(xcenter+x,ycenter-y,zcenter,arr,MAGENTA);
     putxyz(xcenter-x,ycenter-y,zcenter,arr,BROWN);
     putxyz(xcenter+y,ycenter+x,zcenter,arr,BLUE);
     putxyz(xcenter-y,ycenter+x,zcenter,arr,YELLOW);
     putxyz(xcenter+y,ycenter-x,zcenter,arr,GREEN);
     putxyz(xcenter-y,ycenter-x,zcenter,arr,LIGHTGREEN);
}
```

Analog Clock Program

```
#include<graphics.h>

#include<conio.h>

#include<math.h>

#include<dos.h>

void main()

{

int gd=DETECT,gm;

int x=320,y=240,r=200,i,h,m,s,thetamin,thetasec;

struct  time t;

char n[12][3]={"3","2","1","12","11","10","9","8","7","6","5","4"};

initgraph(&gd,&gm,"f:\arun\tc");\put the directory which contains

egavga.bgi

circle(x,y,210);

setcolor(4);

settextstyle(4,0,5);

for(i=0;i<12;i++)

{

if(i!=3)

outtextxy(x+(r-14)*cos(M_PI/6*i)-10,y-(r-14)*sin(M_PI/6*i)-26,n[i]);

else

outtextxy(x+(r-14)*cos(M_PI/6*i)-20,y-(r-14)*sin(M_PI/6*i)-26,n[i]);

}

gettime(&t);

printf("The current time is: %2d:%02d:%02d.%02d

",t.ti_hour, t.ti_min,
t.ti_sec, t.ti_hund);
while(!kbhit())
{
setcolor(5);
setfillstyle(1,5);
circle(x,y,10);
floodfill(x,y,5);
gettime(&t);
if(t.ti_min!=m)
{
setcolor(0);
line(x,y,x+(r-60)*cos(thetamin*(M_PI/180)),y-(r-60)*sin(thetamin*(M_PI/180
)));
circle(x+(r-80)*cos(thetamin*(M_PI/180)),y-(r-80)*sin(thetamin*(M_PI/180))
,10);
line(x,y,x+(r-110)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-110)*sin(M_PI/6*h
-((m/2)*(M_PI/180))));
circle(x+(r-130)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-130)*sin(M_PI/6*h-(
(m/2)*(M_PI/180))),10);
}
if(t.ti_hour>12)
t.ti_hour=t.ti_hour-12;
if(t.ti_hour<4)
h=abs(t.ti_hour-3);
else
h=15-t.ti_hour;
m=t.ti_min;
```

```
if(t.ti_min<=15)

thetamin=(15-t.ti_min)*6;

else

thetamin=450-t.ti_min*6;

if(t.ti_sec<=15)

thetasec=(15-t.ti_sec)*6;

else

thetasec=450-t.ti_sec*6;

setcolor(4);

line(x,y,x+(r-110)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-110)*sin(M_PI/6*h

-((m/2)*(M_PI/180))));

circle(x+(r-130)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-130)*sin(M_PI/6*h-(

(m/2)*(M_PI/180))),10);

line(x,y,x+(r-60)*cos(thetamin*(M_PI/180)),y-(r-60)*sin(thetamin*(M_PI/180

)));

circle(x+(r-80)*cos(thetamin*(M_PI/180)),y-(r-80)*sin(thetamin*(M_PI/180))

,10);

setcolor(15);

line(x,y,x+(r-70)*cos(thetasec*(M_PI/180)),y-(r-70)*sin(thetasec*(M_PI/180

)));

delay(1000);

setcolor(0);

line(x,y,x+(r-70)*cos(thetasec*(M_PI/180)),y-(r-70)*sin(thetasec*(M_PI/180

)));

}

}
```

Game Pack in C++.

```
#include<string.h>

#include<math.h>

#include<graphics.h>

#include<dos.h>

#include<stdlib.h>

#include<stdio.h>

#include<conio.h>

#include<iostream.h>

#include<process.h>

#include<time.h>

void main();

int temp_life;

int i=8,j=18,a[20][20],ri,rj,max=2,speed=2,op_graph=0;

int score=0,nbox=5,life=5,level=1;


clock_t start, end;

void draw(int);

void drawbox(int);

void decr();

void genbox();

void shotbox(int);

void strt();

void putscore(int);

void lifebox(int );
```

```
int global=9;int open_times=0,prev_card=0,comp=0;


void card(int left,int top,int right,int bottom)

{

int x,y;

long int er=0;

if(global==9)

er=2000;

else

er=200;

for (long double g=0;g<er;g++)

{

                x=random(right);

                y=random(bottom);

                if(x<left)

                {

                g--;

                continue;

                }


                if(y<top)

                {

                g--;

                continue;

                }


                int colors=random(15);
                if(colors==1||colors==10)
                {
                g--;
                continue;
                }
                setcolor(BLACK);
                setfillstyle(SOLID_FILL,colors);
                bar3d(x-4,y-4,x+4,y+4,1,1);



                }
}
void open(int left[],int top[],int right[],int bottom[],int game[],int z)
{
sound(900);
delay(16);
nosound();


int x;
global=909;

                for(int j=left[z],k=top[z],l=right[z],i=bottom[z];j<=right[z];j++)
                {
```

```
setfillstyle(SOLID_FILL,BLACK);

bar(j+1,k,l+1,i+1);

setcolor(WHITE);

rectangle(j+1,k,l+1,i+1);

sound(j*20);

delay(5);

nosound();

}



int number=game[z];

char string[5];

setcolor(BLACK);

settextstyle(TRIPLEX_FONT,HORIZ_DIR,1);

outtextxy(left[z]+25,top[z]+35,itoa(number, string, 10));


if(open_times==0)

prev_card=z;


if(open_times==1)

if(game[z]==game[prev_card])

game[prev_card]=game[z]=-100;


open_times++;


if(open_times==2)

for(int k=0,t=19,b=99;k<4;k++,t+=120,b+=121)     //displaying cards
```

```
                         for(int i=0,l=55,r=125;i<4;i++,l+=150,r+=150)

                         {

                         sound(1000);

                         delay(60);

                         nosound();


                         if(k==0 && i==0) delay(90);

                         card(l,t,r,b);

                         open_times=0;

                         }


}


void menu(int a,int b,int c,int d,int e)

{

 setcolor(a);

 settextstyle(TRIPLEX_FONT,HORIZ_DIR,7);

 outtextxy(22,100,"1. BRICKS");


 setcolor(b);

 settextstyle(TRIPLEX_FONT,HORIZ_DIR,7);

 outtextxy(22,200,"2. PAIRS II");


 setcolor(c);

 settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);

 outtextxy(15,300,"3. HELP");
 setcolor(d);

 settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);

 outtextxy(15,350,"4. CREDITS");


 setcolor(e);

 settextstyle(TRIPLEX_FONT,HORIZ_DIR,4);

 outtextxy(15,400,"5. EXIT");


}


void open_graph()

{

    int   x_center = 320, y_center = 240, rad = 220;

    int   z,k,x[10],y[10];


    setcolor(4);

    for(int xi=30;xi>4;xi--)

    {

    sound(xi*20);

    settextstyle(GOTHIC_FONT,HORIZ_DIR,xi);

    outtextxy(xi,220,"Gaming begins here......");

    delay(320);

    cleardevice();

    }

    setcolor(10);

    for (z=0;z<10;z++)

    {
      x[z]=x_center+rad*cos(36*z*3.14159/180);

      y[z]=y_center+rad*sin(36*z*3.14159/180);

    }

    for (z=0;z<10;z++)

    for (k=0;k<10;k++)

                          {

                          sound(k*200);

                          delay(50);

                          line(x[z],y[z],x[k],y[k]);

                          nosound();

                          }


   for(int h=0;h<500;h++)

   {

   sound(h*9);

setfillstyle( random(12),random(15) );

bar3d(random(640),random(480),random(640),random(480),random(30),

random(18));

   }

  op_graph++;

}

void loading()

{

cleardevice();

  setcolor(13);

  settextstyle(TRIPLEX_FONT,HORIZ_DIR,0);

  setusercharsize(2, 1, 1, 1);
  outttextxy(138,198,"LOADING");              //LOADING
  setcolor(13);
settextstyle(TRIPLEX_FONT,HORIZ_DIR,0);
setusercharsize(2, 1, 1, 1);

setcolor(BLUE);
rectangle(119,199,421,231);
rectangle(118,198,422,232);
setcolor(10);
for(int ii=120;ii<=420;ii++)
{
line(ii,200,ii,230);
sound(ii+1500);
delay(10);
nosound();
}

}
void gameover()
{
cleardevice();
settextstyle(GOTHIC_FONT,0,10);
cout<<"a";
for(int g=0;g<450;g++)
{
sound(g*10);
delay(3);
setcolor(LIGHTRED);
if(g%20==0) {cleardevice();setcolor(12);}
outtextxy(70,g,"game over");
nosound();
}
main();
}

void main()
{
 int gdriver = DETECT, gmode;
 initgraph(&gdriver, &gmode, "c:\tc\bgi");
 cleardevice();
 if(op_graph==0) open_graph();

 settextstyle(TRIPLEX_FONT,HORIZ_DIR,0);
 for(int si=1;si<10;si++)
 {
  sound(si*850);
  setcolor(LIGHTCYAN);
  setusercharsize(si,1,si,1);
  outtextxy(13,150,"GAME PACK");
  delay(250);
  nosound();
  cleardevice();
  }
```

```
randomize();

setfillstyle(3,RED);

bar3d(0, 0,639,479, 0, 0);//background


setcolor(WHITE);

settextstyle(4,HORIZ_DIR,7);

outtextxy(220,25,"MENU");


char lion;

int move=5;


lion=getch();

while(lion!=27)

{


 if(lion==13)  break;

 if(lion=='P') move++;

 if(lion=='H') move--;


 if(move<1)move=5;

 if(move>5)move=1;


 if(move==1)  menu(10,8,8,8,8);

 if(move==2)  menu(8,10,8,8,8);

 if(move==3)  menu(8,8,10,8,8);          //7 gray 14-yellow

 if(move==4)  menu(8,8,8,10,8);

 if(move==5)  menu(8,8,8,8,10);
 lion=getch();

 sound(1700);

 delay(7);

 nosound();

}

cleardevice();

if(move==5) exit(0);




if(move==1)

{

 //bricks video game shooting  prg.


loading();


int gd=DETECT,gm;

initgraph(&gd,&gm,"D:\CPP\TC\BGI");


setcolor(10);

outtextxy(500,75,"score :");

outtextxy(500,150,"level :");

outtextxy(500,225,"Life");

lifebox(10);

putscore(10);

strt();
```

```
 }
/*
  PAIRS II


*/
if(move==2)
{
int a[8],game[16],i=0,z=8,k,flag,temp,t,b,l,j=0,u=0,o=0,p=0;
int left[16],top[16],right[16],bottom[16];


randomize();


loading();


for(int ii=0;ii<=640;ii++)
line(ii,0,ii,480);


setcolor(1);
for(ii=0;ii<=640;ii++)
line(0,ii,640,ii);


setcolor(3);
for(ii=640;ii>=0;ii--)
line(ii,0,ii,480);


setcolor(BLUE);
for(ii=640;ii>=0;ii--)
```

```
line(0,ii,640,ii);


setcolor(RED);

for(ii=0;ii<=640;ii++)

{

sound(ii*5);

line(640,ii,ii,480);

delay(5);

nosound();

}


setcolor(0);

for(ii=0;ii<=640;ii++)

{

sound(ii*5);

line(0,ii,640,480);

delay(2);

nosound();

}



//selecting 8 nos (1-100)without repeating

randomize();

a[0]=random(100);

while(i!=8)

{

redo:
```

```
temp=random(100);

flag=1;

for(k=0;k<=i;k++)

if(a[k]==temp){flag=5;break;}

if (flag==5) goto redo;

else

{

a[i++]=temp;

game[z++]=temp;

}

}


//randomly filling the 8 nos from an array without repeating

int r=0;

while(r!=8)

{

label:

temp=a[random(8)];

flag=1;

for(k=0;k<=r;k++)

if(game[k]==temp){flag=5;break;}

if (flag==5) goto label;

else

game[r++]=temp;

}


setfillstyle(11,RED);
```

```
setcolor(YELLOW);

bar(0, 0,638,478);//backgrond


for(k=0,t=19,b=99;k<4;k++,t+=120,b+=121)     //displaying cards

for(i=0,l=55,r=125;i<4;i++,l+=150,r+=150)

{

left[j++]=l;

top[u++]=t;

right[o++]=r;

bottom[p++]=b;

sound(i*200);

card(l,t,r,b);

}

nosound();

for(int you=1;you<=20;you++)

{

if(you%2)

setcolor(LIGHTGREEN);

else

setcolor(RED);

for(k=0,t=15,b=103;k<4;k++,t+=120,b+=121)      //high light cursor

for(i=0,l=50,r=130;i<4;i++,l+=150,r+=150)

for(int we=1;we<=5;we++)

{

sound(we*900);

rectangle(l-we,t-we,r+we,b+we);

nosound();
```

```
}
delay(100);
}



setcolor(0);
char ch=getch();
int pntr=0,turn=1;

while(ch!=27)
{

                    if(ch=='M') pntr++;
                    if(ch=='K') pntr--;
                    if(ch=='H') pntr=pntr-4;
                    if(ch=='P') pntr+=4;

                    if(pntr>=17)
                    pntr=pntr-16;

                    if(pntr<1)
                    pntr=pntr+16;

                    int m=1;
                    if(open_times==1)
                    if(pntr-1==prev_card)
                    {
```

```
                    ch=getch();

                    continue;

          }

int var_new34=9;


if(game[pntr-1]==-100)

var_new34=0;


          end:

          for(k=0,t=15,b=103;k<4;k++,t+=120,b+=121)      //high light cursor

          for(i=0,l=50,r=130;i<4;i++,l+=150,r+=150)

          {


                    setcolor(BLACK);

                    if(m==pntr)

                    {

                              setcolor(LIGHTGREEN);

                              if(game[pntr-1]==-100)

                              setcolor(LIGHTRED);

                    }


                    for(int we=1;we<=5;we++)

                    rectangle(l-we,t-we,r+we,b+we);

                    m++;

          }


if(var_new34)
if(ch==13)
open(left,top,right,bottom,game,pntr-1);    //open card



int y=0;
for(int wer=0;wer<16;wer++)
if(game[wer]==-100)
y++;
if(y==16) gameover();


 ch=getch();
 sound(2500);
 delay(6);
 nosound();
 }
 if(ch==27) main();


}


if(move==3)
{
char ctrl;
do
 {
char help_topic[]="                 ******************HELP********************
USERS
MANUAL
```

1. PAIRS II

The general outline of the game is as follows.

*

It is basically a card game.

* One can flip and see only two cards at a

time simultaneously in the given set of cards.

* One has to match the

cards in the given set.

* One wins the game if all the matches are made

correctly.

* This game is a time-trailed game i.e. one's score becomes

invalid if it exceeds the tine limit.

* To navigate one can use the arrow

keys.

* To open the card you can hit enter.

" ;

char help_brick[]="

BRICKS

The general outline is as follows.

*It is

a brick game.

*It consist of a shooting box and we have to shoot the

bricks.

*To move the shooting box right press the right direction key and

to move left press the left direction key.

```
*To shoot press the up

direction key.

*Basically there are 5 levels in the game.

*You can win

each level when u reach the score of 2500.

*When u hit each brick you get

25 points.

*But if the bricks touch the shooting box or if it touches the

ground one of your life is lost.

*In total there are 5 lives.";

textcolor(10);

for(int i=0;i<strlen(help_topic);i++)

{

                    sound(i*20);

                    cout<<help_topic[i];

                    delay(15);

                    nosound();

}

int cv;

for(i=0,cv=2000;i<strlen(help_brick);i++,cv--)

{

                    sound(cv*2);

                    cout<<help_brick[i];

                    delay(15);

                    nosound();

}
```

```
 ctrl=getch();
 }while(ctrl!=27&&ctrl<28&&ctrl>25);
 if(ctrl==27) main();
}


if(move==4)
{
 cleardevice();
 setcolor(LIGHTGREEN);
 settextstyle(TRIPLEX_FONT,HORIZ_DIR,0);
 setusercharsize(1,1,1,1);

 setcolor(LIGHTGREEN);
 settextstyle(TRIPLEX_FONT,HORIZ_DIR,6);
 outtextxy(320,250,"&");

 setcolor(LIGHTGREEN);
 rectangle(1,1,638,478);;//background
 for(int ab=5,bc=420;ab<350;ab++,bc--)
 {
 sound(ab*620);

 setfillstyle(6,BLACK);
 bar(5, 130,637,250);
 bar(9,350,635,450);
 setcolor(random(15));

 settextstyle(TRIPLEX_FONT,HORIZ_DIR,6);


 settextstyle(TRIPLEX_FONT,HORIZ_DIR,6);

 sound(bc*2000);

 delay(1);

 nosound();

 }

sleep(1);

main();

}



}//void main's block


void strt()

{

int op;

setcolor(0);

line(60+i*25,j*25,60+rj*25,ri*25);


for(ri=0;ri<19;ri++)

for(rj=0;rj<16;rj++)

a[ri][rj]=0;

randomize();

genbox();

drawbox(12);
```

```
start=clock();

op=getch();

while(1)

{

while(!kbhit())

{


end = clock();

if((end - start) / CLK_TCK > speed)

{

decr();

start=clock();

}


}


while(kbhit())

{

op=getch();

switch(op)

{

case 77:

draw(0);

if(i<15) i++;

draw(10);

break;

case 75:
```

```
draw(0);

if(i>0) i--;

draw(10);

break;

case 72:

      shotbox(i);

      break;

case 27:main();

}

}


}



}


void draw(int color)

{

int c;

c=getcolor();

setcolor(color);

setfillstyle(INTERLEAVE_FILL,color);

bar(50+25*i,j*25,75+25*i,j*25+25);

rectangle(50+25*i,j*25,75+25*i,j*25+25);

setcolor(14);

rectangle(50,0,450,475);
```

```
setcolor(c);

}



void genbox()

{

int x;

for(x=0;x<nbox;x++)

{

rj=rand()%16;

if(a[0][rj]==1) x--;

a[0][rj]=1;

}

}




void decr()

{

if(max<17)

{

drawbox(0);

for(ri=max;ri>=0;ri--)

for(rj=0;rj<16;rj++)

a[ri+1][rj]=a[ri][rj];

for(rj=0;rj<16;rj++)
```

```
a[0][rj]=0;
genbox();
drawbox(12);
}
else
{
printf("a");
lifebox(0);
--life;
lifebox(10);
if(life==0) gameover();


else {drawbox(0);strt();}
}
}

void drawbox(int c)
{
setcolor(c);

for(ri=0;ri<18;ri++)
for(rj=0;rj<16;rj++)
    if(a[ri][rj]==1)
    {
    max=ri;
    setfillstyle(INTERLEAVE_FILL,c);
    bar(50+rj*25,ri*25,50+rj*25+25,ri*25+25);

    if(c!=0) setcolor(14);

    rectangle(50+rj*25,ri*25,50+rj*25+25,ri*25+25);

    }

setcolor(14);

rectangle(50,0,450,475);

}



void shotbox(int rj)

{

int ri,r;

drawbox(0);

for(ri=18;ri>=0;ri--)

if(a[ri][rj]==1)

{a[ri][rj]=0;


file://for(r=18;r>=ri;r--)

setcolor(13);

line(60+i*25,j*25,60+rj*25,ri*25);

sound(480);

delay(25);

nosound();


putscore(0);

if(score%100==0 && score!=0 )

{
```

```
putscore(0);

level++;

putscore(10);


if(speed==1)

if(nbox==5) nbox=7;

else if(nbox==7) nbox=9;

else if(nbox==9) nbox=11;

else

{

setcolor(10);

drawbox(12);

cout<<"a";

cleardevice();

gameover();

main();

}

else speed--;

putscore(0);

score++;

putscore(10);

strt();break;}

else{

score++;

putscore(10);

break;}}
```

```
drawbox(12);

setcolor(0);

line(60+i*25,j*25,60+rj*25,ri*25);



}

void putscore(int color)

{

char str[20];

int c;

c=getcolor();

setcolor(color);

itoa(score*25,str,10);

outtextxy(550,75,str);

itoa(level,str,10);

outtextxy(550,150,str);

setcolor(c);

}


void lifebox(int color)

{

int i,c;

c=getcolor();

setcolor(color);

for(i=1;i<=life;i++)

{

setfillstyle(INTERLEAVE_FILL,color);
bar(500,250+i*25,525,275+i*25);

if(color) setcolor(14);

rectangle(500,250+i*25,525,275+i*25);

}

setcolor(c);

}
```

C++ > Computer Graphics sample source codes

Telephone directory which can be used to store, edit, search data

```
#include<iostream.h>

#include<fstream.h>

#include<graphics.h>

#include<process.h>

#include<io.h>

#include<dos.h>

#include<conio.h>

#include<stdio.h>

#include<string.h>


class telephone

{

 char name[25];

 char address[40];
char phoneno[15];

public:

 telephone(){ }

 telephone(char nam[25], char add[40], char phone[15])

 {

  strcpy(name,nam);

  strcpy(address,add);

  strcpy(phoneno,phone);

 }

 void init();

 void button(int x1,int y1,int x2,int y2,char str[]);

 int press(int,int,int,int);

 int unpress(int,int,int,int);

 int mouseini();

 int showmp();

 int hidemp();

 int getmp(int *button,int *x,int *y);

 int setmp();

 int click(int x1,int y1,int x2,int y2,char str[]);

 int screen();

 void login();

 void search();

 void delete1();

 void insert();

 void update();

 void display()

 {
   cout<<"


Name        : "<<name;
   cout<<"
Address     : "<<address;
   cout<<"
Phone Number: "<<phoneno;
  }
  int compare(char nam1[25])
  {
   if(strcmp(nam1,name)==0)
    return 1;
   else
    return 0;
  }
};
void telephone::login()
{
 setcolor(15);
 line (320-150,320-13,320+150,320-13);
 line (320-150,320+12,320+150,320+12);
 line (320-150,320-13,320-150,320+12);
 line (320+150,320+12,320+150,320-13);
 int s=30,w;
 gotoxy(20,23);
 cout<<"LOADING . . .";
 for (int x1=171,x2=171,y1=308,y2=331,y=1,S=0;x1<470;x1++,x2++,y++,S++)
 {
  setcolor(1);
  line (x1,y1,x2,y2);
  w=(x1-169)/3;
  for (int i=34; i<=78; i++)
  {
   gotoxy(i,23) ;
   cout <<" " ;
  }
  gotoxy(34,23); cout<<w<<"%";
  if (x2>270) s=45; if (x2>370) s=10;
  if (x2==320) delay(999); else
   delay(s);
 }
 delay(800);
 for (int i=27; i<=78; i++)
 {
  gotoxy(i,23) ;
  cout <<" " ;
 }
}

void telephone::insert()
{
 telephone tel1;
 fstream file;
 file.open("Phone.txt",ios::app);
 tel1.init();
 file.write((char *) &tel1, sizeof(tel1));
 setcolor(7);
 outtextxy(250,250,"Inserting Record.....");
 file.close();
 delay(3000);
}
void telephone::init()
{
 cout<<"
Enter Name        : "; cin.getline(name,25);
 cout<<"
Enter Address     : "; cin.getline(address,40);
 cout<<"
Enter Phone Number: "; cin.getline(phoneno,15);
}

void telephone::delete1()
{
 telephone tel1;
 fstream file ;
 char nam[25],nam1[25];
 strcpy(nam1,"");
 cout<<"Enter the Name to be deleted: ";
 cin>>nam;
 file.open("Phone.txt", ios::in) ;
 fstream temp ;
```

```
temp.open("temp.txt", ios::out) ;
file.seekg(0,ios::beg) ;
while (file.read((char *) &tel1, sizeof(telephone)))
{
 if(!tel1.compare(nam))
  temp.write((char *) &tel1, sizeof(telephone)) ;
 else
  strcpy(nam1,nam);
}
file.close() ;
temp.close() ;
setcolor(7);
if(strlen(nam1)!=0)
{
 file.open("Phone.txt", ios::out) ;
 temp.open("temp.txt", ios::in) ;
 temp.seekg(0,ios::beg) ;
 while (temp.read((char *) &tel1, sizeof(telephone)))
 {
  file.write((char *) &tel1, sizeof(telephone)) ;
 }
 file.close() ;
 temp.close() ;
 outtextxy(250,250,"Deleting Record.....");
}
else
 outtextxy(250,250,"Record not found");
```

```
  delay(3000);
}

void telephone::update()
{
 telephone tel1;
 fstream file ;
 char nam[25],nam1[25];
 strcpy(nam1,"");
 cout<<"Enter the Name to be updated: ";
 cin>>nam;
 file.open("Phone.txt", ios::in) ;
 fstream temp ;
 temp.open("temp.txt", ios::out) ;
 file.seekg(0,ios::beg) ;
 while (file.read((char *) &tel1, sizeof(telephone)))
 {
  if(!tel1.compare(nam))
   temp.write((char *) &tel1, sizeof(telephone)) ;
  else
   strcpy(nam1,nam);
 }
 file.close() ;
 temp.close() ;
 setcolor(7);
 if(strlen(nam1)!=0)
 {
  file.open("Phone.txt", ios::out) ;

  temp.open("temp.txt", ios::in) ;

  temp.seekg(0,ios::beg) ;

  while (temp.read((char *) &tel1, sizeof(telephone)))

  {

   file.write((char *) &tel1, sizeof(telephone)) ;

  }

  file.close() ;

  temp.close() ;

  char nam[25],add[40],ph[15];

  file.open("Phone.txt",ios::app);

  cout<<"

Enter Name        : ";

cin.getline(nam,25);cin.getline(nam,25);

  cout<<"

Enter Address     : "; cin.getline(add,40);

  cout<<"

Enter Phone Number: "; cin.getline(ph,15);

  tel1=telephone(nam,add,ph);

  file.write((char *) &tel1, sizeof(tel1));

  file.close();

  outtextxy(250,250,"Updating Record.....");

 }

 else

  outtextxy(250,250,"Record not found");

  delay(3000);

}
```

```
void telephone::search()
{
 fstream file;
 telephone tel1;
 int i=1;
 char nam[25];
 file.open("Phone.txt",ios::in);
 cout<<"Enter name to be Searched: ";
 cin>>nam;
 file.seekg(0,ios::beg);
 while(file.read((char *) &tel1, sizeof(telephone)))
 {
  if(tel1.compare(nam))
  {
   tel1.display();
   i=0;
   break ;
  }
 }
 file.close() ;
 if(i)
 {
  setcolor(7);
  outtextxy(250,250,"Record not found");
 }
 getch();
```

```
}

void telephone:: button(int x1,int y1,int x2,int y2,char str[])
{
 int xc,yc,i=0,l=0;
 while(i<strlen(str))
 {
  l+=4;
  i++;
 }
 xc=(x2-x1)/2+x1-l;
 yc=(y2-y1)/2+y1;
 unpress(x1,y1,x2,y2);
 settextstyle(0,0,0);
 setcolor(11);
 outtextxy(xc,yc,str);
}

int telephone:: unpress(int x1,int y1,int x2,int y2)
{
 setlinestyle(0,1,1);
 setfillstyle(1,1);
 bar(x1,y1,x2,y2);
 setcolor(WHITE);
 line(x1,y1,x2,y1);
 line(x1,y1,x1,y2);
 setcolor(0);
 line(x1,y2,x2,y2);

 line(x2,y1,x2,y2);

 return 0;

}


int telephone:: press(int x1,int y1,int x2,int y2)

{

 setlinestyle(0,1,1);

 setfillstyle(1,1);

 bar(x1,y1,x2,y2);

 setcolor(0);

 line(x1,y1,x2,y1);

 line(x1,y1,x1,y2);

 setcolor(WHITE);

 line(x1,y2,x2,y2);

 line(x2,y1,x2,y2);

 return 0;

}


int telephone:: mouseini()

{

 union REGS i,o;

 i.x.ax=0;

 int86(0x33,&i,&o);

 return(o.x.ax);

}

int telephone:: showmp()
```

```
{
 union REGS i,o;
 i.x.ax=1;
 int86(0x33,&i,&o);
 return 0;
}

int telephone:: hidemp()
{
 union REGS i,o;
 i.x.ax=2;
 int86(0x33,&i,&o);
 return 0;
}

int telephone:: getmp(int *button,int *x,int *y)
{
 union REGS i,o;
 i.x.ax=3;
 int86(0x33,&i,&o);
 *button=o.x.bx;
 *x=o.x.cx;
 *y=o.x.dx;
 return 0;
}

int telephone:: setmp()
```

```
{
 union REGS i,o;
 i.x.ax=4;
 int mx=getmaxx(),my=getmaxy();
 i.x.cx=(3*mx/4)+20;
 i.x.dx=(3*my/4)+20;
 int86(0x33,&i,&o);
 return 0;
}

int telephone:: click(int x1,int y1,int x2,int y2,char str[])
{
 int button,x,y;
 int xc,yc,i=0,l=0;
 while(i<strlen(str))
 {
  l+=4;
  i++;
 }
 xc=(x2-x1)/2+x1-l;
 yc=(y2-y1)/2+y1;
 getmp(&button,&x,&y);
 if( (x>x1 && x<x2) && (y>y1 && y<y2) && button==1)
 {
  hidemp();
  press(x1,y1,x2,y2);
  setcolor(11);
  settextstyle(0,0,0);

  outtextxy(xc,yc,str);

  showmp();

  while((button==1))

   getmp(&button,&x,&y);

  hidemp();

  unpress(x1,y1,x2,y2);

  showmp();

  setcolor(11);

  settextstyle(0,0,0);

  outtextxy(xc,yc,str);

  for(i=50;i<500;i=i+50)

  {

   delay(10);

   sound(i+200);

  }

  showmp();

  nosound();

  setcolor(11);

  settextstyle(0,0,0);

  outtextxy(xc,yc,str);

  return 0;

 }

 else return 1;

}


int telephone:: screen()
```

```
{
 settextstyle(0,1,6);
 setcolor(11);
 outtextxy(100,30,"TELEPHONE");
 outtextxy(600,30,"DIRECTORY");
 setmp();
 button(250,100,400,150,"Insert");
 button(250,150,400,200,"Delete");
 button(250,200,400,250,"Update");
 button(250,250,400,300,"Search");
 button(250,300,400,350,"Exit");
 while(1)
 {
  if(click(250,100,400,150,"Insert")==0)
  {
   cleardevice();
   insert();
   return 0;
  }
  if(click(250,150,400,200,"Delete")==0)
  {
   cleardevice();
   delete1();
   return 0;
  }
  if(click(250,200,400,250,"Update")==0)
  {
   cleardevice();
   update();
   return 0;
  }
  if(click(250,250,400,300,"Search")==0)
  {
   cleardevice();
   search();
   return 0;
  }
  if(click(250,300,400,350,"Exit")==0)
   exit(0);
 }
}

void main()
{
 char user[25]="User Name",*pass,*pass1="user";
 int gdriver=DETECT,gmode;
 initgraph(&gdriver,&gmode,"");
 while(1)
 {
  cleardevice();
  settextstyle(0,0,1);
  outtextxy(250,250,"User Name:");
  outtextxy(250,265,"Password :");
  outtextxy(335,250,user);
  pass=getpass("");
  if(strcmp(pass,pass1)==0)
  {
   cleardevice();
   telephone tel;
   char op[8],cp[8],np[8];
   tel.login();
   while(1)
   {
    cleardevice();
    tel.mouseini();
    tel.showmp();
    tel.screen();
   }
  }
  else
  {
   cleardevice();
   settextstyle(0,0,2);
   outtextxy(250,250,"Illegal User....");
   delay(3000);
  }
 }
}
```

C++ > Computer Graphics sample source codes

Tower of Hanoi - A Graphical Representation

```
// [ You can use more than 10 Disks too, just change the value of MAX ]

//


#include <graphics.h>

#include <stdlib.h>

#include <stdio.h>

#include <conio.h>


#define MAX     12

#define BegPos   105

#define AuxPos   305

#define EndPos   505


int width;

typedef struct disc

          {

                 char val1[MAX];

                 char top,pos;

                }; 


void push(disc *tt,int x);

pop(disc *tt);

void tower(int,disc *,disc *,disc *);

void draw_stack(disc *beg,disc *,disc *);

int main(void)

{
  int gdriver = DETECT, gmode, errorcode;

  int i,x=2;

  disc beg,end,aux;

  printf("



                              TOWER OF HANOI

");

  printf("=====================================================");

  printf("


How Many Disks[1-10]:-  ");

  scanf("%d",&x);


  initgraph(&gdriver, &gmode, "d:\TC\BGI");

  errorcode = graphresult();

  if (errorcode != grOk)

  {

          printf("Graphics error: %s

", grapherrormsg(errorcode));

          printf("Press any key to halt:");

          getch();

          exit(1);

  }

         width=50/x;
```

```
            beg.top=end.top=aux.top=0;

            beg.pos=1;end.pos=3;aux.pos=2;


            for(i=0;i<x;i++)

                        push(&beg,(x-i)+1);


            draw_stack(&beg,&end,&aux);

            tower(x,&beg,&end,&aux);


    closegraph();

    return 0;

}

void tower(int n,disc *beg,disc *aux,disc *end)

{

            if(n>0)

/*          {

                        push(end,pop(beg));

                        draw_stack(beg,end,aux);

            }

            else*/

            {

                        tower(n-1,beg,end,aux);

                        push(end,pop(beg));

                        draw_stack(beg,end,aux);

                        tower(n-1,aux,beg,end);

            }

//
```

```
}

void push(disc *tt,int x)

{

        tt->val1[tt->top]=x;

        tt->top++;

}


pop(disc *tt)

{

        int a;

        tt->top--;

        a=tt->val1[tt->top];

        tt->val1[tt->top]=0;

        return a;

}


void draw_stack(disc *beg,disc *end,disc *aux)

{

        int ypos=295,i,height=10,xpos;

        int ver=0;

        cleardevice();


        setfillstyle(1,2);

        bar(20,300,580,310);


        bar(100,100,110,300);

        bar(300,100,310,300);
        bar(500,100,510,300);


        rectangle(20,300,580,310);


        rectangle(100,100,110,300);
        rectangle(300,100,310,300);
        rectangle(500,100,510,300);


        /* END TOWER*/
        ypos=295;
        if(end->pos==1)
                xpos=BegPos;
        else if(end->pos==2)
                xpos=AuxPos;
        else if(end->pos==3)
                xpos=EndPos;


        for(i=0;i<end->top;i++)
        {
                setfillstyle(end->val1[i],end->val1[i]);


bar(xpos-(end->val1[i]*width),ypos,xpos+(end->val1[i]*width),ypos-height);


rectangle(xpos-(end->val1[i]*width),ypos,xpos+(end->val1[i]*width),ypos-height);
                ypos-=(height+2);
        }
        ver=end->pos;
        /* BEG TOWER*/
        if(beg->pos==1)
                xpos=BegPos;
        else if(beg->pos==2)
                xpos=AuxPos;
        else if(beg->pos==3)
                xpos=EndPos;

        ypos=295;
        for(i=0;i<beg->top;i++)
        {
                setfillstyle(beg->val1[i],beg->val1[i]);

bar(xpos-(beg->val1[i]*width),ypos,xpos+(beg->val1[i]*width),ypos-height);

rectangle(xpos-(beg->val1[i]*width),ypos,xpos+(beg->val1[i]*width),ypos-height);
                ypos-=(height+2);
        }

        /* AUX TOWER*/
        ver=ver*10+beg->pos;

        if(ver<20)
        {
                if(ver%10==2)
                        xpos=EndPos;
```

```
				else
					xpos=AuxPos;
			}
			else if(ver<30)
			{
				if(ver%10==1)
					xpos=EndPos;
				else
					xpos=BegPos;
			}
			else if(ver<40)
			{
				if(ver%10==1)
					xpos=AuxPos;
				else
					xpos=BegPos;
			}

			ypos=295;
			for(i=0;i<aux->top;i++)
			{
				setfillstyle(aux->val1[i],aux->val1[i]);

		bar(xpos-(aux->val1[i]*width),ypos,xpos+(aux->val1[i]*width),ypos-height);

		rectangle(xpos-(aux->val1[i]*width),ypos,xpos+(aux->val1[i]*width),ypos-height);
				ypos-=(height+2);
```

```
	}
	getch();
}
```

C++ > Computer Graphics sample source codes

Two-Dimension Transformation In Homogeneous Coordinate

This Program Deals With All Two-D Transformation Such As Translation, Scaling, Rotation, Reflection, Shearing In Homogeneous Coordinates.

Code :

```
//TwoDimensional Transformations In Homogeneous

#include<graphics.h>
#include<iostream.h>
#include<Math.h>
#include<conio.h>
#define MAXSIZE 3

class D_2
{
	private:
	double Points[MAXSIZE][MAXSIZE];
	void Mult(double [MAXSIZE][MAXSIZE]);
```

```
        void MultTwoMat(double [MAXSIZE][MAXSIZE],double [MAXSIZE][MAXSIZE]);

        void Print();

        int x,y;

        public:

        D_2();

        void initialize();

        void GetPoints();

        void Draw(int);

        void DrawCord();

        void Translate();

        void Rotate();

        void Reflect();

        void Display(double[MAXSIZE][MAXSIZE]);

        void Shear();

        void Scale_Fixed();

        void Scale_Dir();

};



D_2::D_2()

{

        for(int i=0;i<MAXSIZE;i++)

        {

                for(int j=0;j<MAXSIZE;j++)

                {

                        if(i == (MAXSIZE-1))

                        Points[i][j] = 1;
                        else
                        Points[i][j] = 0;
                }
          }
        initialize();
        x = getmaxx();
        y = getmaxy();
}
void D_2::initialize()
        {
        int gdrive = DETECT,gmode;
        initgraph(&gdrive,&gmode,"c:    cgi");
        }

void D_2::GetPoints()
{
        closegraph();
        cout<<"Enter The Points Of The Triangle.
";
        for(int j=0;j<MAXSIZE;j++)
        {
        cout<<"Enter Point "<<j+1<<":-";
                for(int i=0;i<MAXSIZE-1;i++)
                {
                        cout<<"
Enter "<<char(i+'X')<<": ";
                        cin>>Points[i][j];
                }
            }
            initialize();
}


void D_2::Mult(double temp[MAXSIZE][MAXSIZE])
{
            int i,j,k;
            double z[MAXSIZE][MAXSIZE];
            for(i=0;i<MAXSIZE;i++)
             {
                     for(j=0;j<MAXSIZE;j++)
                              z[i][j]=0;
             }

             for(i=0;i<MAXSIZE;i++)
              {
                     for(j=0;j<MAXSIZE;j++)
                     {

                     for(k=0;k<MAXSIZE;k++)
                              z[i][j]=z[i][j]+(temp[i][k] * Points[k][j]);
                     }
              }

              for(i=0;i<MAXSIZE;i++)
              {
                for(j=0;j<MAXSIZE;j++)
                {
                        Points[i][j] = z[i][j];
          }
        }
}




void D_2::Draw(int color)
{
        int Poly[2*MAXSIZE+2];
        int k = 0;
        if(color == GREEN)
                DrawCord();
        for(int j=0;j<MAXSIZE;j++)
        {
                for(int i=0;i<MAXSIZE-1;i++)
                {
                if(i==0)
                        Poly[k++] = x/2+Points[i][j];
                else
                        Poly[k++] = y/2-Points[i][j];
                }
        }
                Poly[k++] = Poly[0];
                Poly[k]   = Poly[1];
```

```
            setcolor(color);

            drawpoly(4,Poly);

}


void D_2::Display(double Mat[MAXSIZE][MAXSIZE])

{

 for(int i=0;i<MAXSIZE;i++)

            {

             for(int j=0;j<MAXSIZE;j++)

             {

                        cout<<Mat[i][j]<<"         ";

             }

             cout<<"

";

            }

}


void D_2::Print()

{

            setcolor(GREEN);

            setfillstyle(SOLID_FILL,GREEN);

            fillellipse(19,36,2,2);

            outtextxy(23,34," Original Triangle");

            setcolor(MAGENTA);

            setfillstyle(SOLID_FILL,MAGENTA);

            fillellipse(x-178,y-32,2,2);

            outtextxy(x-175,y-34," Tranformed Triangle");
```

```
}


void D_2::DrawCord()

{

            setcolor(12);

            line(x/2,0,x/2,y);

            line(0,y/2,x,y/2);

            setcolor(10);

            setfillstyle(SOLID_FILL,10);

            fillellipse(x/2,y/2,2,2);

            for(int i=(x/2+50),j=(x/2-50);i<=x,j>=0;i=i+50,j=j-50)

            {

                        fillellipse(i,y/2,2,2);

                        fillellipse(j,y/2,2,2);

            }

            for(i=(y/2+50),j=(y/2-50);i<=x,j>=0;i=i+50,j=j-50)

            {

                        fillellipse(x/2,i,2,2);

                        fillellipse(x/2,j,2,2);

            }


                        outtextxy(x/2+3,y/2+4,"0");


                        outtextxy(x/2+45,y/2+5,"50");

                        outtextxy(x/2+95,y/2+5,"100");

                        outtextxy(x/2+145,y/2+5,"150");

                        outtextxy(x/2+195,y/2+5,"200");
```

```
                    outtextxy(x/2+245,y/2+5,"250");
                    outtextxy(x/2+295,y/2+5,"300");

                    outtextxy(x/2-65,y/2+5,"-50");
                    outtextxy(x/2-115,y/2+5,"-100");
                    outtextxy(x/2-165,y/2+5,"-150");
                    outtextxy(x/2-215,y/2+5,"-200");
                    outtextxy(x/2-265,y/2+5,"-250");
                    outtextxy(x/2-315,y/2+5,"-300");

                    outtextxy(x/2+5,y/2+45,"-50");
                    outtextxy(x/2+5,y/2+95,"-100");
                    outtextxy(x/2+5,y/2+145,"-150");
                    outtextxy(x/2+5,y/2+195,"-200");

                    outtextxy(x/2+5,y/2-50,"50");
                    outtextxy(x/2+5,y/2-100,"100");
                    outtextxy(x/2+5,y/2-150,"150");
                    outtextxy(x/2+5,y/2-200,"200");

}

void D_2::MultTwoMat(double temp[MAXSIZE][MAXSIZE],double
temp1[MAXSIZE][MAXSIZE])
{
          int i,j,k;
          double z[MAXSIZE][MAXSIZE];
```

```
            for(i=0;i<MAXSIZE;i++)
             {
                     for(j=0;j<MAXSIZE;j++)
                              z[i][j]=0;
             }

             for(i=0;i<MAXSIZE;i++)
              {
                     for(j=0;j<MAXSIZE;j++)
                     {

                     for(k=0;k<MAXSIZE;k++)
                              z[i][j]=z[i][j]+(temp[i][k] * temp1[k][j]);
                     }
              }

              for(i=0;i<MAXSIZE;i++)
              {
                     for(j=0;j<MAXSIZE;j++)
                     {
                             temp1[i][j] = z[i][j];
              }
            }
}

void D_2::Translate()
```

```
{
        int Tx,Ty;
        double Trans[MAXSIZE][MAXSIZE];
        closegraph();
        cout<<"Enter Translation Factor Along X-Axis: ";
        cin>>Tx;
        cout<<"Enter Translation Factor Along Y-Axis: ";
        cin>>Ty;
        initialize();
        for(int j=0;j<MAXSIZE;j++)
        {
                for(int i=0;i<MAXSIZE;i++)
                {
                        if(i==j)
                                Trans[i][j] = 1;
                  else
                                Trans[i][j] = 0;
                }
        }

        Trans[0][MAXSIZE-1]  =  Tx;
        Trans[1][MAXSIZE-1]  =  Ty;
        Draw(GREEN);
        Mult(Trans);
        Draw(MAGENTA);
        Print();
}
```

```
void D_2::Rotate()

{

            double ang;

            const double PI = 22.0/7;

            double xr,yr;

            double TransMat[MAXSIZE][MAXSIZE];

            double RotMat[MAXSIZE][MAXSIZE];

            double InvTransMat[MAXSIZE][MAXSIZE];



            closegraph();

            cout<<"Enter Angle Of Rotation: ";

            cin>>ang;

            cout<<"Enter Point Of Rotation:

X: ";

            cin>>xr;

            cout<<"

Y: ";

            cin>>yr;

            initialize();

            ang = (PI * ang)/180.0;

            setcolor(YELLOW);

            setfillstyle(SOLID_FILL,YELLOW);

            fillellipse(x/2+xr,y/2-yr,2,2);

            outtextxy(x/2+xr,y/2-yr-2,"  Point Of Rotation");



            //Transformation Matrix
```

```
//Translate arbitrary point to origin then rotate then translate back.

for(int i=0;i<MAXSIZE;i++)

{

        for(int j=0;j<MAXSIZE;j++)

        {

                if(i == j)

                {

                                TransMat[i][j] = 1;

                                InvTransMat[i][j] = 1;

                                RotMat[i][j] = 1;

                }

                else

                {

                                TransMat[i][j] = 0;

                                InvTransMat[i][j] = 0;

                                RotMat[i][j] = 0;

                }

        }

}



                TransMat[0][2] = -xr;

                TransMat[1][2] = -yr;



                InvTransMat[0][2] = xr;

                InvTransMat[1][2] = yr;



                RotMat[0][0] = cos(ang);
```

```
                    RotMat[0][1] = -sin(ang);
                    RotMat[1][0] = sin(ang);
                    RotMat[1][1] = cos(ang);

        Draw(GREEN);
        Print();
        MultTwoMat(InvTransMat,RotMat);
        MultTwoMat(RotMat,TransMat);
        Mult(TransMat);
        Draw(MAGENTA);
}

void D_2::Reflect()
{
        double ang;
        double a,b,c;
        double xr,yr;
        double TransMat[MAXSIZE][MAXSIZE];
        double RotMat[MAXSIZE][MAXSIZE];
        double InvTransMat[MAXSIZE][MAXSIZE];
        double InvRotMat[MAXSIZE][MAXSIZE];
        double RefMat[MAXSIZE][MAXSIZE];

        closegraph();
        cout<<"Enter The Line (ax+by+c=0): ";
        cout<<"
a: ";
```

```
            cin>>a;

            cout<<"

b: ";

            cin>>b;

            cout<<"

c: ";

            cin>>c;

            if(b!=0)

            {

                        yr = (-c/b);

                        xr = 0;

                        double m = -a/b;

                        ang = atan(m);


            }

            else

            {

                        yr = 0;

                        xr = (-c/a);

                        ang = 22.0/14.0;   // Angle = PI/2

            }



            initialize();

            //Transformation Matrix

            //Translate arbitrary point to origin then rotate then translate back.

            for(int i=0;i<MAXSIZE;i++)
```

```
{
        for(int j=0;j<MAXSIZE;j++)
        {
                if(i == j)
                {
                                TransMat[i][j] = 1;
                                InvTransMat[i][j] = 1;
                                RotMat[i][j] = 1;
                                InvRotMat[i][j] = 1;
                                RefMat[i][j] = 1;


                }
                else
                {
                                TransMat[i][j] = 0;
                                InvTransMat[i][j] = 0;
                                RotMat[i][j] = 0;
                                InvRotMat[i][j] = 0;
                                RefMat[i][j] = 0;
                }
        }
 }


                TransMat[0][2] = -xr;
                TransMat[1][2] = -yr;


                InvTransMat[0][2] = xr;
```

```
            InvTransMat[1][2] = yr;

            RotMat[0][0] = cos(ang);
            RotMat[0][1] = sin(ang);
            RotMat[1][0] = -sin(ang);
            RotMat[1][1] = cos(ang);

            InvRotMat[0][0] = cos(ang);
            InvRotMat[0][1] = -sin(ang);
            InvRotMat[1][0] = sin(ang);
            InvRotMat[1][1] = cos(ang);

        for(i=0;i<MAXSIZE;i++)
        {
            for(int j=0;j<MAXSIZE;j++)
            {
                if(i==j)
                    RefMat[i][j] = pow(-1,i)*1;
                else
                    RefMat[i][j] = 0;
        }
}

    Print();
    Draw(GREEN);
    MultTwoMat(InvTransMat,InvRotMat);
    MultTwoMat(InvRotMat,RefMat);
```

```
            MultTwoMat(RefMat,RotMat);
            MultTwoMat(RotMat,TransMat);
            Mult(TransMat);
            Draw(MAGENTA);
}

void D_2::Shear()
{
            double ang;
            double a,b,c;
            double xr,yr,shx;
            double TransMat[MAXSIZE][MAXSIZE];
            double RotMat[MAXSIZE][MAXSIZE];
            double InvTransMat[MAXSIZE][MAXSIZE];
            double InvRotMat[MAXSIZE][MAXSIZE];
            double ShearMat[MAXSIZE][MAXSIZE];

            closegraph();
            cout<<"Enter The Line (ax+by+c=0): ";
            cout<<"
a: ";
            cin>>a;
            cout<<"
b: ";
            cin>>b;
            cout<<"
c: ";
```

```
cin>>c;

cout<<"Enter Shearing Factor Along X-Axis: ";

cin>>shx;

if(b!=0)

{

        yr = (-c/b);

        xr = 0;

        double m = -a/b;

        ang = atan(m);


}

else

{

        yr = 0;

        xr = (-c/a);

        ang = 22.0/14.0;   // Angle = PI/2

}



initialize();


//Transformation Matrix

for(int i=0;i<MAXSIZE;i++)

{

        for(int j=0;j<MAXSIZE;j++)

        {

                if(i == j)
                {
                        TransMat[i][j] = 1;

                        InvTransMat[i][j] = 1;

                        RotMat[i][j] = 1;

                        InvRotMat[i][j] = 1;

                        ShearMat[i][j] = 1;


                }

                else

                {

                        TransMat[i][j] = 0;

                        InvTransMat[i][j] = 0;

                        RotMat[i][j] = 0;

                        InvRotMat[i][j] = 0;

                        ShearMat[i][j] = 0;

                }

        }

}


                TransMat[0][2] = -xr;

                TransMat[1][2] = -yr;


                InvTransMat[0][2] = xr;

                InvTransMat[1][2] = yr;


                RotMat[0][0] = cos(ang);

                RotMat[0][1] = sin(ang);
```

```
                        RotMat[1][0] = -sin(ang);

                        RotMat[1][1] = cos(ang);



                        InvRotMat[0][0] = cos(ang);

                        InvRotMat[0][1] = -sin(ang);

                        InvRotMat[1][0] = sin(ang);

                        InvRotMat[1][1] = cos(ang);



                        ShearMat[0][1] = shx;



             Print();

             Draw(GREEN);

             MultTwoMat(InvTransMat,InvRotMat);

             MultTwoMat(InvRotMat,ShearMat);

             MultTwoMat(ShearMat,RotMat);

             MultTwoMat(RotMat,TransMat);

             Mult(TransMat);

             Draw(MAGENTA);

}



void D_2::Scale_Fixed()

{

            double sx,sy;

            double xr,yr;

            double TransMat[MAXSIZE][MAXSIZE];

            double ScaleMat[MAXSIZE][MAXSIZE];

            double InvTransMat[MAXSIZE][MAXSIZE];
```

```
	closegraph();
	cout<<"Enter The Scaling Factor Along X-Axis: ";
	cin>>sx;
	cout<<"Enter The Scaling Factor Along Y-Axis: ";
	cin>>sy;
	cout<<"Enter Point Of Scaling:
X: ";
	cin>>xr;
	cout<<"
Y: ";
	cin>>yr;
	initialize();

	//Transformation Matrix
	for(int i=0;i<MAXSIZE;i++)
	{
		for(int j=0;j<MAXSIZE;j++)
		{
			if(i == j)
			{
					TransMat[i][j] = 1;
					InvTransMat[i][j] = 1;
					ScaleMat[i][j] = 1;
			}
			else
			{
```

```
                                        TransMat[i][j] = 0;
                                        InvTransMat[i][j] = 0;
                                        ScaleMat[i][j] = 0;
                        }
                }
        }

                        TransMat[0][2] = -xr;
                        TransMat[1][2] = -yr;

                        InvTransMat[0][2] = xr;
                        InvTransMat[1][2] = yr;

                        ScaleMat[0][0] = sx;
                        ScaleMat[1][1] = sy;

        Draw(GREEN);
        Print();
        MultTwoMat(InvTransMat,ScaleMat);
        MultTwoMat(ScaleMat,TransMat);
        Mult(TransMat);
        Draw(MAGENTA);
}
void D_2::Scale_Dir()
{
        double sx,sy;
        double ang;
```

```
const double PI = 22.0/7;

double RotMat[MAXSIZE][MAXSIZE];

double ScaleMat[MAXSIZE][MAXSIZE];

double InvRotMat[MAXSIZE][MAXSIZE];


closegraph();

cout<<"Enter The Scaling Factor Along X-Axis: ";

cin>>sx;

cout<<"Enter The Scaling Factor Along Y-Axis: ";

cin>>sy;

cout<<"Enter The Direction Of Scaling: ";

cin>>ang;

ang = (PI * ang)/180.0;

initialize();



//Transformation Matrix

for(int i=0;i<MAXSIZE;i++)

{

        for(int j=0;j<MAXSIZE;j++)

        {

                if(i == j)

                {

                                RotMat[i][j] = 1;

                                InvRotMat[i][j] = 1;

                                ScaleMat[i][j] = 1;

                }
```

```
                    else
                    {
                                        RotMat[i][j] = 0;
                                        InvRotMat[i][j] = 0;
                                        ScaleMat[i][j] = 0;
                    }
          }
}

                    RotMat[0][0] = cos(ang);
                    RotMat[0][1] = sin(ang);
                    RotMat[1][0] = -sin(ang);
                    RotMat[1][1] = cos(ang);

                    InvRotMat[0][0] = cos(ang);
                    InvRotMat[0][1] = -sin(ang);
                    InvRotMat[1][0] = sin(ang);
                    InvRotMat[1][1] = cos(ang);

                    ScaleMat[0][0] = sx;
                    ScaleMat[1][1] = sy;

Draw(GREEN);
Print();
MultTwoMat(RotMat,ScaleMat);
MultTwoMat(ScaleMat,InvRotMat);
Mult(InvRotMat);
```

```
             Draw(MAGENTA);
}

void main()
{
            D_2 D1;
            D1.DrawCord();
            getch();
            int ch;

            D1.GetPoints();
            D1.Draw(GREEN);
            getch();

            do
            {
                        closegraph();
                        clrscr();
                        cout<<"1.To ReDraw The Triangle.
";
                        cout<<"2.Translate The Triangle.
";
                        cout<<"3.Scaling The Triangle About Fixed Point.
";
                        cout<<"4.Scaling The Triangle In A Direction.
";
                        cout<<"5.Rotating The Triangle About Arbitrary Point.
```

```
";
                cout<<"6.Reflecting The Triangle About Arbitrary Line.
";
                cout<<"7.Shearing Of The Triangle.
";
                cout<<"8.Exit.
";
                cout<<"Enter The Choice: ";
                cin>>ch;
                D1.initialize();
        switch(ch)
        {
         case 1:
                        D1.GetPoints();
                        D1.Draw(GREEN);
                        getch();
                        break;

        case 2:
                        cleardevice();
                        D1.Translate();
                        getch();
                        closegraph();
                        break;
        case 3:
                        cleardevice();
                        D1.Scale_Fixed();
                        getch();

                        closegraph();

                        break;

        case 4:

                    cleardevice();

                    D1.Scale_Dir();

                    getch();

                    closegraph();

                    break;

        case 5:

                        cleardevice();

                        D1.Rotate();

                        getch();

                        closegraph();

                        break;

        case 6:

                        cleardevice();

                        D1.Reflect();

                        getch();

                        closegraph();

                        break;


         case 7:

                        cleardevice();

                        D1.Shear();

                        getch();

                        closegraph();
```

```
                    break;
        case 8:
                    return;
        default:
                    cout<<"
WRONG CHOICE.
";
                    getch();
                    break;
        }
 }while(1);
}
```

C++ > Computer Graphics sample source codes

Design of Clock in Turbo C++ 3.0 graphics

```
#include<stdio.h>
#include<process.h>
#include<iostream.h>
#include<dos.h>
#include<graphics.h>
#include<conio.h>
#include<math.h>
void draw()
```

```
{
setbkcolor(0);
setlinestyle(0,0,0);
setcolor(9);
circle(320,240,3);
setcolor(11);
setfillstyle(6,13);
circle(320,240,150);
circle(320,240,165);
floodfill(156,242,11);
settextstyle(2,0,5);
setcolor(14);
outtextxy(314,98,"12");
outtextxy(384,114,"1");
outtextxy(434,163,"2");
outtextxy(454,230,"3");
outtextxy(317,369,"6");
outtextxy(177,230,"9");
outtextxy(436,300,"4");
outtextxy(195,302,"8");
outtextxy(195,163,"10");
outtextxy(244,112,"11");
outtextxy(388,353,"5");
outtextxy(248,353,"7");
}
main()
{
```

```
int gd=0,gm;

initgraph(&gd,&gm,"c:\tc\bgi");


draw();

//line(320,240,320,130);

//line(320,240,320,150);

//getch();

float s;

float df;

//s=282*M_PI/180;

//float angle=4.712389;

//float an=4.712389;

float anf=4.712389;


//float angle=0;

int x,y;

int q,w;

int ta,d;


float as;

as=6*M_PI/180;


int c2=0;

int count=0;

struct  time t;

gettime(&t);
```

```
float angle=4.712389+t.ti_sec*.1047198;

float an=4.712389+t.ti_min*.1047198;

while(!kbhit())

{

draw();

gettime(&t);

gotoxy(5,5);

angle=4.712389+t.ti_sec*.1047198;

an=4.712389+t.ti_min*.1047198;

anf=4.712389+t.ti_hour*5*.1047198 ;

if(t.ti_min>=12&&t.ti_min<24)

{

anf=anf+2*.1047198;

}

if(t.ti_min>=24&&t.ti_min<36)

{

anf=anf+(3*.1047198);

}

if(t.ti_min>=36&&t.ti_min<48)

{

anf=anf+(4*.1047198);

}

if(t.ti_min>=48&&t.ti_min<60)

{

anf=anf+(5*.1047198);

}
```

```
gotoxy(2,2);
printf("The current time is: %d: %d: %d
",
    t.ti_hour, t.ti_min, t.ti_sec, t.ti_hund);

cout<<"   ";
setlinestyle(0,0,0);
setcolor(0);
line(320,240,x,y);
line(320,240,q,w);
line(320,240,ta,d);

x=320+140*cos(angle);
y=240+140*sin(angle);
q=320+122*cos(an);
w=240+122*sin(an);
ta=320+86*cos(anf);
d=240+86*sin(anf);
setcolor(10);
setlinestyle(0,0,0);
line(320,240,x,y);
setlinestyle(0,0,2);
setcolor(9);
line(320,240,q,w);
setlinestyle(0,0,3);
setcolor(4);
```

```
line(320,240,ta,d);

angle+=.1047198;

delay(1000);

count++;


/*if(c2==12)

{

setlinestyle(0,0,3);

c2=0;

anf+=.1047198;

} */


}



getch();

}
```

C++ > Computer Graphics sample source codes

Analog clock and calendar

```
#include<stdio.h>

#include<math.h>

#include<iostream.h>

#include<conio.h>
```

```
#include<graphics.h>

#include<stdlib.h>

#include<dos.h>

#include<string.h>

//CLOCK CLASS

class clock

{

int h,m,s,thetamin,thetasec;

struct  time t;

public:

void time();

};

void clock::time()

{

int x=540,y=280,r=200,i;

char n[12][3]={"3","2","1","12","11","10","9","8","7","6","5","4"};

struct REGPACK reg;


setcolor(15);

circle(x,y,88);

circle(x,y,89);

setcolor(6);

settextstyle(5,0,1);

for(i=0;i<12;i++)

{

if(i!=3)

outtextxy(x+(r-132)*cos(M_PI/6*i)-8,y-(r-132)*sin(M_PI/6*i)-16,n[i]);
```

```
else

outtextxy(x+(r-132)*cos(M_PI/6*i)-10,y-(r-132)*sin(M_PI/6*i)-16,n[i]);

}

gettime(&t);


printf("
");
printf("%2d:%02d:%02d",t.ti_hour, t.ti_min,t.ti_sec);


reg.r_ax=3;
intr(0x33,®);
while(reg.r_bx!=1)
{
reg.r_ax=3;
intr(0x33,®);
setcolor(5);
setfillstyle(1,3);
circle(x,y,4);
floodfill(x,y,5);
gettime(&t);
if(t.ti_min!=m)
{
setcolor(0);
line(x,y,x+(r-150)*cos(thetamin*(M_PI/180)),y-(r-150)*sin(thetamin*(M_PI/1
80)));
circle(x+(r-200)*cos(thetamin*(M_PI/180)),y-(r-200)*sin(thetamin*(M_PI/180
)),10);
line(x,y,x+(r-165)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-165)*sin(M_PI/6*h
-((m/2)*(M_PI/180))));
circle(x+(r-200)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-200)*sin(M_PI/6*h-(
(m/2)*(M_PI/180))),10);
        }
if(t.ti_hour>12)
t.ti_hour=t.ti_hour-12;
if(t.ti_hour<4)
h=abs(t.ti_hour-3);
else
h=15-t.ti_hour;
m=t.ti_min;
if(t.ti_min<=15)
thetamin=(15-t.ti_min)*6;
else
thetamin=450-t.ti_min*6;
if(t.ti_sec<=15)
thetasec=(15-t.ti_sec)*6;
else
thetasec=450-t.ti_sec*6;
setcolor(3);
line(x,y,x+(r-165)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-165)*sin(M_PI/6*h
-((m/2)*(M_PI/180))));
circle(x+(r-200)*cos(M_PI/6*h-((m/2)*(M_PI/180))),y-(r-200)*sin(M_PI/6*h-(
(m/2)*(M_PI/180))),5);
line(x,y,x+(r-150)*cos(thetamin*(M_PI/180)),y-(r-150)*sin(thetamin*(M_PI/1
80)));
circle(x+(r-200)*cos(thetamin*(M_PI/180)),y-(r-200)*sin(thetamin*(M_PI/180
)),5);
```

```
setcolor(15);

line(x,y,x+(r-145)*cos(thetasec*(M_PI/180)),y-(r-145)*sin(thetasec*(M_PI/1

80)));

delay(100);

setcolor(0);

line(x,y,x+(r-145)*cos(thetasec*(M_PI/180)),y-(r-145)*sin(thetasec*(M_PI/1

80)));

}


}

//CALENDAR CLASS

class calendar

{

int mon,year,d;

static int s;

clock t;

public:

calendar()

{

year=2006;

mon=5;

  }

  int tday();

  void reqmon();

  void cal();

  void chose();

  displaymenu(char **month,int count,int x1,int y1);
```

```
   getresponse(char **month,int width,int count,int x1,int y1);
   highlight(char **month,int ch,int h,int x1,int y1,int width);
   dehighlight(char **month,int ch,int h,int x1,int y1,int width);
};
int calendar:: tday()
{
int t,total=1;
int days[]={31,28,31,30,31,30,31,31,30,31,30,31};
   for(t=1;t<year;t++)
   {
    if(t%4==0)
    total=total+366;
    else
    total=total+365;
    }
  if(year%4==0)
  days[1]=29;
    for(t=0;t<mon-1;t++)
    {
    total=total+days[t];
    }
  d=total%7;
  return d;
  }
void calendar::reqmon()
{
```

```
int q,r,x1=40,y1=210;

int days[]={31,28,31,30,31,30,31,31,30,31,30,31};

char st2[3],st3[9],st4[5];


  q=days[mon-1];

  settextstyle(1,0,2);

  setcolor(YELLOW);

  for(r=1;r<=d;r++)

  {

    x1+=62;

    s++;

  }

  for(r=1;r<=q;r++)

  {

    itoa(r,st2,10);

    if(s>=6)

    {

            outtextxy(x1,y1,st2);

            y1+=30;

            x1=40;

            s=0;

            continue;

    }

    outtextxy(x1,y1,st2);

    x1+=60;

    s++;

  }
  s=0;

  chose();


}

 void calendar::cal()

 {

  cleardevice();

  setgraphmode(getgraphmode());

  int l=17,t=175,r=70,b=235,g,x=25,y=177;

  char *day[]={"SUN","MON","TUE","WED","THU","FRI","SAT"};

  char st1[4];

  setbkcolor(0);

  setcolor(5);

  settextstyle(1,0,7);

  outtextxy(40,5,"Calendar & Clock");

  setfillstyle(3,BLUE);

  bar(10,165,440,395);

  setfillstyle(1,0);

  bar3d(l,t,r,b,0,0);

  bar3d(l,t+30,r,b+30,0,0);

  bar3d(l,t+60,r,b+60,0,0);

  bar3d(l,t+90,r,b+90,0,0);

  bar3d(l,t+120,r,b+120,0,0);

  bar3d(l,t+150,r,b+150,0,0);

  bar3d(l,t+180,r,b+150,0,0);

  bar3d(l+60,t,r+60,b,0,0);

  bar3d(l+60,t+30,r+60,b+30,0,0);
```

```
bar3d(l+60,t+60,r+60,b+60,0,0);

bar3d(l+60,t+90,r+60,b+90,0,0);

bar3d(l+60,t+120,r+60,b+120,0,0);

bar3d(l+60,t+150,r+60,b+150,0,0);

bar3d(l+60,t+180,r+60,b+150,0,0);

bar3d(l+120,t,r+120,b,0,0);

bar3d(l+120,t+30,r+120,b+30,0,0);

bar3d(l+120,t+60,r+120,b+60,0,0);

bar3d(l+120,t+90,r+120,b+90,0,0);

bar3d(l+120,t+120,r+120,b+120,0,0);

bar3d(l+120,t+150,r+120,b+150,0,0);

bar3d(l+120,t+180,r+120,b+150,0,0);

bar3d(l+180,t,r+180,b,0,0);

bar3d(l+180,t+30,r+180,b+30,0,0);

bar3d(l+180,t+60,r+180,b+60,0,0);

bar3d(l+180,t+90,r+180,b+90,0,0);

bar3d(l+180,t+120,r+180,b+120,0,0);

bar3d(l+180,t+150,r+180,b+150,0,0);

bar3d(l+180,t+180,r+180,b+150,0,0);

bar3d(l+240,t,r+240,b,0,0);

bar3d(l+240,t+30,r+240,b+30,0,0);

bar3d(l+240,t+60,r+240,b+60,0,0);

bar3d(l+240,t+90,r+240,b+90,0,0);

bar3d(l+240,t+120,r+240,b+120,0,0);

bar3d(l+240,t+150,r+240,b+150,0,0);

bar3d(l+240,t+180,r+240,b+150,0,0);

bar3d(l+300,t,r+300,b,0,0);
```

```
 bar3d(l+300,t+30,r+300,b+30,0,0);
 bar3d(l+300,t+60,r+300,b+60,0,0);
 bar3d(l+300,t+90,r+300,b+90,0,0);
 bar3d(l+300,t+120,r+300,b+120,0,0);
 bar3d(l+300,t+150,r+300,b+150,0,0);
 bar3d(l+300,t+180,r+300,b+150,0,0);
 bar3d(l+360,t,r+360,b,0,0);
 bar3d(l+360,t+30,r+360,b+30,0,0);
 bar3d(l+360,t+60,r+360,b+60,0,0);
 bar3d(l+360,t+90,r+360,b+90,0,0);
 bar3d(l+360,t+120,r+360,b+120,0,0);
 bar3d(l+360,t+150,r+360,b+150,0,0);
 bar3d(l+360,t+180,r+360,b+150,0,0);
 settextstyle(1,0,2);
 setcolor(GREEN);
 for(g=0;g<7;g++)
 {
   strcpy(st1,day[g]);
   outtextxy(x,y,st1);
   x+=60;
 }
}
void calendar::chose()
{
int width=0,i,count,xx,yy;
char st[5];
char
```

```
*month[]={"JANUARY","FEBRUARY","MARCH","APRIL","MAY","JUNE","JULY","AUGUST
","SEPTEMBER","OCTOBER","NOVEMBER","DECEMBER"};
struct REGPACK reg;

rectangle(0,0,getmaxx(),getmaxy());
count=sizeof(month)/sizeof(char *);
setcolor(BROWN);
settextstyle(1,0,1);
rectangle(40,90,225,125);
setcolor(CYAN);
outtextxy(45,95,month[mon-1]);
setcolor(BROWN);
rectangle(180,95,220,120);
settextstyle(1,1,1);
setcolor(CYAN);
outtextxy(185,100,"<");

itoa(year,st,10);
settextstyle(1,0,1);
rectangle(250,85,380,130);
setcolor(BROWN);
outtextxy(255,95,st);

setcolor(CYAN);
rectangle(340,90,375,105);
setcolor(BROWN);
settextstyle(1,1,1);
```

```
outtextxy(345,91,">");


setcolor(CYAN);

rectangle(340,110,375,125);

setcolor(BROWN);

settextstyle(1,1,1);

outtextxy(345,111,"<");


xx=getmaxx();

yy=30;

setfillstyle(1,3);

rectangle(xx-30,yy-25,xx-4,yy);

settextstyle(0,0,2);

outtextxy(xx-24,yy-20,"x");

reg.r_ax=1;

intr(0x33,®);

t.time();

while(!kbhit())

{

reg.r_ax=3;

intr(0x33,®);

if(reg.r_bx==1)

{

while(reg.r_bx==1)

{

reg.r_ax=3;

intr(0x33,®);
```

```
}
 if( reg.r_cx<=220 && reg.r_cx>=180 && reg.r_dx<=120 && reg.r_dx>=95 )
 {
  settextstyle(3,0,3);
  displaymenu(month,count,45,130);
  for(i=0;i<count;i++)
  {
  if(textwidth(month[i])>width)
  width=textwidth(month[i]);
  }
  while(mon!=13)
  {
  mon=getresponse(month,width,count,45,130);
  tday();
  cal();
  reqmon();


  }
  }
  if(reg.r_cx>=340 && reg.r_cx<=375 && reg.r_dx>=90 && reg.r_dx<=105)
  {
  if(year<2060)
  year++;
  itoa(year,st,10);
  setfillstyle(SOLID_FILL, BLACK);
  bar(251,86,320,129);
  settextstyle(1,0,1);
 outtextxy(255,95,st);
 tday();
 cal();
 reqmon();
 }

if(reg.r_cx>=340 && reg.r_cx<=375 && reg.r_dx>=110 && reg.r_dx<=125)
{
 if(year>0)
 year--;
 itoa(year,st,10);
 setfillstyle(SOLID_FILL, BLACK);
 bar(251,86,320,129);
 settextstyle(1,0,1);
 outtextxy(255,95,st);
 tday();
 cal();
 reqmon();
 }
if(reg.r_cx>=610 && reg.r_cx<=635 && reg.r_dx>=5 && reg.r_dx<=30)
 {
		reg.r_ax=2;
		intr(0x33,®);
		cleardevice();

		setbkcolor(11);
```

```
            setcolor(BROWN);
            settextstyle(1,0,5);
            outtextxy(100,200,"EXITING");
            int o=0;
            for(int n=0;n<6;n++)
            {
              outtextxy(350+o,200,".." );
              o+=20;
              delay(200);
            }
            exit(0);
          }

    }
   }

  }

calendar::displaymenu(char **month,int count,int x1,int y1)
{
int i,h;
setfillstyle(SOLID_FILL,BLACK);
bar(x1-4,y1-4,225,getmaxy()) ;
setcolor(BLUE);
h=textheight(month[0]);
for(i=0;i<count;i++)
{
```

```
outtextxy(x1,y1+i*(h+5),month[i]);
delay(10);
}
return 0;
}
calendar::getresponse(char **month,int width,int count,int x1,int y1)
{
int choice=1,premon=0,x2,y2;
int i,h;
struct REGPACK reg;
h=textheight(month[0]);
y2=y1+count*(h+5);
x2=x1+width;
//setcolor(15);

setcolor(BROWN);
rectangle(x1-5,y1-5,x2+57,y2+1);

reg.r_ax=1;
intr(0x33,®);
while(!kbhit())
{
reg.r_ax=3;
intr(0x33,®);
if( reg.r_cx>=x1 && reg.r_cx<=x2 && reg.r_dx>=y1 && reg.r_dx<=y2 )
{
 for(i=1;i<=count;i++)
```

```
{
 if(reg.r_dx<=y1+i*(h+5))
 {
 choice=i;
 break;
 }
}
 if(premon!=choice)
 {
 reg.r_ax=2;
 intr(0x33,®);
 highlight(month,choice,h,x1,y1,width);
 if(premon)
  dehighlight(month,premon,h,x1,y1,width);
 premon=choice;
 }
 reg.r_ax=1;
 intr(0x33,®);
 if(reg.r_bx==1)
 {
 while(reg.r_bx==1)
 {
  reg.r_ax=3;
  intr(0x33,®);
  }
  if( reg.r_cx>=x1 && reg.r_cx<=x2 && reg.r_dx>=y1 && reg.r_dx<=y2 )
  mon=choice;
  return(mon);
  }
 }
 else if(reg.r_cx>=40 && reg.r_cx<=225 && reg.r_dx>=90 &&
reg.r_dx<=125)
 {
 reg.r_ax=1;
 intr(0x33,®);
 if(reg.r_bx==1)
 {
 while(reg.r_bx==1)
 {
  reg.r_ax=3;
  intr(0x33,®);
  }
  if( reg.r_cx<=220 && reg.r_cx>=180 && reg.r_dx<=120 && reg.r_dx>=95
)
 {
 setfillstyle(SOLID_FILL, BLACK);
 bar(40,139,226,469);
 return(mon);
 }
 }
 }
 }
   return 0;
 }
 calendar::highlight(char **month,int ch,int h,int x1,int y1,int width)
 {
  setfillstyle(SOLID_FILL,RED);
  bar(x1,y1+(ch-1)*(h+5),x1+width,y1+ch*(h+5));
  setcolor(0);
  settextstyle(TRIPLEX_FONT,0,3);
  outtextxy(x1,y1+(ch-1)*(h+5),month[ch-1]);
  return 0;
 }
calendar::dehighlight(char **month,int ch,int h,int x1,int y1,int
width)
 {
  setfillstyle(SOLID_FILL,BLACK);
  bar(x1,y1+(ch-1)*(h+5),x1+width,y1+ch*(h+5));
  setcolor(1);
  settextstyle(TRIPLEX_FONT,0,3);
  outtextxy(x1,y1+(ch-1)*(h+5),month[ch-1]);
  return 0;
 }


//main class


class windoo
{
calendar c;
```

```
public:

void menu();

};
void windoo::menu()
{
  c.tday();
  c.cal();
  c.reqmon();
  c.chose();

}

int calendar::s=0;
void main()
{
clrscr();
int gd=DETECT,gm;
struct REGPACK reg;
initgraph(&gd,&gm,"c:\tc\bgi ");

cleardevice();
setcolor(BLUE) ;
rectangle(20,20,580,450);
setcolor(18);
```

```
settextstyle(1,0,1);

outtextxy(60,40,"*********************************************************

********");

outtextxy(60,420,"********************************************************

*********");

setcolor(18);

settextstyle(1,1,1);

outtextxy(50,5,"********************************************************");

outtextxy(500,10,"*******************************************************")

;



getch();


reg.r_ax=0;

intr(0x33,®);

reg.r_ax=1;

intr(0x33,®);

reg.r_ax=2;

intr(0x33,®);

windoo w;

w.menu();

getch();


}
```

C++ > Computer Graphics sample source codes

Progress bar

```
#include<graphics.h>

#include<conio.h>

#include<alloc.h>

#include<dos.h>

void main()

{

int gd=DETECT,gm;

initgraph(&gd,&gm,"c:\tc ");   //put your directory where egavga.bgi

is

void   *buffer;

unsigned int size;

setbkcolor(BLUE);

line(230,330,370,330);

line(230,350,370,350);


line(226,335,226,345);


line(226,335,230,330);

line(226,345,230,350);


line(374,335,374,345);

line(374,335,370,330);
line(374,345,370,350);

outtextxy(275,365,"Loading");  //put you text here

int x=232,y=336,x1=236,y1=344;

for(int i=1;i<5;i++)

{

setfillstyle(1,RED);

bar(x,y,x1,y1);

x=x1+2;

x1=x1+6;

}

size=imagesize(232,336,256,344);

buffer=malloc(size);

getimage(232,336,256,344,buffer);

x=232;

int m=0;

while(!kbhit())

{

putimage(x,336,buffer,XOR_PUT);

x=x+2;

if(x>=350)

{

m++;

x=232;

if(m==5)                        // m is no of times bar moves

return;

}

putimage(x,336,buffer,XOR_PUT);
delay(20);                  // delay(time) is the speed of moving
bar
                            // less delay means fast and vice versa
}
getch();
}
```

C++ > Computer Graphics sample source codes

Quick Sort Program with Text Graphics

```
        #include <iostream.h>
        #include <stdio.h>
        #include <dos.h>
        #include <conio.h>

        #define   MAX   15

        #define ValueOf( x )      ( x.value() )
        #define Exchange( x , y )         ( x.exchange(y) )

        class element
        {
                int _value;
```

```
		int _color;
	public:
		element() { _color = 15; }
		void get()
		{
			scanf ( "%d", &_value );
		}
		int value (){	return _value;}
		void exchange ( element &e )
		{
			element temp;
			temp = e;
			e = *this;
			*this = temp;
		}

		void setcolor ( int col ) { _color = col; }

		void show()
		{
			textbackground ( _color );
			if ( _color == 15 )
			{
				textcolor ( 0 );
			}
			else
			{
```

```
                                        textcolor ( 15 );
                                }
                                cprintf ( " %d " , _value );
                                printf ( " " );
                        }
                };
                /*
                int element :: value ()
                {
                        return _value;
                }
                  */
                void QuickSort ( element * , int , int );
                int partition ( element * , int  , int  );

                void Display  ( element *A , int p , int r );
                void main()
                {
                        element array[MAX];
                        int i = 1;

                        textbackground ( 0 );
                        textcolor ( 15 );
                        clrscr ();
                        printf ( "
        Enter %d elements:-
        > ", MAX - 1 );
```

```
                for ( i = 1; i < MAX; i++ )
                        array[i].get();

                printf ( "
" );
                for ( i = 1; i < MAX; i++ )
                        array[i].show();

                getch();
                printf ( "
" );
                QuickSort ( array , 1 , MAX - 1 );

                printf ( "
 " );
                for ( i = 1; i < MAX; i++ )
                        array[i].show();

                getch();
        }

        void QuickSort ( element *A , int p , int r )
        {
                int q;
                if ( p < r )
                {
                        q = partition ( A , p , r );
```

```
                    QuickSort ( A , 1 , q - 1 );

                    QuickSort ( A , q + 1 , r );

          }

}


int partition ( element *A , int p , int r )

{

          int key , i = 1 , j = 1;


          key = ValueOf ( A[r] );

          A[r].setcolor ( RED );


          i = p - 1;

          for ( j = p ; j <= r; j++ )

          {

                    if ( ValueOf ( A[j] ) <= key )

                    {

                              i = i + 1;

                              Exchange ( A[j] , A[i] );

                    }

                    else

                    {

//                            A[j].setcolor ( BLUE );

                    }

                    Display ( A  , 1 , MAX );

                    delay ( 100 );

          }
```

```
                    A[i].setcolor ( GREEN );

                    Display ( A  , 1 , MAX );

                    printf ( "

> %d at correct position. ", ValueOf ( A[i] ) );

                    return i ;

          }


          void Display  ( element *A , int p , int r )

          {

                    if ( wherey () > 23 )

                    {

                              getch();

                              textbackground ( 0 );

                              textcolor ( 15 );

                              clrscr();

                    }

                    printf ( "



" );

                    for ( int i = p; i < r; i++ )

                    {

                              A[i].show();

                    }

          }
```

C++ > Computer Graphics sample source codes

logical discription of logial errors in graphics

```
#include<iostream.h>

#include<conio.h>

#include<dos.h>

#include<stdio.h>

#include<graphics.h>

#include<math.h>

#include<string.h>

#include<time.h>

float main(void)

 {

  int*p1,*p2,*p;

  clock_t start,end;

  time_t t,t1;

  int z=0,z1=0;

  int gdriver = DETECT, gmode, errorcode;

  initgraph(&gdriver, &gmode, "");

  int r=0;

  int ch,x=10,y=350;

  int poly[100],poly1[100],variable1;

  setcolor(14);

  ellipse(100,105,180,0,10,15);

  ellipse(93,125,320,50,3,7);

  ellipse(105,125,130,270,3,7);

  ellipse(110,112,250,90,3,2);

  line(93,128,108,133);
```

```
putpixel(100,120,4);

poly[0]=105;

poly[1]=105;

poly[2]=103;

poly[3]=108;

poly[4]=101;

poly[5]=109;

poly[6]=101;

poly[7]=111;

poly[8]=100;

poly[9]=108;

poly[10]=98;

poly[11]=108;

poly[12]=96;

poly[13]=111;

poly[14]=96;

poly[15]=119;

poly[16]=88;

poly[17]=113;

poly[18]=89;

poly[19]=105;

poly[20]=105;

poly[21]=105;

setcolor(8);

drawpoly(11,poly);

setfillstyle(1,8);

floodfill(94,108,8);
```

```
setcolor(14);

poly1[0]=105;

poly1[1]=105;

poly1[2]=103;

poly1[3]=108;

poly1[4]=101;

poly1[5]=109;

poly1[6]=101;

poly1[7]=111;

poly1[8]=100;

poly1[9]=108;

poly1[10]=98;

poly1[11]=108;

poly1[12]=96;

poly1[13]=111;

poly1[14]=96;

poly1[15]=119;

drawpoly(8,poly1);

line(105,105,110,105);

setfillstyle(1,14);

floodfill(106,110,14);

floodfill(111,112,14);

setcolor(2);

for(variable1=0;variable1<=5;variable1+=2)

ellipse(100,101,0,180,10,variable1);

setcolor(4);

for(variable1=0;variable1<=5;variable1+=3)
```

```
ellipse(100,105,0,180,20,variable1);

line(80,105,120,105);

setfillstyle(1,0);

fillellipse(107,111,1,2);

setcolor(0);

arc(107,111,70,160,3);

setcolor(6);

setfillstyle(1,6);

fillellipse(99,112,1,3);

setfillstyle(1,4);

fillellipse(99,115,2,2);

setcolor(8);

for(variable1=0;variable1<=3;variable1++)

ellipse(107,118,70,180,4,variable1);

setcolor(4);

line(93,128,108,133);

line(108,133,110,138);

line(110,138,93,133);

line(93,133,93,128);

setfillstyle(2,4);

floodfill(96,131,4);

setcolor(2);

ellipse(100,193,20,70,15,60);

ellipse(101,184,120,170,15,60);

line(93,133,105,137);

line(86,174,114,173);

setfillstyle(1,2);
```

```
floodfill(90,170,2);

setcolor(3);

line(104,140,102,150);

line(94,140,94,151);

line(94,140,103,140);

line(100,150,108,165);

line(100,165,108,165);

line(94,151,100,165);

setcolor(14);

line(102,166,107,166);

line(103,168,109,168);

line(102,166,103,168);

line(107,166,109,168);

setfillstyle(1,14);

floodfill(105,167,14);

setcolor(0);

line(102,169,110,169);

setfillstyle(9,2);

floodfill(100,145,3);

setcolor(12);

getimage(78,95,122,202,p1);

line(108,175,106,195);

line(92,175,94,195);

line(108,175,92,175);

line(106,195,94,195);

setfillstyle(6,12);

floodfill(100,180,12);
```

```
setcolor(8);

setfillstyle(6,8);

ellipse(103,200,0,180,10,3);

line(93,200,113,200);

floodfill(103,199,8);

getimage(78,95,122,202,p2);

putimage(78,95,p2,1);

putimage(78,95,p1,1);

setcolor(12);

setfillstyle(6,12);

int po[100],pol[100];

po[0]=110;

po[1]=174;

po[2]=120;

po[3]=196;

po[4]=108;

po[5]=196;

po[6]=96;

po[7]=174;

po[8]=110;

po[9]=174;

drawpoly(5,po);

pol[0]=96;

pol[1]=174;

pol[2]=89;

pol[3]=174;

pol[4]=87;
```

```
pol[5]=196;

pol[6]=97;

pol[7]=196;

pol[8]=101;

pol[9]=184;

drawpoly(5,pol);

floodfill(103,177,12);

floodfill(93,177,12);

setcolor(8);

setfillstyle(6,8);

ellipse(119,200,0,180,10,3);

ellipse(97,200,0,180,10,3);

line(109,200,129,200);

line(107,200,87,200);

floodfill(119,199,8);

floodfill(97,199,8);

getimage(78,95,130,202,p1);

putimage(78,95,p1,1);

putimage(x-5,350,p1,1);

setcolor(15);

rectangle(0,458,getmaxx()+10,getmaxy());

setfillstyle(6,15);

floodfill(10,464,15);

setfillstyle(6,15);

fillellipse(400,100,60,40);

int v=0,v1=0,v2[1000],l1,l2,l3;

int w1=0,w2,w3=20,w4;
```

```
here1:

delay(90);

setcolor(0);

setfillstyle(1,10);

w2=sqrt(abs(w3*w3-w1*w1));

fillellipse(w1,w2,w3,w3);





w1++;

if(w1==getmaxx()+12)

w1=0;

while(kbhit())

ch=getch();

if(ch== 77)

  {

  x=x+5;

  //sound(3000);

  }

else if (ch== 72)

  {

  v=350;

  for(int i=y,j=y-200,k=y+107;i>=y-200;i-=4,j+=4,k--)

        {

        if(i>=250)

          {

             if(x%2==0)
```

```
{
 //sound(i+100);
  if(w1==getmaxx()+12)
  w1=0;
  fillellipse(w1,w2,w3,w3);
  w1++;
 delay(15);
 if(i==y)
 putimage(x,i+4,p1,1);
 putimage(x,i+4,p1,1);
 putimage(x,i,p1,2);
 if(!kbhit())
 z = 10;
 while(kbhit())
          {
              z+=2;
              if (z>100)break;
              v2[i]=getch();
              delay(15);
              switch(v2[i])
                {
                         case 77:
                           x=x+10;
                           putimage(x,i,p1,1);
                           putimage(x-10,i,p1,1);
                           break;
                         case 75:
```

```
  x=x-10;

  putimage(x,i,p1,1);

  putimage(x+10,i,p1,1);

  break;

case 32:

  l2=i;

  for(l1=x+50;l1<=getmaxx();l1++)

  {

  while(kbhit())

   {

        int t = getch();

        switch(t)

         {

          case 77:

          x=x+10;

          putimage(x,i,p1,1);

          putimage(x-10,i,p1,1);

          break;

          case 75:

          x=x-10;

          putimage(x,i,p1,1);

          putimage(x+10,i,p1,1);

          break;

         }

   }

  delay(3);

  setcolor(0);
```

```
                              setfillstyle(1,4);
                              fillellipse(l1-2,l2,5,5);
                              if(!kbhit())ungetch(t);
                             }
                            default:
                             break;
                            case 27:
                              goto here2;
                         }
                        }

                  ungetch(v2[i]);
          }
else
          {
           ////sound(i+100);
           delay(15);
           if(i==y)
           putimage(x,i+4,p2,1);
           putimage(x,i+4,p2,1);
           putimage(x,i,p2,2);
          if(!kbhit())
           z = 10;
           while(kbhit())
                  {
                     z+=2;
                     if (z>100)break;
```

```
               v2[i]=getch();

               delay(15);

               switch(v2[i])

                 {

                         case 77:

                           x=x+10;

                           putimage(x,i,p2,1);

                           putimage(x-10,i,p2,1);

                           break;

                         case 75:

                           x=x-10;

                           putimage(x,i,p2,1);

                           putimage(x+10,i,p2,1);

                           break;

                         case 32:


                         default:

                          break;

                         case 27:

                           goto here2;

                }

             }

     ungetch(v2[i]);

    }


      }
```

```
                if(i<250)
                      {
                        if(x%2==0)
                              {
                                ////sound(j+100);
                                delay(15);
                                if(j==y)
                                putimage(x,i-4,p,1);
                                putimage(x,j-4,p1,1);
                                putimage(x,j,p1,2);
                               }
                        else
                               {
                                ////sound(i+100);
                                delay(15);
                                if(j==y)
                                putimage(x,i-4,p,1);
                                putimage(x,j-4,p2,1);
                                putimage(x,j,p2,2);
                               }
                      }
             }
  }
else if (ch== 75)
{
  x=x-5;
  ////sound(3000);
```

```
}
else if (ch== 27)
goto here2;
else       goto here;
if(ch==77||ch==75)
   {
   if(x%2==0)
      {
               if(r==1)
                  {
                          if(ch==77)
                          putimage(x-5,y,p2,1);
                          if(ch==75)
                          putimage(x+5,y,p2,1);
                  }
               putimage(x,y,p1,1);
      }
  else
      {
               if(r==1)
                  {
                          if(ch==77)
                          putimage(x-5,y,p1,1);
                          if(ch==75)
                          putimage(x+5,y,p1,1);
                        }
                          if(r==0)
                        putimage(x-10,350,p1,1);

                        putimage(x,y,p2,1);

                }

  r=1;

  }


  here:

  nosound();

  ch=0;

  goto here1;

  here2:

  nosound();

}
```