**COMPUTER GRAPHICS**

**MINI PROJECT–SHOOT\_A\_BALL**

* **G.Saiprasad 114A1012**
* **Sunil P 116A1058**
* **Shweta S 116A1072**

**Aim:**

To build a game using graphics in c language. The aim of the game is to shoot grenades falling from above with a shooter before they fall on the spikes below. The game terminates on missing the grenades 5 times. The score is calculated by no. of hits.

**Theory:**

Some of the functions used in creation of this games are as follows:

**settextstyle:**

Settextstyle function is used to change the way in which text appears, using it we can modify the size of text, change direction of text and change the font of text.

Declaration: void settextstyle( int font, int direction, int charsize);  
font argument specifies the font of text, Direction can be HORIZ\_DIR (Left to right) or VERT\_DIR (Bottom to top).

**Outtextxy:**

Outtextxy function display text or string at a specified point(x,y) on the screen.

Declaration: void outtextxy(int x, int y, char \*string);  
x, y are coordinates of the point and third argument contains the address of string to be displayed.

**Setfillstyle:**

setfillstyle function sets the current fill pattern and fill color.

Declaration: void setfillstyle( int pattern, int color);

**Bar:**

Declaration: void bar(int left, int top, int right, int bottom);

Bar function is used to draw a 2-dimensional, rectangular filled in bar . Coordinates of left top and right bottom corner are required to draw the bar. Left specifies the X-coordinate of top left corner, top specifies the Y-coordinate of top left corner, right specifies the X-coordinate of right bottom corner, bottom specifies the Y-coordinate of right bottom corner. Current fill pattern and fill color is used to fill the bar. To change fill pattern and fill color use [setfillstyle](https://www.programmingsimplified.com/c/graphics.h/setfillstyle" \o "setfillstyle).

**Fillellipse:**

Declaration of fillellipse function :-  
void fillellipse(int x, int y, int xradius, int yradius);  
x and y are coordinates of center of the ellipse, xradius and yradius are x and y radius of ellipse respectively.

**Sound:**

Sound function produces the sound of a specified frequency. Used for adding music to c program, try to use some random values in loop, vary delay and enjoy.

Declaration:- void sound(unsigned frequency);

**Setbkcolor:**

Declaration: void setbkcolor(int color);

setbkcolor function changes current background color e.g. setbkcolor(YELLLOW) changes the current background color to YELLOW.  
Remember that default drawing color is WHITE and background color is BLACK

**kbhit:**

kbhit() is present in conio.h and used to determine if a key has been pressed or not. To use kbhit function in your program you should include the header file “conio.h”. If a key has been pressed then it returns a non zero value otherwise returns zero

**randomize():**

C program to generate pseudo-random numbers using rand and random function (Turbo C compiler only). As the random numbers are generated by an algorithm used in a function they are pseudo-random, this is the reason that word pseudo is used. Function rand() returns a pseudo-random number between 0 and RAND\_MAX. RAND\_MAX is a constant which is platform dependent and equals the maximum value returned by rand function.

**nosound():**

nosound function turn off the PC speaker.

Declaration : void nosound();

The flow of the program is as follows:

In this program we have created the introduction screen in the function called **logo()** where we have used in-built functions like outtextxy(),delay(), sleep() functions accordingly to provide some time lapse between screens and cleardevice() to clear the screen for next display.

We have created the grenades using ellipse() and fillellipse() functions, the shooter using bar() function, and the spikes using line().

In the blast() function we are creating sparks on collision.

In the main() function we have initialized the graphic interface, declared the variables, called the logo() , blast() functions and for background music we have use sound() and nosound() functions accordingly.

We have also used labels such as goto, ep1, ep2, restart to direct the program during its execution.

Thus by using iterations we have displayed the screens at different instances and thus created the game.

**Program:**

#include<graphics.h>

#include<conio.h>

#include<stdlib.h>

void logo()

{

settextstyle(10,0,6);setcolor(9);

outtextxy(50,150,"Game\_shooter");sleep(1);cleardevice();

settextstyle(5,0,8);setcolor(3);

outtextxy(100,100,"CGPROJECT");sleep(2);setcolor(11);cleardevice();

settextstyle(7,0,6);setcolor(6);

outtextxy(90,100,"SHOOT\_A\_BALL");

setfillstyle(2,6);

bar(0,200,600,240);

sleep(2);settextstyle(6,0,5);setcolor(14);

outtextxy(100,200,"Press any key to continue");delay(100);

getch();

cleardevice();

settextstyle(1,0,3);setcolor(14);

outtextxy(250,50,"INSTRUCTIONS");

outtextxy(50,100,"defend the wall against the bombs");

outtextxy(50,130,"otherwise you will fall on the spike");

outtextxy(50,200,"press space \_ to shoot");

outtextxy(50,250,"press left <- or right -> arrow to move");

delay(1000);

setcolor(15);

}

void blast(int s,int cs)

{

int e=5,i;

if(cs==0)

cs=1;

for(i=1;i<6;i++)

{

setfillstyle(1,12);

fillellipse(s\*50+25,cs\*25,e,5);

setfillstyle(i+1,i);

fillellipse(s\*50+25,cs\*25,5,e);

delay(15);

sound(300);

e=e+5;

}

nosound();

}

void main()

{

int gd=DETECT,gm;

int a,m,s,mus,i,hits,c[11],bm,score,mute,dly;

initgraph(&gd,&gm,"c:\\TurboC3\\bgi");

logo();

restart:

clrscr();

cleardevice();

printf("\n\n\n\n\n\n\t\t\t\t\t\t\t\t\t ");

s=1;mus=1;hits=0;bm=2;score=0;mute=1,dly=200;

for(i=0;i<11;i++)

c[i]=0;

randomize();

a=1;

con:

while(!kbhit())

{

a=random(10);

if(c[a]==0)

{

c[a]=1;

}

if(mute%2==0)

mus=random(1000);

else

{

mus=0;

nosound();

}

ep1:

cleardevice();

setbkcolor(1);

setcolor(15);

settextstyle(2,0,5);

outtextxy(575,75,"score: ");

if(score>9)

printf("\b");

if(score>100)

printf("\b");

printf("\b%d",score);

setfillstyle(11,9);

bar(50,15,550,480);

setfillstyle(5,2);

bar(s\*50+20,360,s\*50+30,400);

setcolor(15);

rectangle(s\*50+20,360,s\*50+30,400);

rectangle(50,15,550,480);

setcolor(4);

if(mus!=0)

sound(mus);

for(i=50;i<550;i=i+20)

{

line(i,480,i+10,440);

line(i+10,440,i+20,480);

}

setcolor(15);

if(bm!=0)

{

setfillstyle(2,bm);

bar(50,400,550,420);

rectangle(50,400,550,420);

}

line(50,0,50,480);

line(550,0,550,480);

for(i=1;i<11;i++)

{

if(c[i]!=0)

{

setfillstyle(1,8);

bar(1\*50+15,c[i]\*25,i\*50+35,c[i]\*25+5);

fillellipse(i\*50+25,c[i]\*25-3,3,2);

fillellipse(i\*50+35,c[i]\*25-7,8,2);

fillellipse(i\*50+25,c[i]\*25+25,20,20);

if(bm!=0)

c[i]++;

}

if(c[i]==16)

{

blast(i,c[i]);

c[i]=0;

bm++;

}

}

if(bm==0)

{

delay(500);

goto ep2;

}

delay(dly);

if(mus!=0)

sound(mus+100);

if(dly==0)

dly=dly+2;

delay(dly+300);

if(mus!=0)

sound(mus);

dly=dly-2;

if(bm==7)

{

bm=0;

goto ep1;

ep2:

cleardevice();

nosound();

settextstyle(1,0,8);setcolor(3);

outtextxy(100,220,"GAME OVER");

setcolor(15);

printf("\n\t\t\t\Score is :%d",score);

printf("\n\t\t\t\Shots missed :%d",hits-score);

printf("\n\t\t\t\Total hits :%d",hits);

sleep(2);

scanb:

settextstyle(1,0,2);

outtextxy(160,300,"Press HOME to restart");

outtextxy(160,400," OR ");

outtextxy(160,420,"press Esc to exit");

m=getch();

if(m==27)

exit(0);

if(m==71)

goto restart;

else

goto scanb;

}

}

m=getch();

if(m==0)

m=getch();

if(m==77)

{

if(s<10)

s++;

}

if(m==75)

{

if(s>1)

s--;

}

if(m==32)

{

setcolor(4);

line(s\*50+20,360,s\*50+10,340);

line(s\*50+30,360,s\*50+40,340);

line(s\*50+25,360,s\*50+25,340);

blast(s,c[s]);

hits++;

if(c[s]!=0)

score++;

c[s]=0;

}

if(m==9)

getch();

if(m==13)

mute++;

if(m==27)

{

nosound();

exit(0);

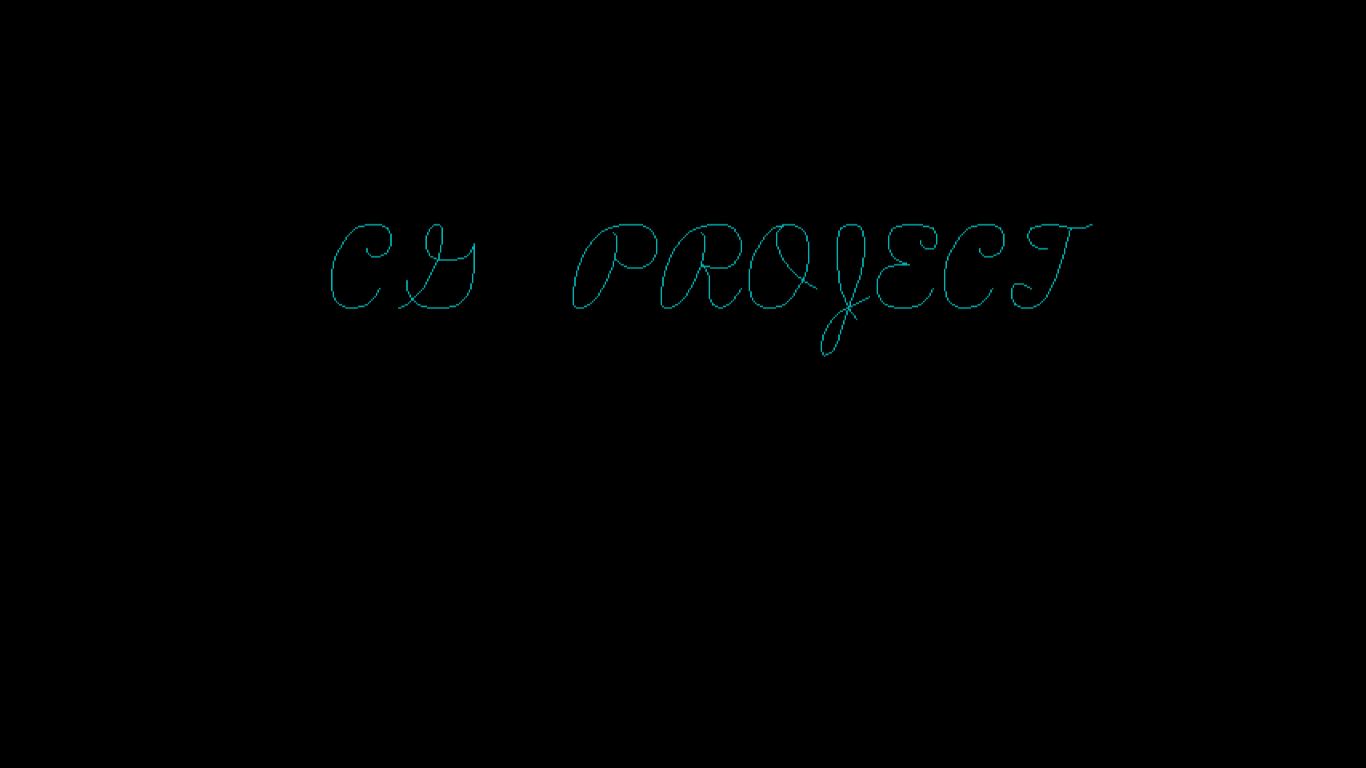
}

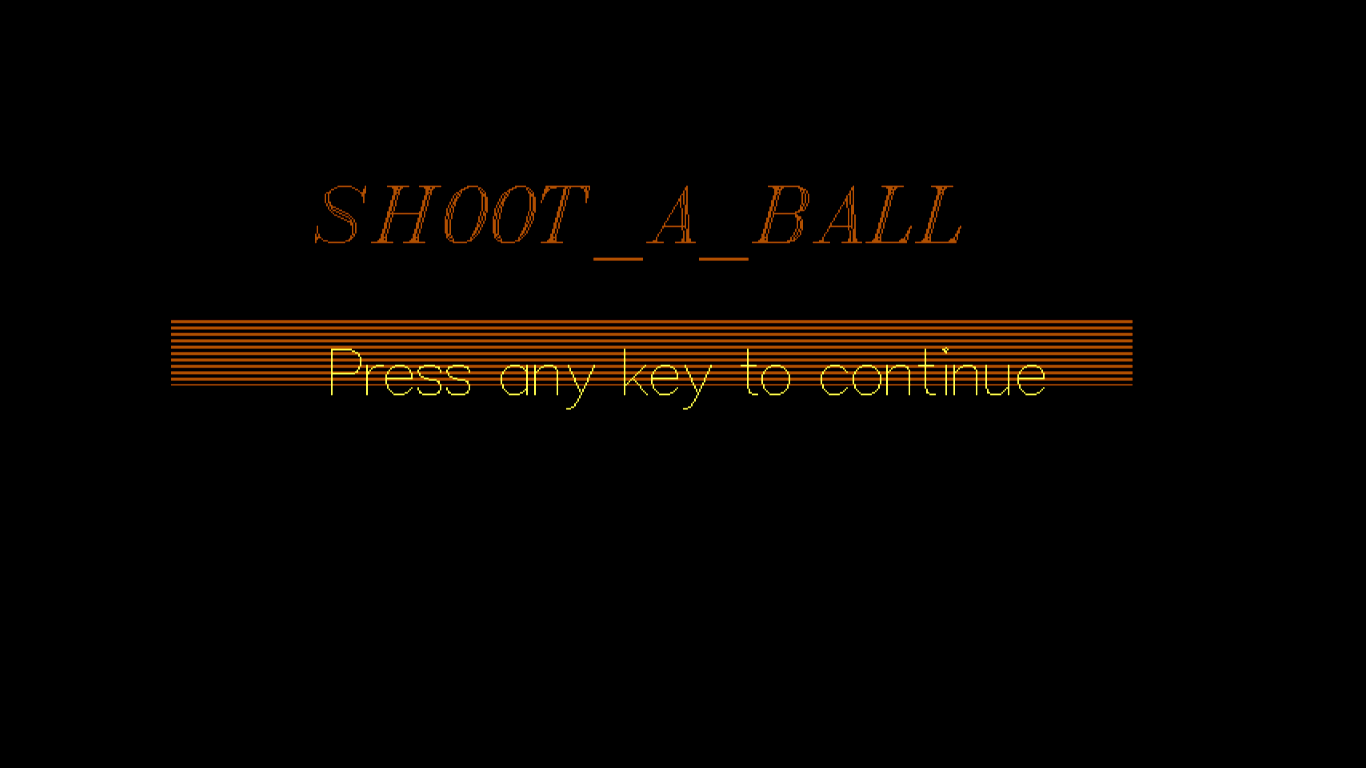
goto con;

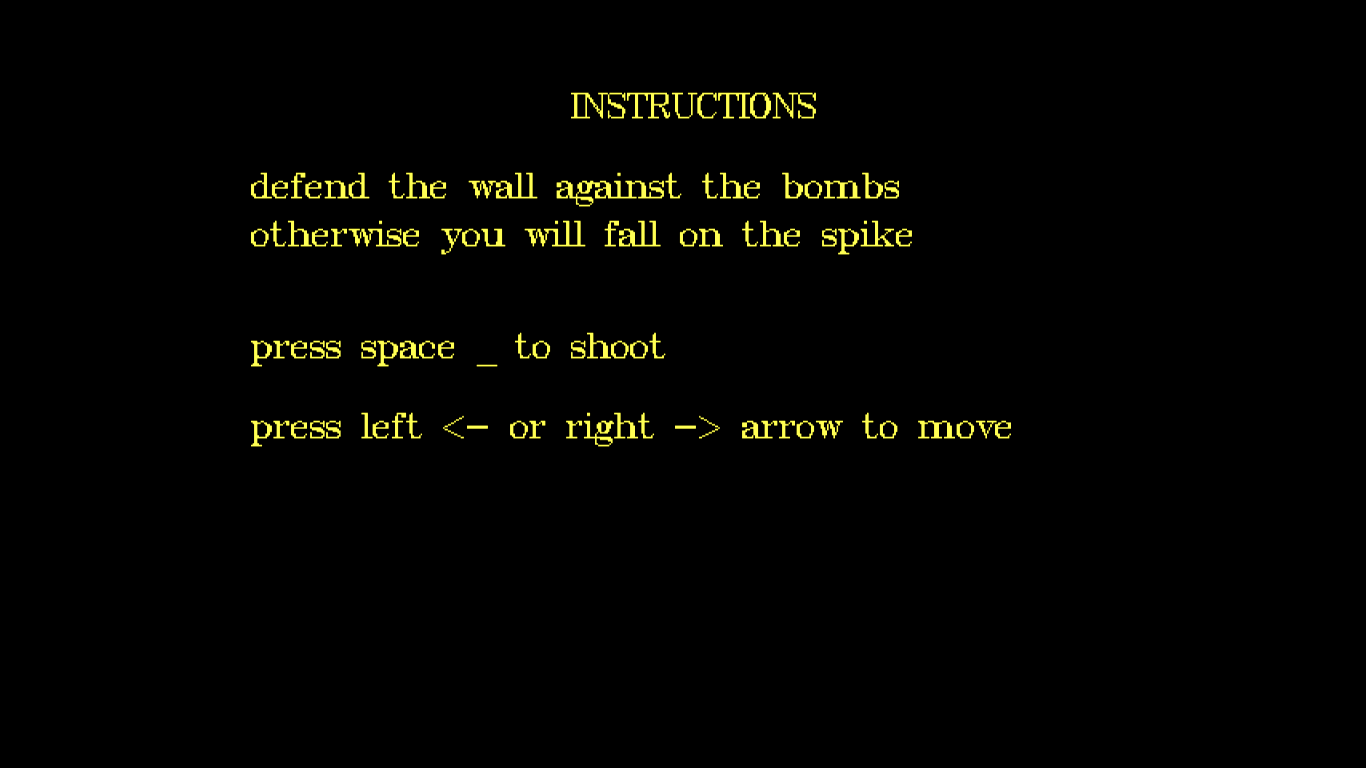
}

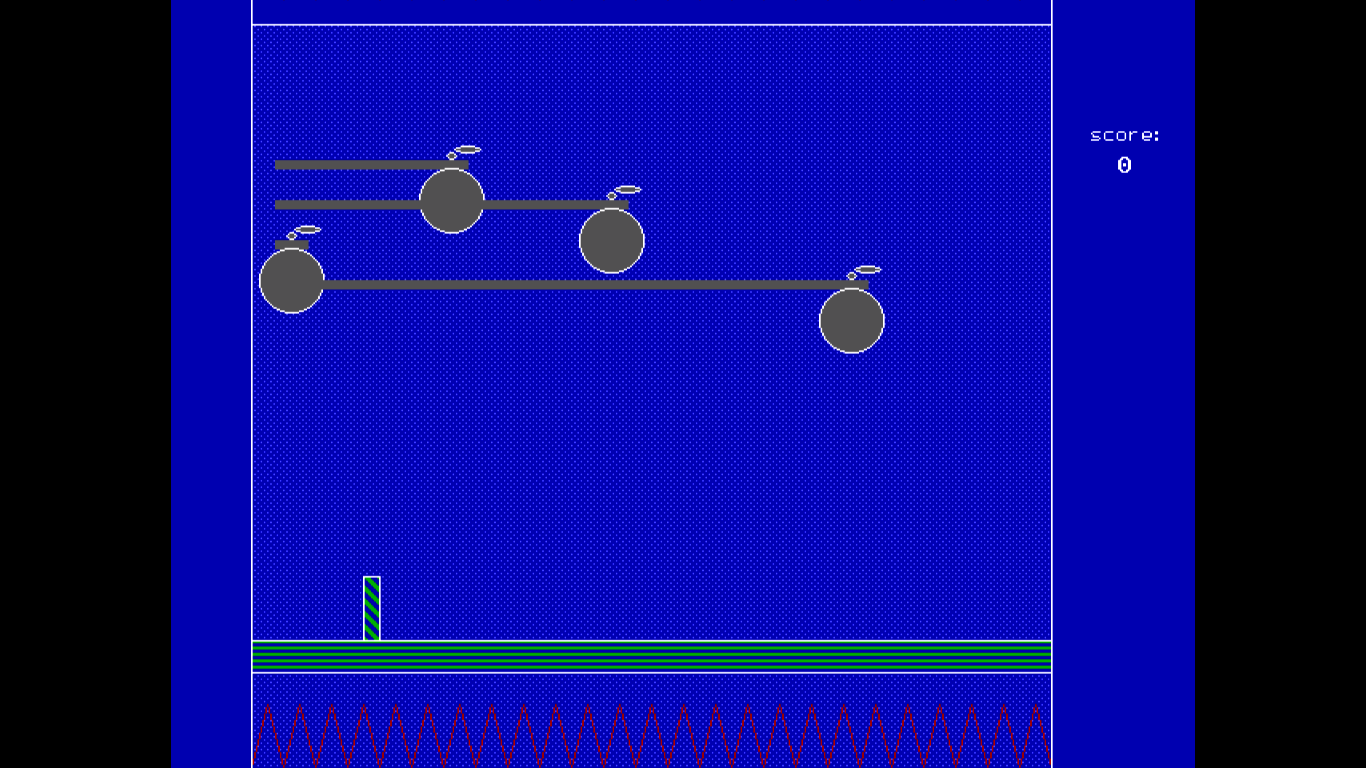
**Game Snapshot:**

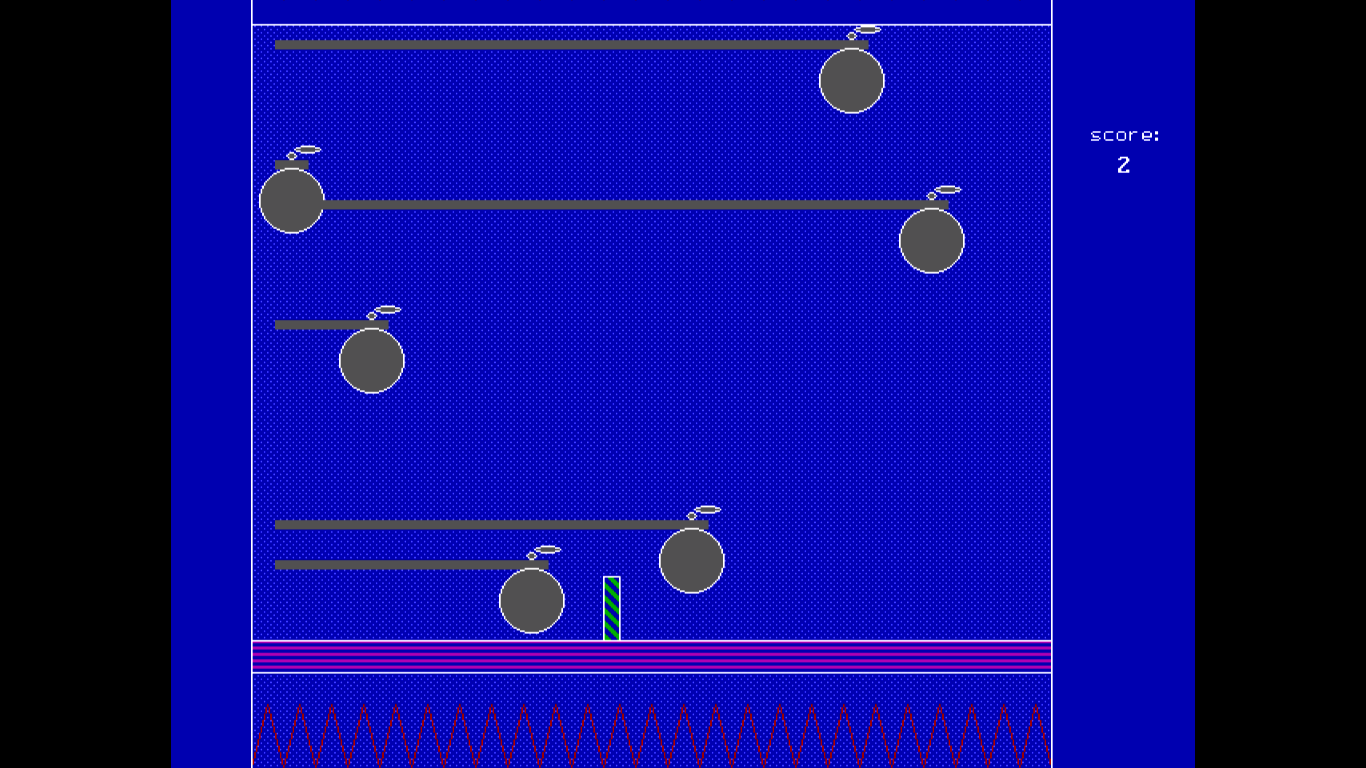


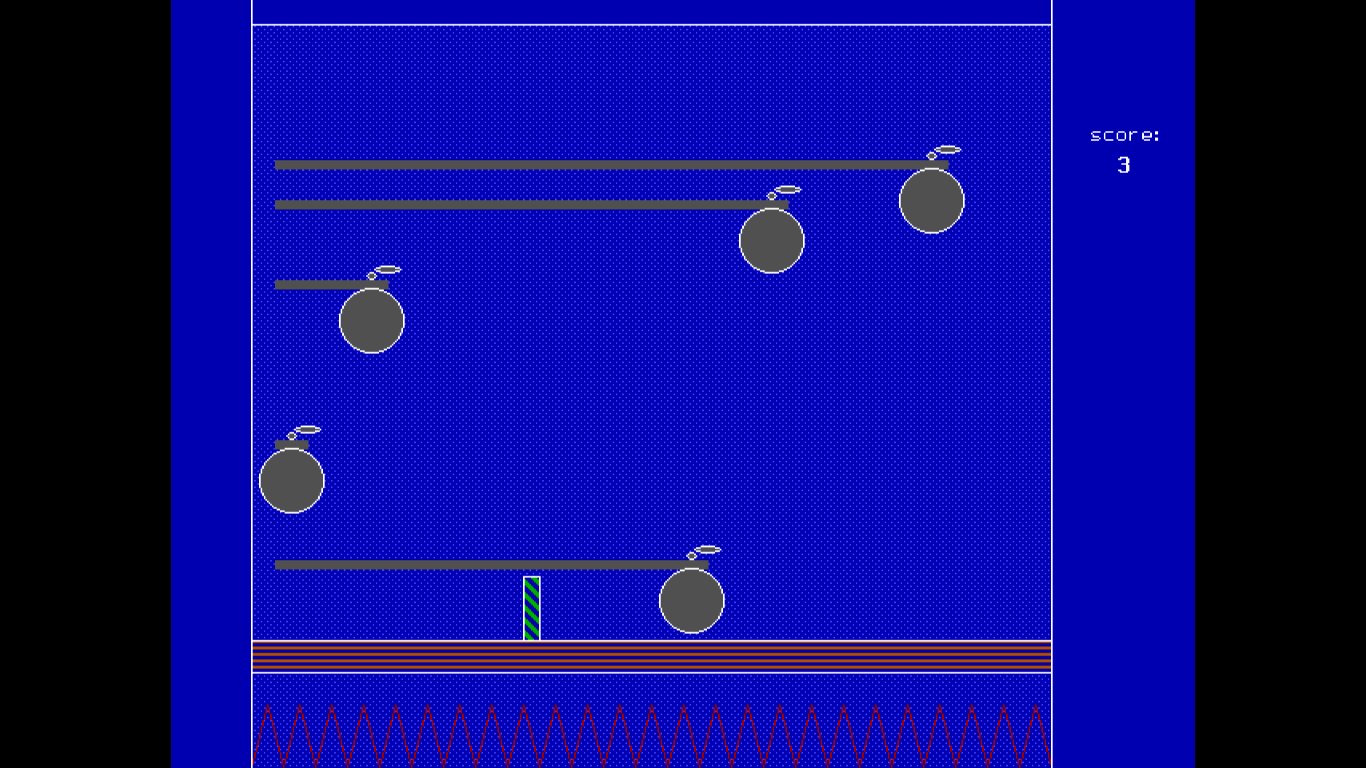


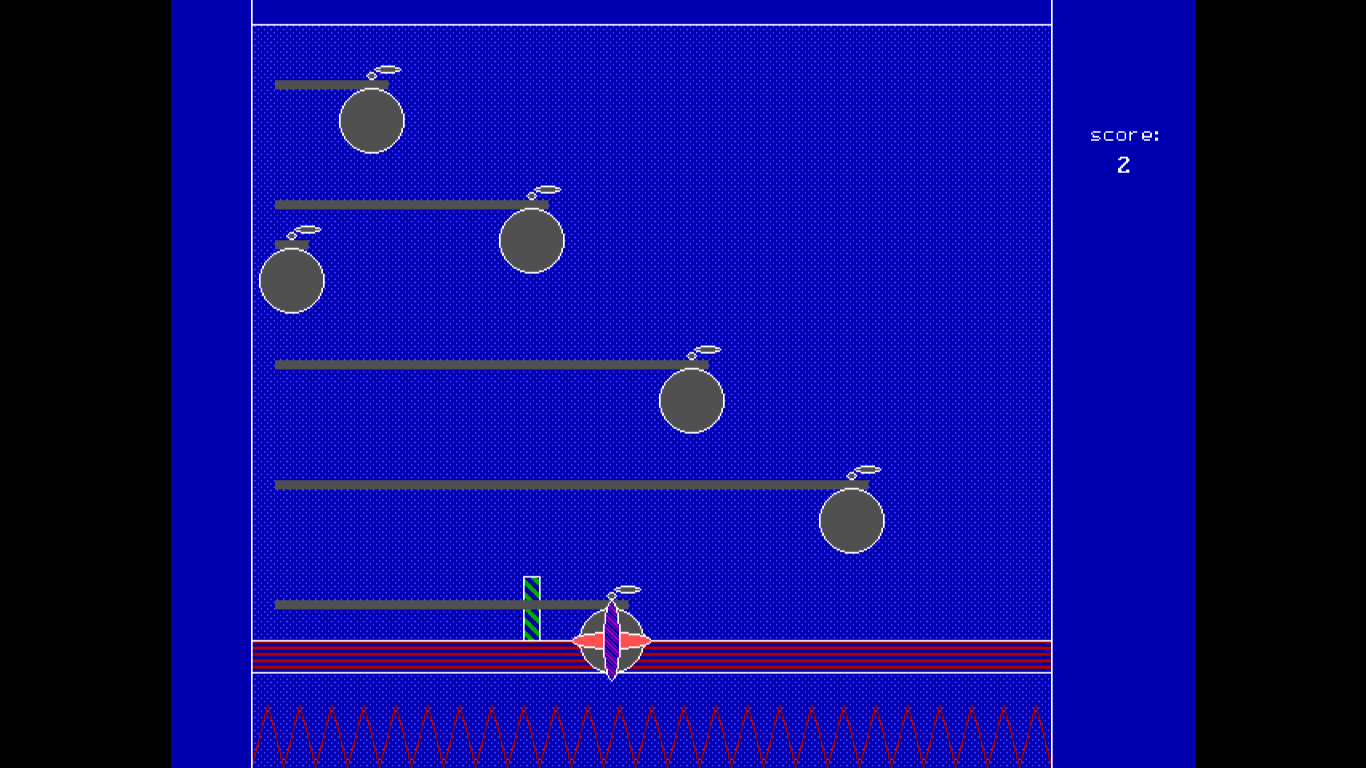


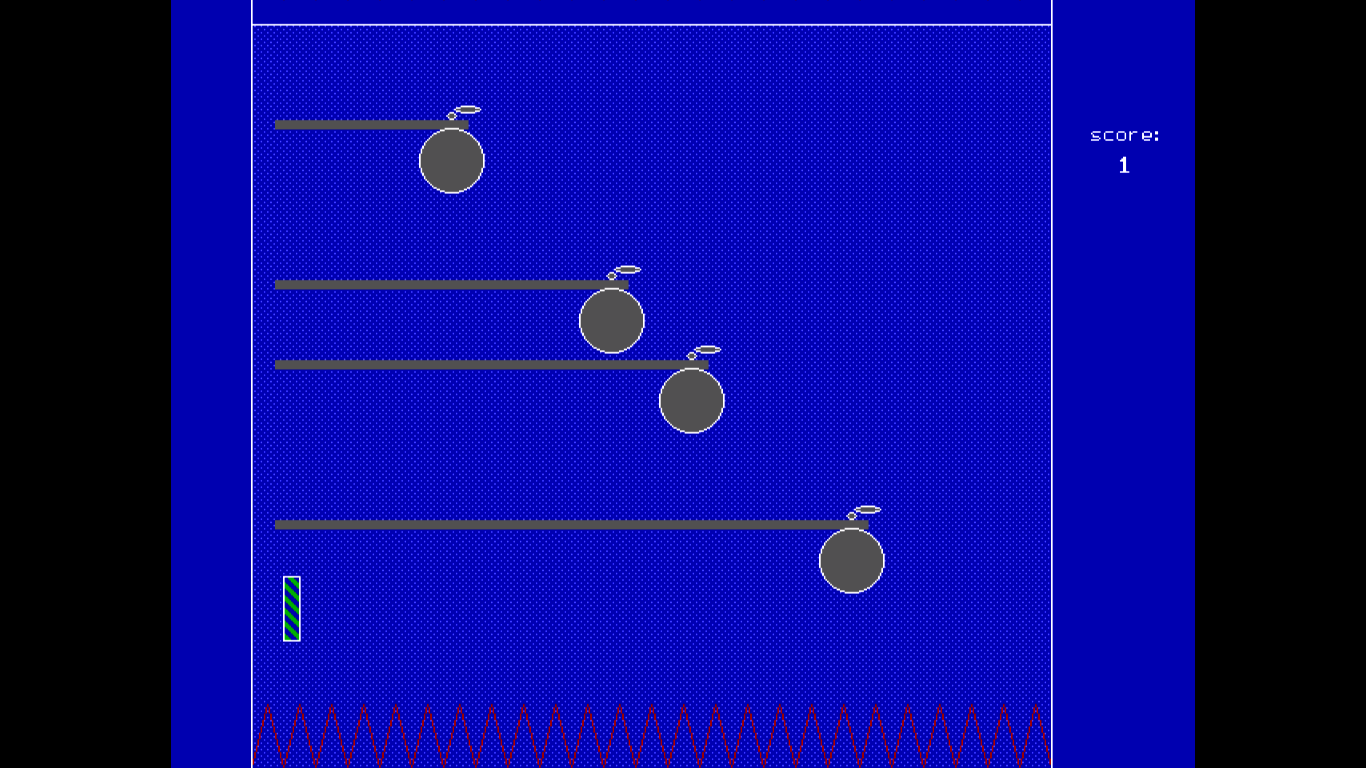














**Conclusion:**

Thus we have successfully created a game using graphics in c and studied the various in-built functions part of graphics.h. It gave us an insight into game creation at a very basic level and understanding the various dynamics involved in game creation. This would enable us to build better and enhanced games in the future.