THE BRICK GAME

HEMANSHU SONDHI
ROHAN VERMA

```
|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|_
___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|
_|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|_
___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|
_|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|_
___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|
_|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|_
             _____     _____    _______    _____      __  __   _________
            /\  _`\  /\  _`\ /\__  _\ /\  _`\   /\ \/\ \ /\______  \
            \ \ \L\ \\ \ \L\ \/_/\ \/ \ \ \/\_\\ \ \/'/'\/_____//'/'
             \ \  _ <'\ \ ,  /  \ \ \  \ \ \/_/_\ \ , <       //'/'
              \ \ \L\ \\ \ \\ \  \_\ \__\ \ \L\ \\ \ \\`\    //'/'___
               \ \____/ \ \_\ \_\/\_____\\ \____/ \ \_\ \_\ /\_______\
                  \/___/   \/_/\/ /\/_____/ \/___/   \/_/\/_/ \/_______/

___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|
_|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|_
___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|
_|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|_
___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|___|
       ______       _                    _     _
      (______ \    | |                  (_)   (_)
       ______) )___ | |__  _____ _____   _     _ _____  ____ _____  _____
      |  ___  // _ \|  _ \(_____ |  _ \  | |   | | ____ |/ ___)    \(_____ |
      | |   \ \ |_| | | | / ____ | | | |  \ \ / /| _____| |   | | | / ____ |
      |_|    |_\___/|_| |_\______|_| |_|   \___/ |______)_|   |_|_|_\______|

             _     _                                  _
            (_)   (_)                                | |
             _______ _____ _____   _____ _____   ____| |__  _   _
            |  ___  | ____ |     \(_____ |  _ \ /____)  _ \| | | |
            | |   | | _____| | | / ____ | | | |____ | | | | |_| |
            |_|   |_|______)_|_|_\______|_| |_(____/|_| |_|____/

                  ______                  __ _     _
                 / ______)               | | |   (_)
                ( (_____   ____  _____   __| | |__  _
                 \_____ \ / _ \|  _ \ / _  |  _ \| |
                 ______) ) |_| | | | ( (_| | | | | |
                (_______/ \___/|_| |_|\____|_| |_|_|


 #_                                                                       d
 ##_                                                                     d#
 NN#p                                                                  j0NN
 40NNh_                                                              _gN#B0
 4JF@NNp_                                                          _g0WNNL@
 JLE5@WRNp_                                                      _g@NNNF3_L
 _F`@q4WBN@Np_                                                _gNN@ZL#p"Fj_
 "0^#-LJ_9"NNNMp__                                         _gN#@#"R_#g@q^9"
 a0,3_j_j_9FN@N@0NMp__                                 __ggNZNrNM"P_f_f_E,0a
  j  L 6 9""Q"#^q@NDNNNMpg_____                _____gggNNW#W4p^p@jF"P"]"j  F
 rNrr4r*pr4r@grNr@q@Ng@q@N0@N#@NNMpmggggmqgNN@NN@#@4p*@M@p4qp@w@m@Mq@r#rq@r
   F Jp 9__b__M,Juw*w*^#^9#""EED*dP_@EZ@^E@*#EjP"5M"gM@p*Ww&,jL_J__f  F j
 -r#^^0""E" 6  q  q__hg-@4""*,_Z*q_"^pwr""p*C__@""0N-qdL_p" p  J" 3""5^^0r-
   t  J  __,Jb--N""",  *_s0M`""q_a@NW__JP^u_p"""p4a,p" _F""V--wL,_F_ F  #
 _,Jp*^#""9   L  5_a*N"""q__INr" "q_e^"*,p^""qME_ y"""p6u,f  j'  f "N^--LL_
    L  ]   k,w@#"""_  "_a*^E   ba-" ^qj-""^pe"  J^-u_f  _f "q@w,j   f  jL
    #_,J@^""p  `_ _jp-""q  _Dw^" ^cj*""*,j^  "p#_  y""^wE_ _F   F"^qN,_j
```

INTRODUCTION

The Following Project is a game that is called BRICKZ. The objective of the game is to clear the BRICKS present on the game screen using the ball and the board. Control the paddle at the bottom of the screen and Press the launch key to release the ball. The ball then goes whizzing about the screen and goes on to destroy bricks. Try to move the paddle in a way so you will be a good BRICK breaker. Scoring is based on the number of BRICKS destroyed.

INDEX

REQUIREMENTS

Hardware USED

- Printer; to print the required documents of the project.
- Compact Drive
- Processor : i5 Processor Intel

Software Required

- Operating system : Windows XP
- Turbo C++, for execution of program and
- MS Word, for Presentation of output.

ACKNOWLEDGEMENT

We are overwhelmed in all humbleness and gratefulness to acknowledge our debt to all those who have helped us to move these ideas well above the level of simplicity and into something concrete.

We are very thankful to our guide Mrs. PUJA MALHOTRA for her valuable help. She was always there to show us the right track when we needed her help. It is with the help of her valuable suggestions, guidance and encouragement, that we were able to perform this project work.

We would also like to thank our colleagues, who often helped and gave us support at critical junctures during the completion of this project.

ROHAN VERMA

HEMANSHU SONDHI

CERTIFICATE

This is to certify that
ROHAN VERMA & HEMANSHU SONDHI
of Class 12th A have successfully completed this activity under my supervision.

They have worked hard on this project very sincerely and honestly. This report has been examined and approved by me.

____

MRS PUJA MALHOTRA

THE PROJECT HAS BEEN DIVIDED INTO THREE SUB PARTS

- *LIB.H* → *CONTAINS THE CLASS DECLARATION.*

- *BRICK.EXE* → *CONTAINS THE MAIN PART of the project i.e. THE FUNCTIONS AND THE MAIN() BRICK GAME.*

- *LEVELER.EXE* → *USED TO DESIGN THE LEVELS.*

SOURCE CODE

LIB.H

```
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <process.h>
#include <ctype.h>
#include <string.h>
#include <iomanip.h>
#include <graphics.h>
#include <dos.h>

class Velocity;
class Brick;
class Ball;
class PowerUp;
class Board;
class Panel;
class TextBox;

class Velocity
{
	public:
	int sp, di;
};
class Panel
{
	public:
	int width,change;
	int points; char* pts;
	void score(int);
	void draw();
	Panel()
	{
		change = 1;
		width = 100;
		points = 0;
		pts = "";
	}
};
class Board
{
	public:
	int xC, yC, len, di,sp,height;
	void draw(void);
	void draw1(void);
	void brd(void);
	void clear();
	void pCatch(PowerUp& p, int index);
	void launch(Ball& b,int tedi);
	Board()
```

```
		{
			height = 7;
			xC=getmaxx()/2;
			sp = 15;
			yC = getmaxy()-height;
		}
};
class Ball
{
	public:
	int radius, xC, yC;
	int type,tcat;
	int alive;
	int m,n;
	int index;
	Velocity vC;
	void draw();
	void clear();
	void move(int,int);
	int  checkCollision(int,int,int,int);
	void collision();
	Ball()
	{
		vC.di = 2;
		vC.sp = 1;
		radius = 4;
		type = 0;
		alive = 0;
		tcat=0;
	}
};
class PowerUp
{
	public:
	int xC, yC, type;
	int height, width;
	int flag;
	int color;
	int alive;
	int fall(void);
	int clear(void);
	int draw(void);
	PowerUp()
	{
		height=10;
		width=50;
		flag = 0;
		alive = 0;
	}
};
class Brick
{
	public:
	int xC, yC;
	int alive, type;
	int height, width;
	int powerUp;
	PowerUp p;
```

```
        int f1,f2;
        int brickInit(void);
        void brickDesignInit(void);
        int aliveInit(void);
        Brick()
        {
               alive  =  1;
               height = 10;
               width  = 50;
               f1=0;
               f2=0;
               xC =0;yC =0;
               powerUp= brickInit();
         }

        int drawAlive(void);
        int drawDead(void);
        int brickCheck(Ball&);
};
```

BRICK.CPP

```cpp
//-------------------
#include <lib.h>
#include <time.h>
#include <fstream.h>
//-------------------
//Objects:
Panel panel;
Ball c[6];
Brick b[10][5];

//Music Frequencies
int
mE = 329,
mB = 493,
mFs = 698,
mEs = 659,
mGs = 783,
mG = 392,
mA = 440,
mD = 587,
mC = 523,
mF = 349;

//Movement Control Structure
//            -changeable in options menu in game.
struct key
{
      int lef,righ,laun;
}play1,play2;

void keysDeclare()
{
play1.lef=52;
play1.righ=54;
play1.laun=113;
}

class Star
{
      public:
      int xC, yC, s;
      int draw();

      Star()
      {
            draw();
      }

};

//------------------
```

```
//public vars:
int fb=0,brdstop=0,fc=0,ts=1,temp=0;
int nob=1;//no. of balls
unsigned int dx=6; //delay speed
char* levf;
int t=0;
int multi=0;
int level=1;
clock_t start, now;
double timeD;
char* currentPower = "  ";
char s1[2];
int highsc[5] = {0,0,0,0,0};
int go=0;
int launchb=1;
int m,n;

//-------------------


void Board::brd(void)
{
        setcolor(6);
        setfillstyle(1,6);
        sector(xC+len/2, yC+7/2,0,360, len/2, 7/2 );
}

void Board::draw(void)
{
        if(!brdstop || di==27 )
    {
                if(di==play1.lef)
                {
                        if(xC<=4)
                        {
                                brd();
                        }
                        else
                        {
                                sound(10000);
                                xC-=sp;
                                brd();
                        }
                }
                else if(di== play1.righ)
                {
                        if (xC>=getmaxx()-panel.width-len-4)
                        {
                                brd();
                        }
                        else
                        {
                                sound(10000);
                                xC+=sp;
                                brd();
                        }
                }
                else
```

```
                        if(di==27)
                                exit(0);
                else
                        brd();
        }
}

void Board::clear()
{
        if(!brdstop&&(di==play1.lef||di==play1.righ))
        {
                setcolor(0);
                setfillstyle(1,0);
                bar(xC, yC,xC+len,yC+7);
                setcolor(15);
        }
}

void Board::pCatch(PowerUp& p,int index)
{
        if ((p.yC + p.height >= yC) && (p.yC <=yC) && (( (p.xC >= xC) && (p.xC <= xC +
len) )|| ( (p.xC+p.width >= xC) && (p.xC+p.width <= xC + len))))
        {
                if(p.alive)
                {
                        if (p.type == 1)
                        {
                                brdstop=0;
                                c[index].type = p.type;
                                currentPower = "FIRE BALL";panel.change = 1;
                                dx=6;
                        }

                else
                        if (p.type == 2)
                        {
                                panel.score(100);
                                currentPower = "SCORE++";
                                panel.change = 1;
                                p.type=0;
                                dx=6;
                        }
                else
                        if (p.type == 3)
                        {
                                dx = 5;
                                currentPower = "FAST";panel.change = 1;
                                c[index].type = 0;
                        }
                else
                        if (p.type == 4)
                        {
                                dx = 8;
                                currentPower = "SLOW";panel.change = 1;
                                c[index].type = 0;
                        }
                else
                        if(p.type==5)
                        {
                                dx=6;
                                c[index].type=0 ;
                                currentPower = "MULTI";
                                t=1;
                                sound(350);
                                brdstop=0;
                        }
                }
        }
}

void Board::launch(Ball& b,int tedi)
{
        setcolor(0);
        setfillstyle(1,0);
        bar(b.xC-16+b.radius,b.yC-16,b.xC+16+b.radius,b.yC+b.radius);
        b.draw();
        setcolor(2);
        switch(b.vC.di)
        {
                case 1:
                        line(b.xC+b.radius,b.yC,b.xC-16+b.radius,b.yC-8);
                        break;
                case 2:
                        line(b.xC+b.radius,b.yC,b.xC-12+b.radius,b.yC-12);
                        break;
                case 3:
                        line(b.xC+b.radius,b.yC,b.xC-8+b.radius,b.yC-16);
                        break;
                case 4:
                        line(b.xC+b.radius,b.yC,b.xC+8+b.radius,b.yC-16);
                        break;
                case 5:
                        line(b.xC+b.radius,b.yC,b.xC+12+b.radius,b.yC-12);
                        break;
                case 6:
                        line(b.xC+b.radius,b.yC,b.xC+16+b.radius,b.yC-8);
                        break;
                        default : break;
        }
        setcolor(15);
}
//-------------------
void Ball::draw()
{       if (alive)
        {
                if (type == 0 || type==2)
                {
                        setcolor(15);
                        pieslice(xC+radius, yC+radius,0,360, radius);
                        setfillstyle(9, 5);
                        floodfill(xC+radius,yC+radius,15);
                        setcolor(15);
                }
                if (type == 1)
                {
                        setcolor(15);
                        circle(xC+radius, yC+radius,radius);
                        setfillstyle(1,62);
                        floodfill(xC+radius,yC+radius,15);
                        setcolor(15);
                        setcolor(62);
                        now = clock();
                        timeD = difftime(start,now);
                        if (timeD >= 5)
                        {
                                c[0].type = 0;
                                panel.change = 1;
                                currentPower = "";
                        }
                }
        }
}

void Ball::move(int di,int sp)
{
        m=xC;n=yC;
        switch (di)
        {
        case 1:
                xC-=2*sp;yC-=sp;
                break;
        case 2:
                xC-=sp;yC-=sp;
                break;
        case 3:
                xC-=sp;yC-=2*sp;
                break;
        case 4:
                xC+=sp;yC-=2*sp;
                break;
        case 5:
                xC+=sp;yC-=sp;
                break;
        case 6:
                xC+=2*sp;yC-=sp;
                break;
        case 7:
                xC+=2*sp;yC+=sp;
                break;
        case 8:
                xC+=sp;yC+=sp;
                break;
        case 9:
                xC+=sp;yC+=2*sp;
                break;
        case 10:
                xC-=sp;yC+=2*sp;
                break;
        case 11:
                xC-=sp;yC+=sp;
                break;
        case 12:
                xC-=2*sp;yC+=sp;
                break;
```

```
		default:
			break;
		}
}
void Ball::clear()
{
		if (alive)
		{
			if (type == 0 || type==2)
			{
				setcolor(0);
				setfillstyle(0,0);
				bar(m,n,m+2*radius,n+2*radius);
				pieslice(m+radius, n+radius,0,360, radius);
				setcolor(15);
			}
			if (type == 1)
			{
				setcolor(0);
				setfillstyle(0,0);
				bar(m,n,m+2*radius,n+2*radius);
				pieslice(m+radius, n+radius,0,360, radius);
				line(xC,yC,xC+5,yC+5);
				setcolor(15);
			}
		}
}

int Ball::checkCollision(int bX,int getmaxx,int getmaxy,int z)
{		if (alive)
			{
				if(xC <= 0 || yC <= 0 || xC >= getmaxx-(2*radius) || yC >= getmaxy-
(2*radius) )
				{
					if (xC<=0)
					{
						if (vC.di==2) vC.di=5;
						if (vC.di==11) vC.di=8;
						if (vC.di==3) vC.di=4;
						if (vC.di==10) vC.di=9;
						if (vC.di==1) vC.di=6;
						if (vC.di==12) vC.di=7;
					}
					if (yC<=0)
					{
						if (vC.di==5) vC.di=8;
						if (vC.di==2) vC.di=11;
						if (vC.di==4) vC.di=9;
						if (vC.di==3) vC.di=10;
						if (vC.di==6) vC.di=7;
						if (vC.di==1) vC.di=12;
					}
					if (xC>=getmaxx-(2*radius))
					{
						if (vC.di==8) vC.di=11;
						if (vC.di==5) vC.di=2;
						if (vC.di==7) vC.di=12;
						if (vC.di==6) vC.di=1;
```

```
                        if (vC.di==9) vC.di=10;
                        if (vC.di==4) vC.di=3;
                }

                if ((yC + 2*radius)>= (getmaxy-7))
                {
                        if(xC>=bX && xC<=bX+70 && type==2 && fb==1 )
                        {
                                if(vC.di>6)
                                vC.di=1;
                                tcat=1;
                                for(int k=0;k<6;k++)
                                {
                                        if(k!=z)
                                        {
                                                if(c[k].alive)
                                                {
                                                        if((c[k].yC + 2*radius)>=
(getmaxy-7))

                                                        if(!c[k].vC.sp)
                                                        {       c[k].tcat=0;
                                                                c[k].vC.sp=1;
                                                        }
                                                }
                                        }
                                }
                                vC.sp=0;
                                brdstop=1;
                        }
                        else
                        {
                                if ((xC >= bX) && (xC <= (bX + 11)) )
                                {
                                        if (vC.di == 10 || vC.di==12 ||
vC.di==11)

                                                vC.di=1;
                                        if (vC.di == 8 ||vC.di==9 ||vC.di==7)
                                                vC.di=1;
                                }
                                if((xC >= bX+12) && (xC <= (bX + 23)) )
                                {
                                        if (vC.di == 10 || vC.di==12 ||
vC.di==11)

                                                vC.di=2;
                                        if (vC.di == 8 ||vC.di==9 ||vC.di==7)
                                                vC.di=4;
                                }
                                if ((xC >= bX+24) && (xC <= (bX + 35)) )
                                {
                                        if (vC.di == 10 || vC.di==12 ||
vC.di==11)

                                                vC.di=3;
                                        if (vC.di == 8 ||vC.di==9 ||vC.di==7)
                                                vC.di=4;
                                }
                                if ((xC >= bX+36) && (xC <= (bX + 47)) )
                                {
```

```
                                                if (vC.di == 10 || vC.di==12 ||
vC.di==11)
                                                        vC.di=3;
                                                if (vC.di == 8 ||vC.di==9 ||vC.di==7)
                                                        vC.di=4;
                                        }
                                        if ((xC >= bX+48) && (xC <= (bX +59 )) )
                                        {
                                                if (vC.di == 10 || vC.di==12 ||
vC.di==11)
                                                        vC.di=3;
                                                if (vC.di == 8 ||vC.di==9 ||vC.di==7)
                                                        vC.di=5;
                                        }
                                        if ((xC >= bX+60) && (xC <= (bX + 70)) )
                                        {
                                                if (vC.di == 10 || vC.di==12 ||
vC.di==11)
                                                        vC.di=6;
                                                if (vC.di == 8 ||vC.di==9 ||vC.di==7)
                                                        vC.di=6;
                                        }
                                }
                        }
                        if (yC>=(getmaxy))
                        {
                                clear();
                                if (nob>0)
                                        nob--;
                                alive=0;
                                Ball bt;
                            if (!c[0].alive && nob>0)
                                {
                                        for(int i=0;i<6;i++)
                                        {
                                                if (c[i].alive)
                                                {
                                                        bt=c[0];
                                                        c[0]=c[i];
                                                        c[i]=bt;
                                                        goto d;
                                                }
                                        }
                                }
d:                              if(!nob)
                                {
                                        outtextxy(getmaxx/2-(textwidth("Game
Over!")/2),getmaxy/2-textheight("Game Over!"),"Game Over!");
                                        getch();
                                        go=1;
                                }
                        }
                }
        }
        return 0;
}

//-------------------
```

```
int PowerUp::fall()
{
      return 1;
}
int PowerUp::draw()
{
       if(alive)
       {
              char* msg1;
              setcolor(15);
              if(type == 1)
                     {
                            setfillstyle(5,color);
                            msg1=" FIRE " ;
                     }
              else
                     if (type == 2)
                     {
                            setfillstyle(5,color);
                            msg1="SCORE+";
                     }
              else
                     if (type == 3)
                     {
                            setfillstyle(5,color);
                            msg1=" FAST ";
                     }
              else
                     if (type == 4)
                     {
                            setfillstyle(5,color);
                            msg1 = " SLOW " ;
                     }
              else
                     if (type == 5)
                     {
                            setfillstyle(5,color);
                            msg1 = " MULTI " ;
                     }
              else
                     setfillstyle(5,12);
                     bar(xC,yC,xC+width,yC+height);
                     settextstyle(1,0,1);
                     setusercharsize(45,100,1,2);
                     outtextxy(xC +(width-textwidth(msg1))/2,yC+height,msg1);
                     bar(xC,yC+height
+textheight(msg1)+1,xC+width,yC+2*height+1+textheight(msg1));
       }
       return 0;
}
int PowerUp::clear()
{
       if(alive)
       {
              setcolor(0);
              setfillstyle(0,0);
              settextstyle(1,0,0);
```

```
                setusercharsize(45,100,1,2);
                bar(xC,yC-1,xC+width,yC+textheight("X")+ 2*height);
                setcolor(15);
        }
        return 0;
}

//-------------------

int Brick::brickInit(void)
{
      int r = random(2);
        int a = random(2);

        if(!r && !a)
                return random(5)+1;
        else
                return 0;
}

int Brick::aliveInit(void)
{
        int r = random(3);
        if (r == 0)
                r++;
        return r;
}
void Brick::brickDesignInit(void)
{
        int r = random(2);
        if(r == 0)
                alive = 0;
}


Brick::drawDead(void)
{
        if (!alive)
        {
                if (!f2)
                {
                        setcolor(0);
                        setfillstyle(0,0);
                        bar(xC,yC,xC+width,yC+height);
                        rectangle(xC, yC, xC+ width, yC + height);
                        setcolor(15);
                        f2=1;
                }
        }
        return 0;
}
Brick::drawAlive(void)
{
        if (alive)
        {
                if (!f1)
                {
                        setcolor(4);
```

```
				if(powerUp)
					setfillstyle(9,powerUp);
				else
					setfillstyle(9,12);
				bar(xC,yC,xC+width,yC+height);
				setcolor(15);
				f1=1;
			}
			setcolor(4);
			rectangle(xC, yC, xC+ width, yC + height);
		}
		return 0;
}
int Brick::brickCheck(Ball& b)
{
		if (b.type == 1)
		{
			if(	(	(b.yC > yC && b.yC < (yC + height))	&&	(b.xC > xC &&
b.xC < (xC + width))	)	)
			{
				alive=0;
				f2=0;
				return 1;
			}
		}
		if (b.vC.di==2 || b.vC.di==3 || b.vC.di==1 || b.vC.di==5 || b.vC.di==6 ||
b.vC.di==4)
		{
			if((b.yC >= yC) && (b.yC <= yC+height))
			{
				if(( (b.xC >= xC) && (b.xC < xC + width)) || ((b.xC+(2*b.radius)>=
xC ) && ((b.xC+(2*b.radius))<= (xC+width))))
				{
					if(alive)
					{
						f2=0;
						if(b.type == 0 || b.type == 2)
						{
							if(b.vC.di == 5) b.vC.di = 8;
							if(b.vC.di == 6) b.vC.di = 7;
							if(b.vC.di == 4) b.vC.di = 9;
							if(b.vC.di == 2)b.vC.di = 11;
							if(b.vC.di == 1)b.vC.di = 12;
							if(b.vC.di == 3)b.vC.di = 10;
							return 1;
						}
					}
				}
			}
		}
		if (b.vC.di==11 || b.vC.di==10 || b.vC.di==12 || b.vC.di==8 || b.vC.di==7 ||
b.vC.di==9)
		{
			if((b.yC + 2*b.radius >= yC)&&(b.yC + 2*b.radius <= yC+height))
			{
				if(( (b.xC >= xC) && (b.xC < xC + width)) || ((b.xC+(2*b.radius)>=
xC ) && ((b.xC+(2*b.radius))<= (xC+width))))
				{
```

```
                        if(alive)
                        {
                                f2=0;
                                if(b.type == 0 || b.type == 2 )
                                {
                                        if (b.vC.di == 11) b.vC.di = 2;
                                        if(b.vC.di == 10) b.vC.di = 3;
                                        if(b.vC.di == 12) b.vC.di = 1;
                                        if(b.vC.di == 8) b.vC.di = 5;
                                        if(b.vC.di == 7) b.vC.di = 6;
                                        if(b.vC.di == 9) b.vC.di = 4;
                                }
                                return 1;
                        }
                }
            }
    }
    if (b.vC.di==5 || b.vC.di==4 || b.vC.di==6 || b.vC.di==8 || b.vC.di==7 ||
b.vC.di==9)
    {
            if( (b.xC + 2*b.radius >= xC) && (b.xC + 2*b.radius <= xC+width))
            {
                    if(((b.yC >= yC) && (b.yC <= (yC + height))) || (((b.yC +
(2*b.radius))>=yC) && ((b.yC + (2*b.radius))<=(yC +height))))
                    {
                        if(alive)
                        {
                                f2=0;
                                if(b.type == 0 || b.type == 2)
                                {
                                        if(b.vC.di == 8) b.vC.di = 11;
                                        if(b.vC.di == 7) b.vC.di = 6;
                                        if(b.vC.di == 9) b.vC.di = 4;
                                        if(b.vC.di == 5) b.vC.di = 2;
                                        if(b.vC.di == 6) b.vC.di = 1;
                                        if(b.vC.di == 4) b.vC.di = 3;
                                        f2=0;
                                }
                                return 1;
                        }
                    }
            }
    }
    if (b.vC.di==11 || b.vC.di==10 || b.vC.di==12 || b.vC.di==2 || b.vC.di==1 ||
b.vC.di==3)
    {
            if( (b.xC <= xC+width) && (b.xC >= xC))
            {
                    if(((b.yC >= yC) && (b.yC <= (yC + height))) || (((b.yC +
(2*b.radius))>=yC) && ((b.yC + (2*b.radius))<=(yC +height))))
                  {
                        if(alive)
                        {
                                f2=0;
                                if(b.type == 0 || b.type == 2)
                                {
                                        if(b.vC.di == 11) b.vC.di = 8;
                                        if(b.vC.di == 12) b.vC.di = 7;
                                        if(b.vC.di == 10) b.vC.di = 9;
                                        if(b.vC.di == 2)b.vC.di = 5;
                                        if(b.vC.di == 1)b.vC.di = 6;
                                        if(b.vC.di == 3)b.vC.di = 4;
                                        f2=0;
                                }
                                return 1;
                        }
                }
        }
    }
    return 0;
}


//-------------------

void Panel::draw()
{
    //Base
    setfillstyle(1, 7);
    bar(getmaxx()-width, 0,getmaxx(),getmaxy());
    setfillstyle(0, 0);

    //Score
    char* strScore=itoa(points,strScore,10);
    setfillstyle(0, 7);

    //bar(getmaxx()-panel.width/2-textwidth(strScore)/2,0-
textheight(strScore),getmaxx()/2+textwidth(strScore),textheight(strScore));
    setcolor(0);
    settextstyle(1,0,2);
    outtextxy(getmaxx()-panel.width/2-(textwidth(strScore)/2),50-
textheight(strScore),strScore);
    outtextxy(getmaxx()-panel.width/2-(textwidth("SCORE:")/2),50-
textheight(strScore)-textheight(strScore),"SCORE:");
    outtextxy(getmaxx()-panel.width/2-(textwidth(currentPower)/2), 100-
textheight(currentPower),currentPower);

    //Default Change
    change = 0;
}
void Panel::score(int amount)
{
    points += amount;
    change = 1;
}

//-------------------

int Star::draw(void)
{
    delay(1);
    int temp = random(21);
    setcolor(temp);
    xC = rand() % getmaxx();
    yC = rand() % getmaxy();
    line(xC,yC,xC+s,yC+s);
```

```
        line(xC+s,yC,xC,yC+s);
        setcolor(15);
        settextstyle(0,0,2);
        outtextxy(xC,yC,"You Win!!");
        setcolor(0);
        settextstyle(4,0,6);
        outtextxy(getmaxx()/2-(textwidth("You Win!!")/2),getmaxy()/2-textheight("You
Win!!"),"You Win!!");
        return 0;
}

//-------------------
//Functions:

void split()
{
        //if (c[0].type==3)
        //{
        setcolor(0);
        bar(c[0].xC-10,c[0].yC-10,c[0].xC+2*c[0].radius+10,c[0].yC+2*c[0].radius+10);
        for(int i = 0; i <6; i++)
        {
                c[0].type=0;
                c[i]=c[0];
                c[i].vC.di=(i+1);
                c[i].alive = 1;
        }
        cleardevice();
        panel.change = 1;
        for (i = 0; i < 10; i++)
        {
                for (int j = 0; j < 5; j++)
                {
                        b[i][j].f1 = 0;
                }
        }
        nob=6;
}

void youWin()
{
        cleardevice();
        Star x[2500];
        getch();
}
void flash(int& x,int& y,int d )
{
        int poly[18]={      x, y, x+20, y, x+50, y+getmaxy()/24, x+20, y+getmaxy()/12, x,
y+getmaxy()/12, x, y+ (getmaxy()/12) - (2 * (getmaxy()/24) )/5, x+30, y+
(getmaxy()/24), x,y +  (3 * (getmaxy()/24) )/5, x,y };
        setfillstyle(1,d);
        drawpoly(9,poly);
        fillpoly(9,poly);
        x+=4;
}

void startMenu(int& x,int& y,int& m,int& n)
{   int d=1;
```

```
        setbkcolor(d);
        setcolor(d);
        flash(m,n,d);

        d=random(15);
        setcolor(d);
        m=x;n=y;

        flash(x,y,d);
        if (x>=getmaxx()/2+(textwidth("SINGLE PLAYER")/2))
              x=getmaxx()/2-(textwidth("SINGLE PLAYER")/2);

        setcolor(12);
        settextstyle(1,0,5);
        outtextxy(getmaxx()/2-(textwidth("SINGLE PLAYER")/2),2*getmaxy()/12," SINGLE
PLAYER ");
        outtextxy(getmaxx()/2-(textwidth("HIGH-SCORE")/2),2*getmaxy()/12+(textheight("S")
+ 10)," HIGH-SCORE ");
        outtextxy(getmaxx()/2-(textwidth("HELP")/2),2*getmaxy()/12 + 2*(textheight("S") +
10)," HELP ");
        outtextxy(getmaxx()/2-(textwidth("OPTIONS")/2),2*getmaxy()/12 +
3*(textheight("S") + 10)," OPTIONS ");
        outtextxy(getmaxx()/2-(textwidth("EXIT")/2),2*getmaxy()/12 + 4*(textheight("S") +
10)," EXIT ");
}

void gameStart()
{
        setcolor(5);
        settextstyle(4,0,12);
        outtextxy(getmaxx()/2-(textwidth("BRICKZ")/2),getmaxy()/2-
textheight("BRICKZ"),"BRICKZ");
        settextstyle(4,0,2);
        outtextxy(getmaxx()/2-(textwidth("Press any key to start...")/2),getmaxy()/2-
textheight("Press any key to start...")+200,"Press any key to start...");
        getch();
        cleardevice();
        settextstyle(0,0,1);
        outtextxy(getmaxx()/2-(textwidth("Loading...")/2),getmaxy()/2-
textheight("Loading..."),"Loading...");
        delay(1000);
        cleardevice();
}


void ballSwap(Ball& a, Ball& b)
{
        Ball t;
        t= a;
        a = b;
        b =t;
}

void readhigh()
{
        ifstream f;
        f.open("highscore.dat");
        f.read((char*)&highsc,sizeof(highsc)*5);
        f.close();
}
void highscore()
{
        readhigh();
        int k = 0;
        int temp=0;
        for(int  i=4;i>=0;i--)
        {
                if(highsc[i]<panel.points)
                {
                        temp=highsc[i];
                        highsc[i]=panel.points;
                        if(i!=4)
                        {
                                highsc[i+1]=temp;
                        }
                }
                else
                        i = -1;
        }
        ofstream f;
        f.open("highscore.dat");
        f.write((char*)&highsc,sizeof(highsc)*5);
        f.close();
}

void leveldesign()
{
        ifstream f;
        f.open(levf);
        for(int i=0;i<10;i++)
        {
                for(int k=0;k<5;k++)
                {
                        f.read((char*)&b[i][k],sizeof(b[i][k]));
                }
        }
        f.close();
        for( i = 0; i < 10; i++)
        {
                for (int j = 0; j < 5; j++)
                {
                        b[i][j].powerUp = b[i][j].brickInit();
                        b[i][j].f1 = 0;
                        b[i][j].f2 = 0;
                }
        }

}

void keyb(int& x1,int& y1)
{
        setcolor(12);
        settextstyle(1,0,5);
        outtextxy(100,getmaxy()/6," CONTROLS ");
        int t1=textheight("S");
        settextstyle(1,0,4);
        int t2=textheight("S");
        setcolor(13);
        outtextxy(100,getmaxy()/6+t1+10," PLAYER-1 ");
        setcolor(10);
        char x[2];
        x[1]='\0';

        x[0]=play1.lef;
        outtextxy(100,getmaxy()/6+t1+t2+10," LEFT ");
        outtextxy(260,getmaxy()/6+t1+t2+10,x);

        x[0]=play1.righ;
        outtextxy(100,getmaxy()/6+t1+2*t2+10," RIGHT ");
        outtextxy(260,getmaxy()/6+t1+2*t2+10,x);

        x[0]=play1.laun;
        outtextxy(100,getmaxy()/6+t1+3*t2+10," LAUNCH ");
        outtextxy(260,getmaxy()/6+t1+3*t2+10,x);

        x[0]=play2.lef;
        outtextxy(300,getmaxy()/6+t1+t2+10," LEFT ");
        outtextxy(560,getmaxy()/6+t1+t2+10,x);

        x[0]=play2.righ;
        outtextxy(300,getmaxy()/6+t1+2*t2+10," RIGHT ");
        outtextxy(560,getmaxy()/6+t1+2*t2+10,x);

        x[0]=play2.laun;
        outtextxy(300,getmaxy()/6+t1+3*t2+10," LAUNCH ");
        outtextxy(560,getmaxy()/6+t1+3*t2+10,x);

        int d=1;
        x1+=4;
        setbkcolor(d);
        setcolor(d);
        flash(m,n,d);
        d=random(15);

        setcolor(d);
        m=x1;n=y1;

        flash(x1,y1,d);
}

void posinit(Board& b1)
{
        if(launchb==1)
        {
                int a=0,temdir=0,flag1=0;
                b1.launch(c[0],temdir) ;
                temdir++;
                for(int i=0;i<6;i++)
                {
                        c[i].clear();
                        if(i!=0)
                        c[i].alive=0;
                }
                c[0].xC=b1.xC + b1.len/2;
```

```
            c[0].yC=b1.yC-2*c[0].radius;
            c[0].draw();
            while(launchb==1)
            {
                  temdir=c[0].vC.di;
                  if(kbhit())
                  {
                        a=getch();
                        if(a==play1.lef&& temdir!=1)
                        {
                              c[0].vC.di--;
                              b1.launch(c[0],temdir) ;
                        }
                        if(a==play1.righ&& temdir!=6)
                        {
                              c[0].vC.di++;
                              b1.launch(c[0],temdir) ;
                        }
                        if(a == play1.laun)
                              launchb=0;
                        if(a==27)
                              exit(0);
                  }
            }
      }
 }

int main()
{
refresh:
      panel.points = 0;
      keysDeclare(); // initialize the keys for the game
      s1[1]='/0'; //used for the levels

      //GRAPHICS INITIALIZATION:
      /* request auto detection */int gdriver = DETECT, gmode, errorcode;/* initialize
graphics and local variables */initgraph(&gdriver, &gmode, "");/* read result of
initialization */errorcode = graphresult(); if (errorcode != grOk)  /* an error
occurred */{printf("Graphics error: %s\n", grapherrormsg(errorcode));printf("Press any
key to halt:");getch();exit(1); /* terminate with an error code */}

      /* GAME BEGINS */
      setbkcolor(BLUE);
      int l=0,gamest=0,xs=getmaxx()/2-textwidth("SINGLE PLAYER")/2,ys=getmaxy()/6
+10,xs1=-1,ys1=-1; //variables

      //Welcome Screen:
      gameStart(); //the gameStart function opens the BRICKZ screen

      randomize();
      int n=1;

      //index the balls
      for (int inc = 0; inc < 6; inc++)
      {
            c[inc].index = inc;
      }
```

```
        //Home Screen Music
        int mI=0,mI2=0;
        int note[15] = {mC, mD, mE, mF, mG, mA, mB, mC, mB, mA, mG, mF, mE, mD, mC}; //Sa
Re Ga Ma Pa Dha Ni Sa Ni Dh Pa Ma Ga Re Sa

    MainScreen:
        l=0,gamest=0,xs=getmaxx()/2-textwidth("SINGLE PLAYER")/2,ys=getmaxy()/6 +10,xs1=-
1,ys1=-1;
        //stay in startMenu untill the game has started
        while(!gamest)
        {
                mI2++;
                startMenu(xs,ys,xs1,ys1); // startMenu() opens the Menu Screen

                //Music
                sound(note[mI]);
                if(mI2 % 100 == 0)
                {
                        mI++; mI2 = 0;
                }
                delay(5);
                if (mI > 15)
                        mI = 0;

                //check the keyboard buffer and fetch the key to variable l
                if(kbhit())
                {
                        l=getch();
                }
                //Check l
                if (l==13)
                {
                        if (n==1||n==2||n==3||n==5||n==4)
                        gamest=1;
                        //if(n==3)
                }
                else
                        if (l==50)
                        {
                                if(n<5)
                                {       ys+=(textheight("S")+10);
                                        n++;
                                }
                        }
                else
                        if (l==56)
                        {
                                if (n>1)
                                {
                                        ys-=(textheight("S")+10);
                                        n--;
                                }
                        }
                        l=0;
                }
                cleardevice();

                //HELP File
```

```
            int p=0,x=100,y=getmaxy()/6+74+10;
            if(n==3)
            {
                    cleardevice();
                    int yC = -50;
                    settextstyle(1,0,2);nosound();
                    int z;
                    do
                    {
                            if(kbhit())
                            {
                                    z=getche();
                                    if(z==56)
                                    {
                                            yC-=50;
                                            cleardevice();
                                    }
                                    else
                                            if(z == 50 )
                                    {
                                            yC+=50;
                                            cleardevice();
                                    }

                                    if( z==27)
                                    {
                                            n=1;
                                            gamest=0;
                                            l=0;
                                            cleardevice();
                                            z=0;
                                            goto MainScreen;
                                    }
                                    z=0;
                            }
                            outtextxy(10,yC + 50,"CONTROLS");
                            outtextxy(10,yC + 100,"Use LEFT to move the board Left.
[DEFAULT num4]");
                            outtextxy(10,yC + 150,"Use RIGHT to move the board Right.
[DEFAULT num6]");
                            outtextxy(10,yC + 200,"You can launch the ball by pressing
LAUNCH [DEFAULT Q]");
                            outtextxy(10,yC + 250," ");
                            outtextxy(10,yC + 300,"HOW TO PLAY");
                            outtextxy(10,yC + 350,"Move the board left and right to
collide with");
                            outtextxy(10,yC + 400,"board but make sure the ball does not
fall off the screen!!");
                            outtextxy(10,yC + 450," ");
                            outtextxy(10,yC + 500,"POWERUPS");
                            outtextxy(10,yC + 550,"FIRE:     The fire powerup puts the
ball on fire, allowing it");
                            outtextxy(10,yC + 600,"          to destroy the bricks
without it changing its path");
                            outtextxy(10,yC + 650,"SCORE++:  Increases your score
instantly");
                            outtextxy(10,yC + 700,"MULTI:    The balls split up into 6
balls allowing you to ");
```

```
                        outtextxy(10,yC + 750,"         cause more havoc in less 
time");
                        outtextxy(10,yC + 800,"FAST:    The FAST powerup causes your 
ball to go into the next gear");
                        outtextxy(10,yC + 850,"SLOW:    Getting too fast for you? 
Get one!");
                }while(z != 27);
                n=1;
                gamest=0;
                l=0;
                cleardevice();
                goto MainScreen;
        }
        if(n==4)
        {
                int m=1;
                do{
                        p=0;
                        if(kbhit())
                        {
                                p=getch();
                                if (p==50)
                                {
                                        if(m!=3&&m!=6)
                                        {
                                                settextstyle(1,0,4);
                                                y+=(textheight("S"));
                                                m++;
                                        }
                                }
                                if (p==56)
                                {
                                        if (m!=1&&m!=4)
                                        {
                                                y-=(textheight("S"));
                                                m--;
                                        }
                                }
                                if (p==52)
                                {
                                        if(m>=4)
                                        {
                                                settextstyle(1,0,4);
                                                x-=300;
                                                m-=3;
                                        }
                                }
                                if (p==54)
                                {
                                        if (m<=3)
                                        {
                                                x+=300;
                                                m+=3;
                                        }
                                }
                        }
                        if(m<4)
                        {
```

```
                        if(x>=250)
                        x=100;
                }
                else
                        if(m<=6)
                        {
                                if(x>=550)
                                x=300;
                        }
                int a;
                if(p==13)
                {
                        a=getche();
                        switch (m)
                        {
                                case 1:play1.lef=a;break;
                                case 2:play1.righ=a;break;
                                case 3:play1.laun=a;break;
                                case 4:play2.lef=a; break;
                                case 5:play2.righ=a;break;
                                case 6:play2.laun=a;break;
                        }

                }
                if(p==27)
                {
                        gamest=0;
                        n=1;
                        ys=getmaxy()/6 +10;
                        cleardevice();
                        goto MainScreen;
                }
                keyb(x,y);
        }while(p!=27||p!=52) ;

}
cleardevice();
setbkcolor(0);
if(n==5 && gamest)
{
        nosound();
        exit(0);
}
//Declare Maximums of the Screen
const int mX=getmaxx()-panel.width,mY=getmaxy(); int a=0;
levf="level1.lvl";

//DECLARING OBJECTS:-
// declaring board
Board b1,b2;
b1.di = 53; b1.len = 70;
b1.xC = ((mX/2)-((b1.len)/2));
// declaring ball
c[0].alive =1;
c[0].xC=b1.xC + b1.len/2;
c[0].yC=b1.yC-2*c[0].radius;
int z = 5;
//init Variables
```

```
		int tedi=0; //Temporary Direction
		int bricksAlive = 10 * 5;
		//-------------------
		//GAME LOOP:-
		//-------------------
		//      preRender() ;
		readhigh();
	levR:
		itoa(level,s1,10);
		levf[5]=s1[0];
		leveldesign();
		launchb=1;
		int fn=0,tick=10;
		if (n==2)
		{
			nosound();
			highscore();
			cleardevice();
			settextstyle(3,2,1);
			setbkcolor(2);
			closegraph();
			cout << "\n\t\t\t\tHIGHSCORE "<<endl;
			for(int i = 0; i < 5; i++)
			{
				cout << "\t\t\t\t #" << (i + 1) << "." << highsc[i]<< endl;
			}
			getch();
			level = 1;
		}
		do
		{
			nosound(); //undoSound
			setcolor(0);
			line(getmaxx()-panel.width,0,getmaxx()-panel.width,getmaxy());
			line(getmaxx()-panel.width+1,0,getmaxx()-panel.width+1,getmaxy());
			if (panel.change)
			{
				panel.draw();
			}
			if (n==1)
			{
				b1.di = 1;
				b2.di=1;
			}
			if (t)
			{
				for(int x=0;x<6;x++)
				{
					if(c[x].alive)
						c[x].clear();
				}
				split();
				t = 0;
			}
			if (tick>0)
				tick--;
			for(int k=0;k<6;k++)
			{
```

```
                                if(c[k].alive)
                                //Check the keyboard buffer:-
                                {
                                        if(kbhit() )
                                        {
                                                a = getche();

                                                b1.di=a;

                                                if(a==27 )
                                                {
qw:                                                     highscore();
                                                        cleardevice();
                                                        settextstyle(3,2,1);
                                                        setbkcolor(2);
                                                        closegraph();
                                                        cout << "\n\t\t\t\tHIGHSCORE "<<endl;
                                                        for(int i = 0; i < 5; i++)
                                                        {
                                                                cout << "\t\t\t\t #" << (i + 1) <<
"." << highsc[i]<< endl;
                                                        }
                                                        getch();
                                                        level = 1;
                                                        goto refresh;
                                                }
                                        }
                                }
                        }
                        for (int z=0; z < 6; z++)
                        {
                                if(c[z].alive)
                                {
                                        c[z].move(c[z].vC.di,c[z].vC.sp);
                                        c[z].checkCollision(b1.xC,mX,mY-7,z);
                                        if (go)
                                        {
                                                go=0;
                                                cleardevice();
                                                n=1;
                                                gamest=0;
                                                goto qw;
                                        }
                                }
                        }
                        bricksAlive = 10 * 5;
                        for (int i = 0; i < 10; i++)
                        {
                                for (int j = 0; j < 5; j++)
                                {
                                        for (z=0; z < 6; z++)
                                        {
                                                if(c[z].alive)
                                                {
                                                        if(b[i][j].brickCheck(c[z]))
                                                        {
                                                                if (b[i][j].alive)
                                                                {
```

```
                                                                panel.score(5);
                                                                sound(110);
                                                        }
                                                        if(b[i][j].powerUp)
                                                        {
                                                                if(b[i][j].alive)
                                                                {
                                                                        b[i][j].p.xC =
b[i][j].xC;
                                                                        b[i][j].p.yC =
b[i][j].yC ;
                                                                        b[i][j].p.flag = 1;
                                                                        b[i][j].p.alive = 1;
                                                                        if(b[i][j].powerUp !=
0)
                                                                        {
                                                                                b[i][j].p.type
= b[i][j].powerUp;
                                                                                b[i][j].p.color
= b[i][j].powerUp;
                                                                        }
                                                                }
                                                        }
                                                        b[i][j].alive = 0;
                                                }
                                        }
                                }
                                // used to not draw powerup if it yC between other
brick.
                                if(b[i][j].p.flag)
                                {
                                        for(int k=0;k<10;k++)
                                        {
                                                for(int p=0;p<5;p++)
                                                {
                                                        if(b[k][p].xC==b[i][j].p.xC)
                                                                if
((b[k][p].yC+b[k][p].height+1) == b[i][j].p.yC )
                                                                {
                                                                        if (b[k][p].alive)
                                                                                b[k][p].f1=0;
                                                                        else
                                                                                b[k][p].f2=0;
                                                                }
                                                }
                                        }
                                }
                                if (b[i][j].alive == 0)
                                        bricksAlive--;
                                if(b[i][j].p.flag)
                                {
                                        b[i][j].p.clear();
                                        b[i][j].p.draw();
                                        b[i][j].p.yC++;
                                        if (b[i][j].p.yC > getmaxy() && b[i][j].alive ==
1)
                                        {
                                                b[i][j].p.alive = 0;
```

```
                }
                for(int k=0;k<6;k++)
                {
                    if(c[k].alive)
                    {
                        b1.pCatch(b[i][j].p, c[k].index);
                    }
                }
            }
        }
    }
    for (i = 0; i < 10; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            b[i][j].drawDead();
        }
    }
    for (i = 0; i < 10; i++)
    {
        for (int j = 0; j < 5; j++)
        {
            b[i][j].drawAlive();
        }
    }
    //-- Clearing Objects
    delay(dx);
    for (z=0; z < 6; z++)
    {
        if(c[z].alive)
        {
            c[z].clear();
            c[z].draw();
        }
    }
    b1.clear();
    b1.draw();
    posinit(b1);
} while(bricksAlive != 0);
if(level<5)
{
    bricksAlive = 10*5 ;
    for(int i=0;i<10;i++)
    {
        for(int j=0;j<5;j++)
        {
            b[i][j].alive=1;
            b[i][j].f1=0;
        }
}
level++;
cleardevice();
for(int q=0;q<6;q++)
{
    c[q].alive=0;
    c[q].clear();
}
b1.xC = ((mX/2)-((b1.len)/2));
```

```
            c[0].clear();
            c[0].alive=1;
            c[0].type=0;
            c[0].xC=b1.xC + b1.len/2;
            c[0].yC=b1.yC-2*c[0].radius;
            c[0].draw();
            goto levR;
        }
        getch();
        youWin();
        /* clean up */
        closegraph();
        return 0;
}
```

LEVELER.CPP

```
/*THE LEVEL CREATOR*/
//-------------------
#include <lib.h>
//-------------------
Brick b[10][5] ;
Panel panel;
char* levf;
int Brick::brickInit(void)
{
	int r = random(2);
	int a=random(2);
	if(!r&&!a)
		return random(4);
	else
		return 0;
}
void levelini()
{
	int ixC = 200, iyC = 100;
	for (int i = 0; i < 10; i++)
	{
		ixC=200;
		for (int j = 0; j < 5; j++)
		{
			b[i][j].xC=ixC;
			ixC+=50;
			b[i][j].yC =iyC;
		}
		iyC+=10;
	}
}

Brick::drawDead(void)
{
	if (!alive)
	{
		if (f2)
		{
			setcolor(0);
			setfillstyle(0,0);
			bar(xC,yC,xC+width,yC+height);
			rectangle(xC, yC, xC+ width, yC + height);
			setcolor(15);
			f2=1;
		}
	}
	return 0;
}
Brick::drawAlive(void)
```

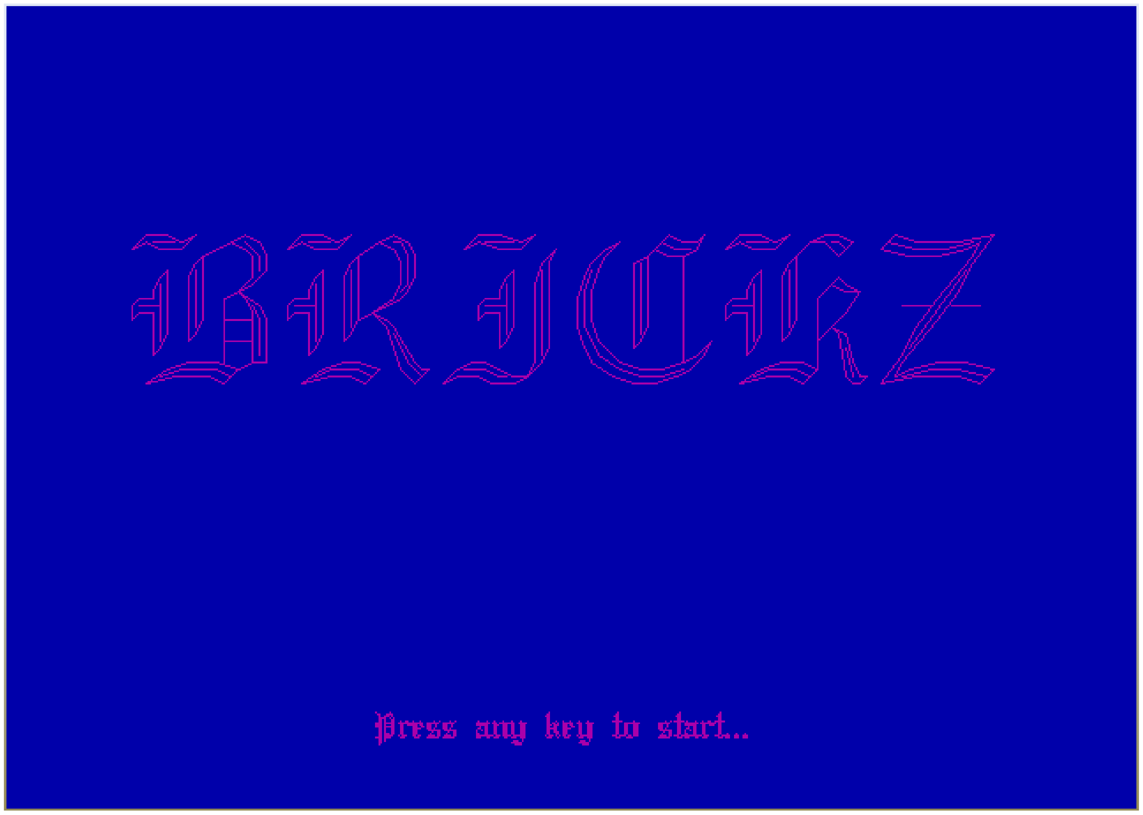
```
{
	if (alive)
	{
		if (!f1)
		{
			setcolor(12);
			if(powerUp == 1)
				setfillstyle(5,14);
			else
				if (powerUp == 2)
					setfillstyle(5,11);
			else
				if (powerUp == 3)
					setfillstyle(5,13);
			else
				setfillstyle(5,12);
			bar(xC,yC,xC+width,yC+height);
			rectangle(xC, yC, xC+ width, yC + height);
			setcolor(15);
			f1=1;
		}
	}
	return 0;
}
//-------------------

void Panel::draw()
{
	//Base
	setfillstyle(1, 7);
	bar(getmaxx()-width, 0,getmaxx(),getmaxy());
	setfillstyle(0, 0);
	//Score
	char* strScore=itoa(points,strScore,10);
	setfillstyle(0, 7);
	bar(getmaxx()-panel.width/2-textwidth(strScore)/2,0-
textheight(strScore),getmaxx()/2+textwidth(strScore),textheight(strScore));
	settextstyle(0,0,1);
	outtextxy(getmaxx()-panel.width/2-(textwidth(strScore)/2),getmaxy()/2-
textheight(strScore),strScore);
	change = 0;
}
void Panel::score(int amount)
{
	points += amount;
	change = 1;
}

//-------------------

void gameStart()
{
	setcolor(5);
	settextstyle(4,0,12);
	outtextxy(getmaxx()/2-(textwidth("BRICKZ")/2),getmaxy()/2-
textheight("BRICKZ"),"BRICKZ");
	settextstyle(4,0,2);
	outtextxy(getmaxx()/2-(textwidth("Press any key to start...")/2),getmaxy()/2-
textheight("Press any key to start...")+200,"Press any key to start...");
	getch();
	cleardevice();
	settextstyle(0,0,1);
	outtextxy(getmaxx()/2-(textwidth("Loading...")/2),getmaxy()/2-
textheight("Loading..."),"Loading...");
	delay(1000);
	cleardevice();
}

void fileHandle()
{
	ofstream f;
	f.open(levf);
	f.write((char*)&b,sizeof(Brick)*50);
	f.close();
}

void fileHandler()
{
	ifstream f;
	f.open(levf);
	f.read((char*)&b,sizeof(Brick )* 50);
	for (int i = 0; i < 10; i++)
	{
		for (int j = 0; j < 5; j++)
		{
			f.read((char*)&b[i][j],sizeof(b[i][j]));
		}
	}
	f.close();
	for(i=0;i<10;i++)
	{
		for(int j=0;j<5;j++)
		{
			b[i][j].f1=0;
		}
	}
}

//-------------------
int briche(int i1,int j1)
{
	for(int r=0;r<10;r++)
	{
		for(int u=0;u<5;u++)
		{
			if(r!=i1 || u!=j1)
			{
				if((b[i1][j1].xC==b[r][u].xC) && (b[i1][j1].yC==b[r][u].yC))
				goto q;
			}
		}
	}
	return 1;
q:	return 0;
}
```

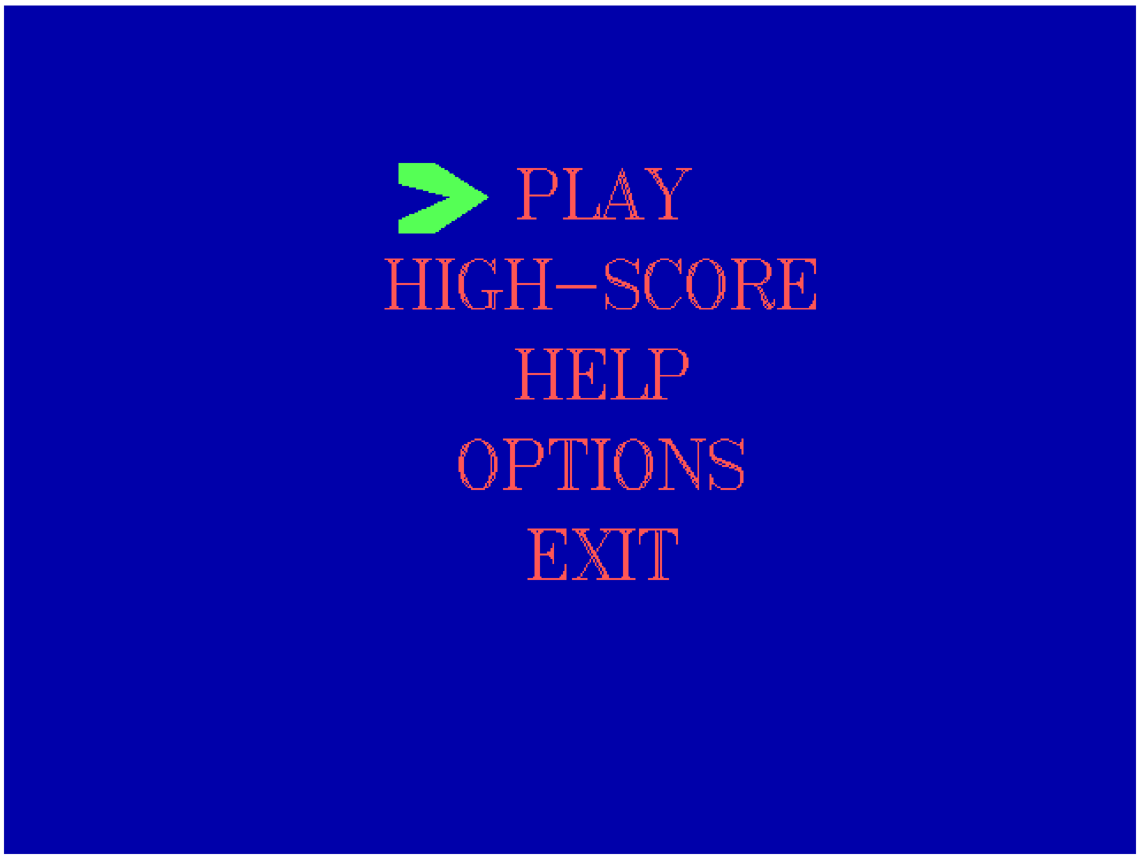
```
//main:
int main()
{      //GRAPHICS INITIALIZATION:
       /* request auto detection */int gdriver = DETECT, gmode, errorcode;/* initialize
graphics and local variables */initgraph(&gdriver, &gmode, "");/* read result of
initialization */errorcode = graphresult();
       if (errorcode != grOk)  /* an error occurred */
       {printf("Graphics error: %s\n", grapherrormsg(errorcode));printf("Press any key
to halt:");getch();exit(1); /* terminate with an error code */}
       /* start code */

       levf="LEVEL1.LVL\0";
       //Welcome Screen:
       gameStart();
       char lev;
       int a=0,bricksAlive = 10 * 5, bricnum=1,brchan=0;
       unsigned int i1=0,j1=0;
       //--------------------
       //GAME LOOP:-
       //-------------
       levelini();
       cout<<"enter level to draw/edit? ";
       cin>>lev;
       levf[5]=lev;`
    fileHandler();
       cleardevice();
       setcolor(10);
       settextstyle(3,0,2);
       outtextxy(getmaxx()-textwidth("CHANGE"),100,"CHANGE");
       do
       {
              setcolor(12);

       rectangle(b[i1][j1].xC,b[i1][j1].yC,b[i1][j1].xC+b[i1][j1].width,b[i1][j1].yC+b[i
1][j1].height);
              if (panel.change)
              {
                     panel.draw();
              }
              j1=(bricnum%5);
           i1=(bricnum/5);
              setcolor(15);

       rectangle(b[i1][j1].xC,b[i1][j1].yC,b[i1][j1].xC+b[i1][j1].width,b[i1][j1].yC+b[i
1][j1].height);
              setcolor(9);
              //Check the keyboard buffer:-
              if(kbhit())
              {
                     a = getche();
                     if(a==113)
                     {
                            if(!brchan)
                                   brchan=1;
                            else
                            brchan=0;
                     }
```

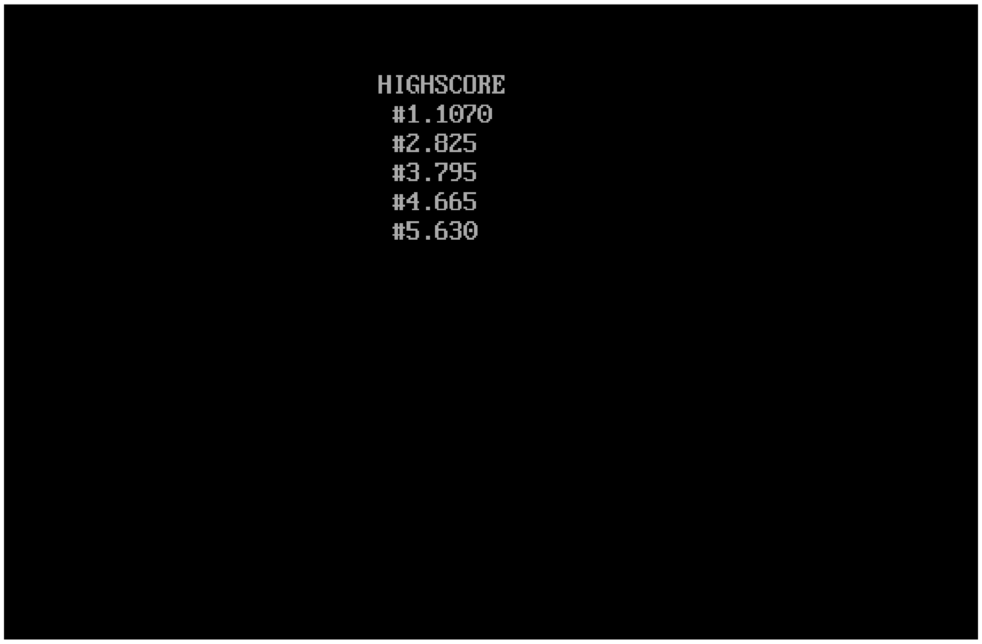
```
if (a==50 )
{
	if(!brchan)
		bricnum+=5;
	else
	{
		b[i1][j1].f2=1;
		b[i1][j1].alive=0;
		b[i1][j1].drawDead();
		b[i1][j1].f1=0;
		b[i1][j1].alive=1;
		b[i1][j1].yC+=b[i1][j1].height;

		if(!briche(i1,j1))
			b[i1][j1].yC-=b[i1][j1].height;
		b[i1][j1].drawAlive();
	}
}
if(a==56)
{
	if(!brchan)
		bricnum-=5;
	else
	{
		b[i1][j1].f2=1;
		b[i1][j1].alive=0;
		b[i1][j1].drawDead();
		b[i1][j1].f1=0;
		b[i1][j1].alive=1;
		b[i1][j1].yC-=b[i1][j1].height;
		if(!briche(i1,j1))
			b[i1][j1].yC+=b[i1][j1].height;
		b[i1][j1].drawAlive();
	}
}
if(a==52)
{
	if(!brchan)
		bricnum-=1;
	else
	{
		b[i1][j1].f2=1;
		b[i1][j1].alive=0;
		b[i1][j1].drawDead();
		b[i1][j1].f1=0;
		b[i1][j1].alive=1;
		b[i1][j1].xC-=b[i1][j1].width;
		if(!briche(i1,j1))
			b[i1][j1].xC+=b[i1][j1].width;
		b[i1][j1].drawAlive();
	}
}
if(a==54)
{
	if(!brchan)
		bricnum+=1;
	else
	{
```

```
                    b[i1][j1].f2=1;
                    b[i1][j1].alive=0;
                    b[i1][j1].drawDead();
                    b[i1][j1].f1=0;
                    b[i1][j1].alive=1;
                    b[i1][j1].xC+=b[i1][j1].width;
                    if(!briche(i1,j1))
                        b[i1][j1].xC-=b[i1][j1].width;
                    b[i1][j1].drawAlive();
                }
            }
            if(a==27)
            {
                fileHandle();
                bricksAlive = 0;
            }
            a=0;
        }
        for (i = 0; i < 10; i++)
        {
            for (int j = 0; j < 5; j++)
            {
                b[i][j].drawDead();
            }
        }
        for (i = 0; i < 10; i++)
        {
            for (int j = 0; j < 5; j++)
            {
                b[i][j].drawAlive();
            }
        }
    } while(bricksAlive != 0);
    closegraph();
    return 0;
}
```

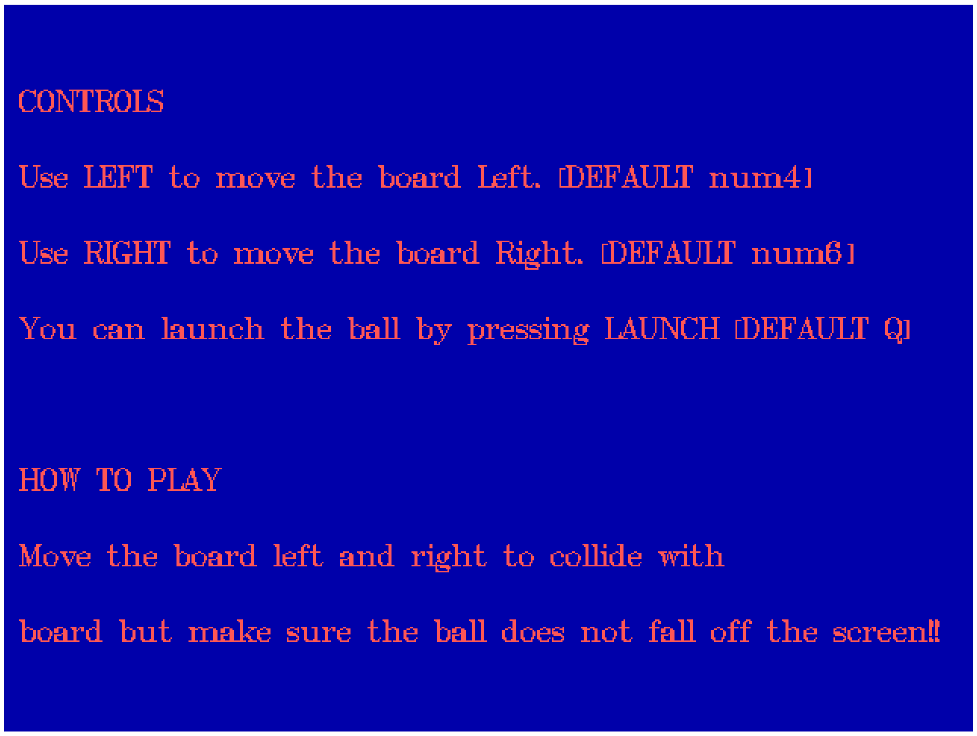
OUTPUTS


BRICKZ
Press any key to start..

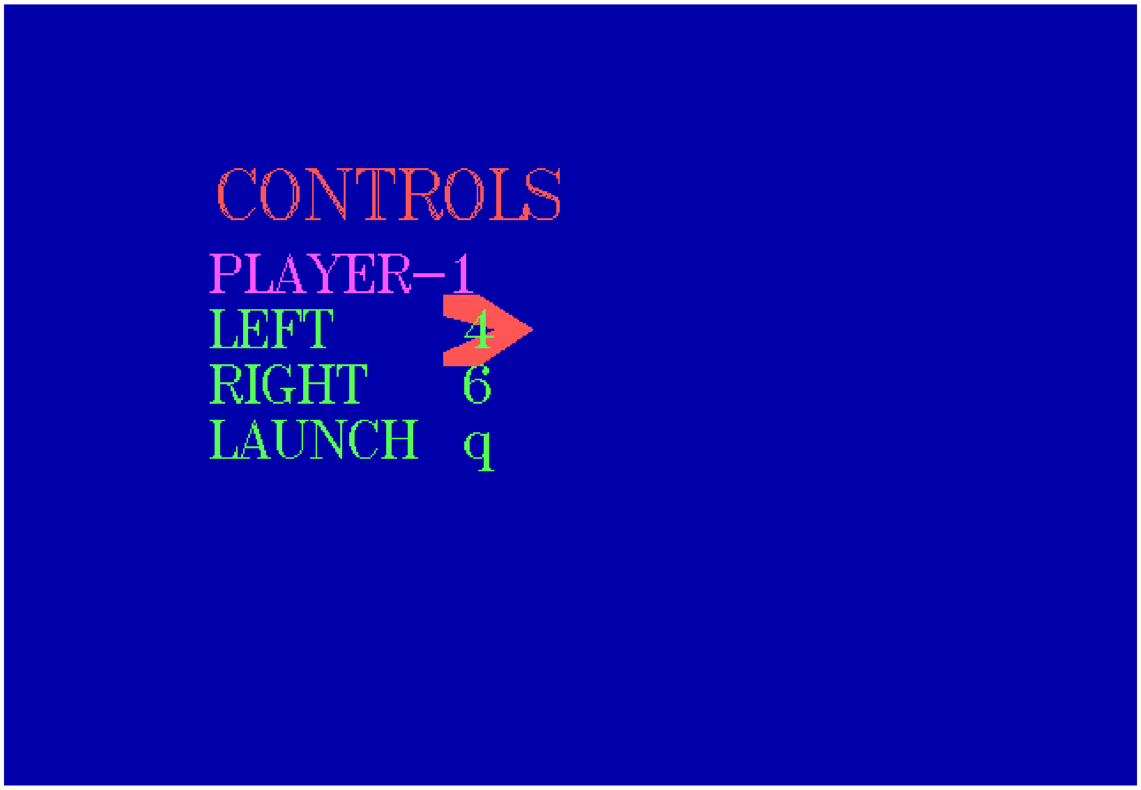

PLAY
HIGH-SCORE
HELP
OPTIONS
EXIT

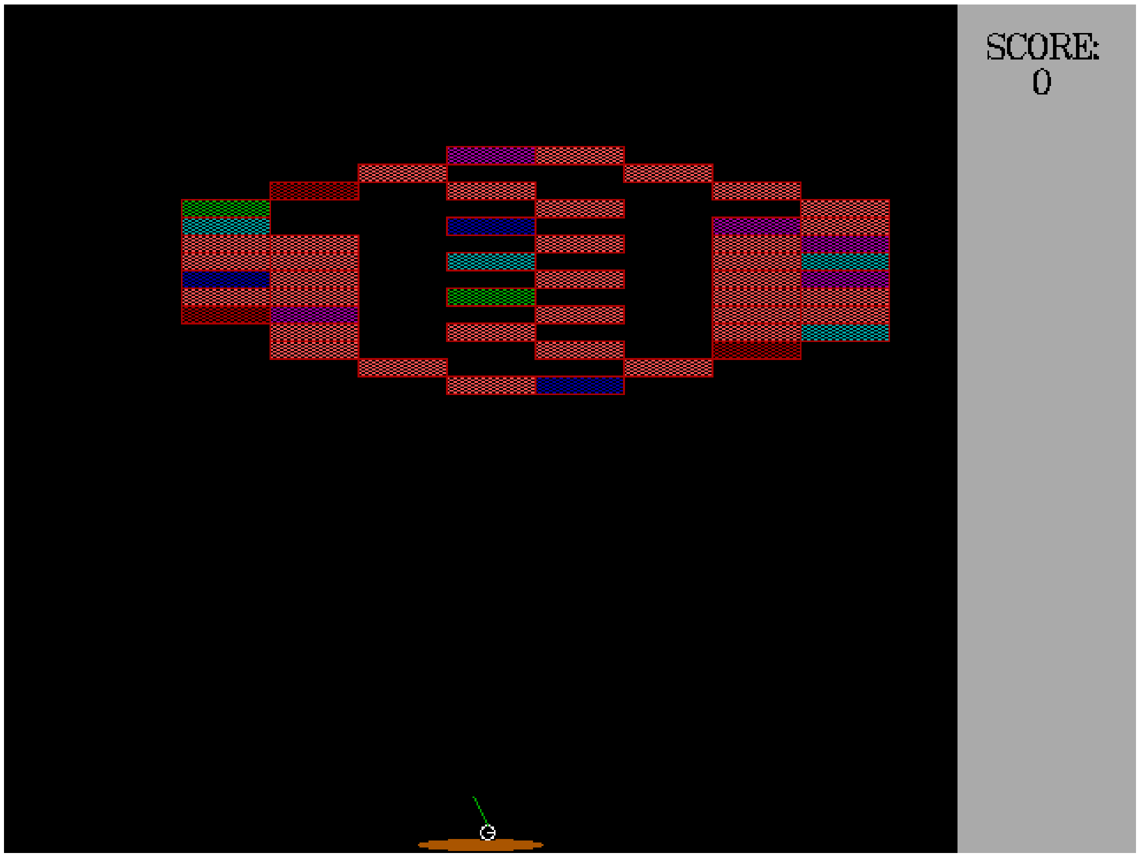


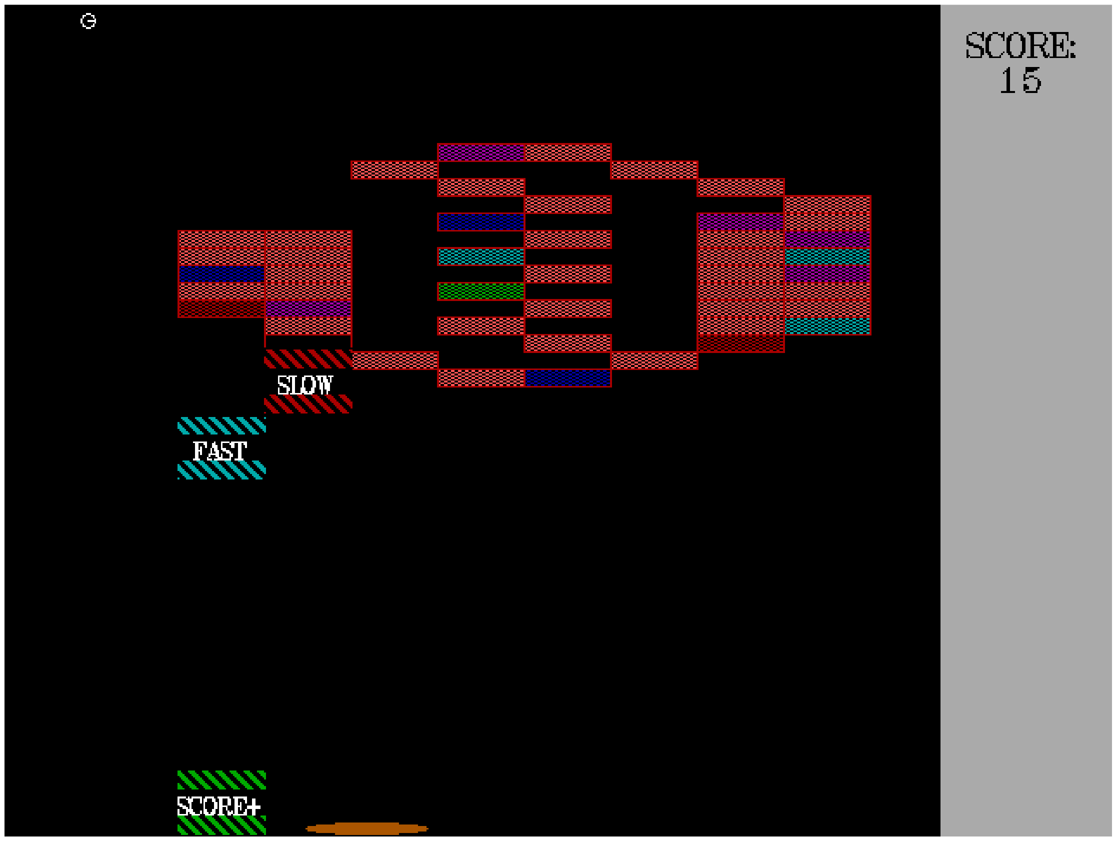
```
HIGHSCORE
 #1.1070
 #2.825
 #3.795
 #4.665
 #5.630
```

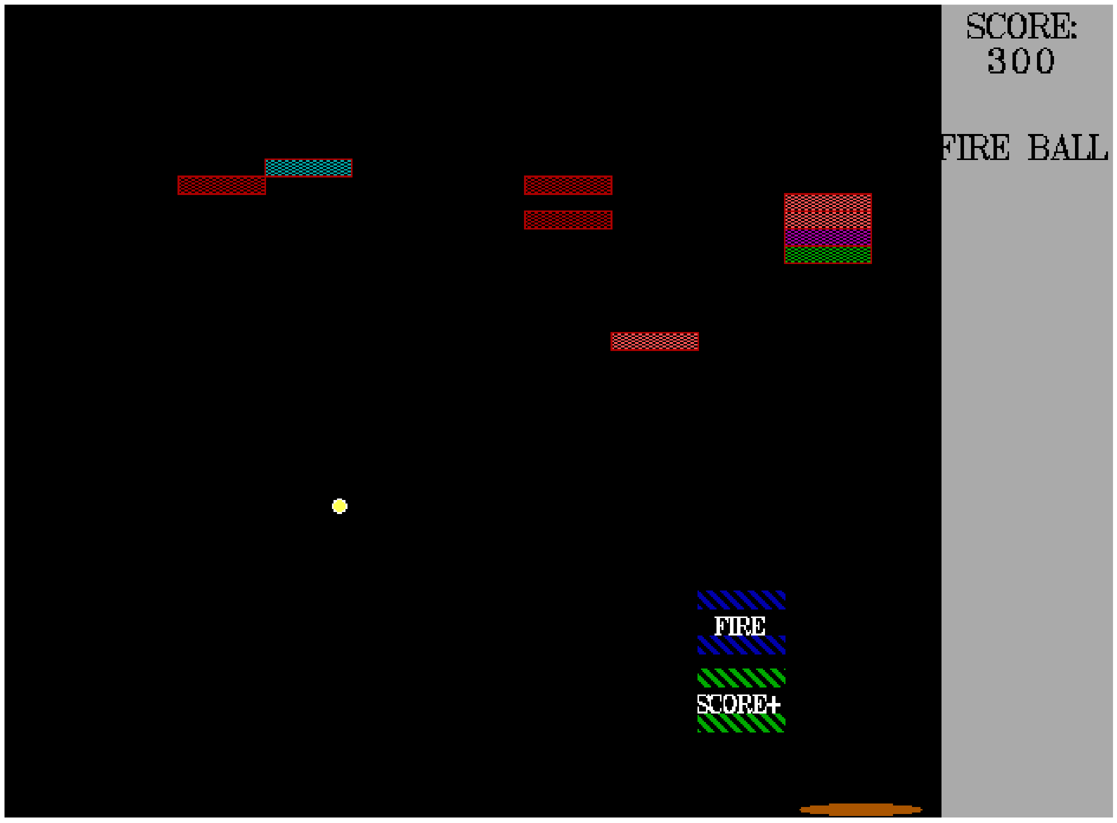


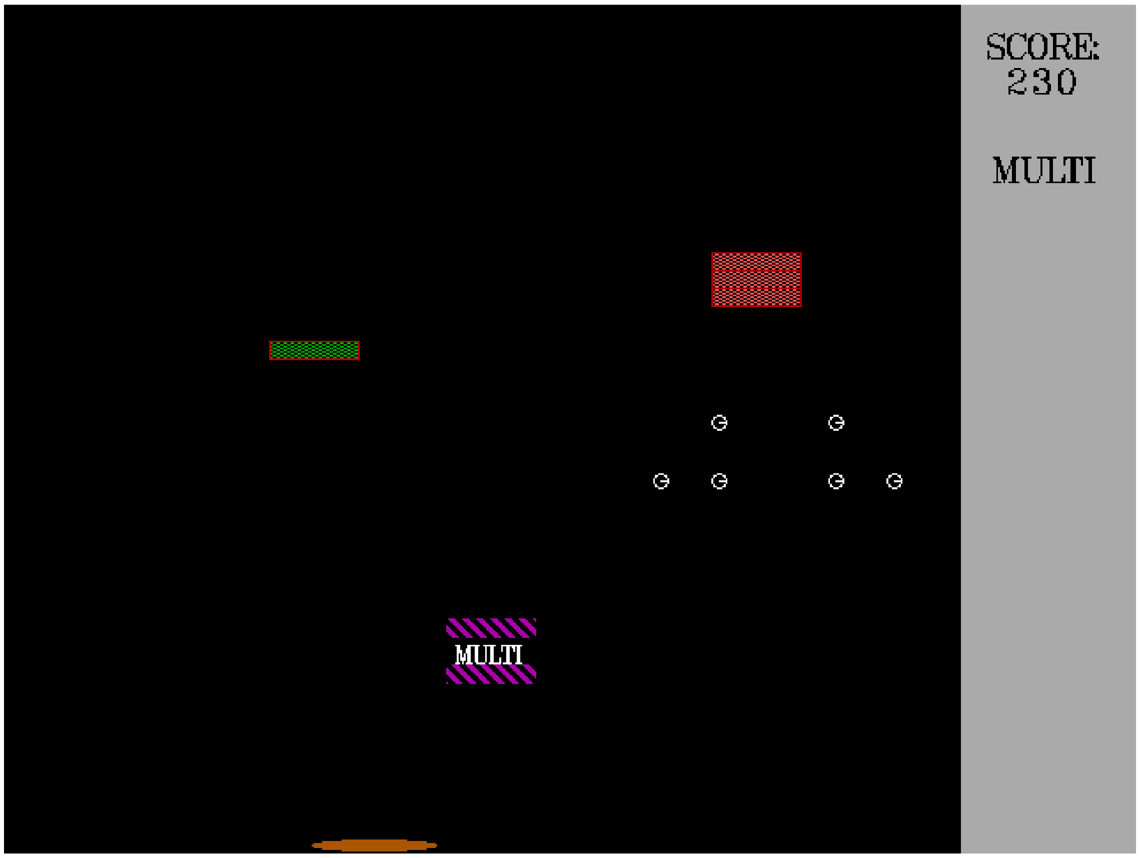
```
CONTROLS

Use LEFT to move the board Left. [DEFAULT num4]

Use RIGHT to move the board Right. [DEFAULT num6]

You can launch the ball by pressing LAUNCH [DEFAULT Q]


HOW TO PLAY

Move the board left and right to collide with

board but make sure the ball does not fall off the screen!!
```


CONTROLS
PLAYER-1
LEFT 4
RIGHT 6
LAUNCH q




SCORE:
15
SLOW
FAST
SCORE+


SCORE:
300
FIRE BALL
FIRE
SCORE+


SCORE:
230
MULTI
MULTI

ENHANCEMENTS

We have already added a lot of features most notable being POWERUPS and LEVEL EDITOR in our program.

But we want to enhance and make our project more creative and interesting in the future which we were not able to add due to time constraint by adding new features like:-

- LASER POWERUP
- FIRE GUN
- CATCH POWERUP
- Incorporating the feature of LIVES
- Adding INSTANT REPLAY

CONCLUSION

We have made a self-sufficient game in which we can design the levels for BRICKZ by another executable (LEVELER).

We have also learnt about topics like object oriented programming, classes and file handling etc. while making our project.

The project has helped us to increase our logic and to think about the programs in programming language.

It has also increased our thinking skills in terms of objects and object-oriented programming by making this project.

BIBLIOGRAPHY

- COMPUTER SCIENCE WITH C++ by Sumita Arora
- Turbo C++ help file.

FLOWCHART