LYNOX

- V. Sreeniketh

- R. Priyadharshan

- S. Adyant

- R. Srikrishnan

- K.P.S. Harishankar

ACKNOWLEDGEMENT

We would sincerely like to extend our heartfelt thanks to our Computer Science teachers Ms. Harifa Begum and Ms. Nirmala Devi for their guidance and for the help they rendered in completing our project.

INDEX

1. SYNOPSIS
2. PROJECT DESCRIPTION
3. PROGRAM LISTING
4. SAMPLE OUTPUT
5. BIBLIOGRAPHY

SYNOPSIS

BOGGLE:

Boggle is an extremely interesting word game where the player’s vocabulary is put to test by finding out as many words as possible from the “Boggle Grid” in two minutes under interesting constraints.

**GRAVITY:**

Gravity is a simulation of a planet orbiting around a star. You can move the sun around or change its mass and see the various conic sections come to life from Newton's law of gravity.

**ATARI BREAKOUT:**

**Atari Breakout is an exciting brick breaking game with subtle new features which will put the player’s reflexes and intuitive knowledge of motion to test.**

**MISSILES:**

Missiles is a simple arcade game where you control a plane and dodge it from a missile that follow your direction. The objective is to stay alive for the maximum amount of time. Though the game sounds simple, it can turn out to be extremely fun.

**OSU:**

OSU is a game which tests your mouse reflexes and helps your mouse accuracy by making you click in the randomly spawned circles within as much time as possible.

**MASTERMIND:**

**Mastermind is a test and battle of strategies and intelligence where the system/second player sets a 4-color code and the other player has 10 chances to find out the code given the number of correct colors and whether their positions are right or not.**

**PROJECT DESCRIPTION**

**BOGGLE:**

**Initially, the timer is set for two minutes and a pseudo-random grid is displayed. The player has to click on any particular letter to select it. Once a letter is selected, the player will be allowed to select only a letter that is adjacent (to the left, right, top, bottom and all 4 diagonals) to continue forming the word and in the end, has to click on the submit icon to submit the word. Then, the word will be evaluated and scores will be given based on the number of letters in the word.**

**ATARI BREAKOUT:**

**To start the game, the player must click the mouse in the direction he or she wants the ball to move. The initial velocity of the ball will be set based on the coordinates of the mouse click. Once the game starts, the mouse disappears but the user will be able to control the rectangular-bar/big-circlular-pad with the help of the mouse. The ball keeps accelerating and the game uses geometry to track the trajectory of the ball. Each brick is worth 50 points and once all the bricks are knocked off, a new layout is loaded. Once all the layouts are exhausted, a new game with a circular-pad replacing the traditional rectangular-bar is loaded. This offers a more complex and difficult level and this code uses collisions and normals to a circle to track the trajectory of the ball. At any point of time, the user can give a left click to restart the game and right click to quit.**

**OSU:**

This game is about moving your mouse and clicking the circles which are being spawned on the screen. The circle lasts for a certain amount of time after which it will be erased off the screen. There will be 60 circles and the player has to click every one of them to get the score until 60. Each click gives you a point.

**GRAVITY:**

This is a simulation of the earth orbiting the sun. It is a great tool that students can and should use to understand Kepler's laws and Newton's law of gravitation. You can move the sun or change the velocity of earth or mass of the sun and much more! Though the simulation is not representative of the actual earth and the sun, it follows Newton's laws. An additional feature that we provide is that the user can create their own gravitational law that needn't be the 1/r² rule! Isn't that amazing? You can **watch** the trajectory of the earth around the sun rather than solving a bunch of differential equations to find out the locus traced by the earth. You can **see** how different gravitational laws would have resulted in a universe that's quite different from what we have. In the future we wish to create more such simulators with many more features. We believe in transforming the way you learn. We make sure we teach you the right physics.

**MISSILES:**

**Missiles is a game developed by our team that aims at hand-eye coordination and reflexes. The game starts with a plane that you are allowed to control with the mouse. The plane follows the direction of the mouse pointer and a missile comes in from a random direction and it always follows the plane. The player must somehow escape from the missile and the objective of the game is to survive for the maximum amount of time. This game improves concentration and focus. This can be used as a warmup for something more intense.**

**MASTERMIND:**

**Mastermind is a test and battle of strategies, logic and deductive reasoning. There are 2 players (or a player and a machine). In single player, the player is the codebreaker and the machine is the codemaker. In double player, each player is either a codemaker or a codebreaker. The codemaker sets up a four-color code and hides it from the codebreaker.** The codebreaker tries to guess the pattern, in both order and color, within ten turns. Each guess is made by placing a row of code pegs on the decoding board. Once placed, the codemaker provides feedback by giving the number of whites and reds. A red is placed for each code from the guess which is correct in both color and position. A white indicates the existence of a correct color placed in the wrong position. If there are duplicate colours in the guess, they cannot all be awarded a key unless they correspond to the same number of duplicate colours in the hidden code. Once feedback is provided, another guess is made; guesses and feedback continue to alternate until either the codebreaker guesses correctly, or ten incorrect guesses are made.

**PROGRAM LISTING**

**BOGGLE**

**BOGGLE.CPP**

#include<iostream.h>

#include<conio.h>

#include<stdlib.h>

#include<math.h>

#include<fstream.h>

#include<stdio.h>

#include<string.h>

char a[4][4][6];

char b[4][4];

char bog[18];

void initialise()

{

char st[7];

ifstream fin("e:\\boggle.txt",ios::in);

for(int i=0;i<4;i++)

{

for(int j=0;j<4;j++)

{

fin>>st;

for(int k=0;k<6;k++)

{

a[i][j][k]=st[k];

}

}

}

fin.close();

}

void copy(char a[],char b[])

{

for(int i=0;i<6;i++)

{

a[i]=b[i];

}

}

void swap(char a[],char b[])

{

char temp[6];

copy(temp,a);

copy(a,b);

copy(b,temp);

}

void shuffle()

{

int a1,b1,a2,b2;

randomize();

for(int n=0;n<50;n++)

{

a1=random(4);

a2=random(4);

b1=random(4);

b2=random(4);

swap(a[a1][b1],a[a2][b2]);

}

}

void setboard()

{

int x;

randomize();

for(int i=0;i<4;i++)

{

for(int j=0;j<4;j++)

{

x=random(6);

b[i][j]= a[i][j][x];

}

}

}

void convert()

{

int k=0;

for(int i=0;i<4;i++)

{

for(int j = 0 ;j<4 ; j++)

{

bog[k]=b[i][j];

k++;

}

}

bog[k]='\0';

}

**GRID.CPP**

#include<iostream.h>

#include<graphics.h>

#include<stdio.h>

#include<conio.h>

#include<stdlib.h>

#include<fstream.h>

#include<string.h>

#include<time.h>

#include<dos.h>

#include<ctype.h>

#include"mouse.cpp";

#include"BOGGLE.cpp";

//mouse m;

char word[17];

int score,yc;

float x,y;

void set\_timer()

{

struct time t;

int h,m,s;

gettime(&t);

h=t.ti\_hour;

m=t.ti\_min;

s=t.ti\_sec;

x=s+60\*m+3600\*h;

}

int timer()

{

struct time t;

int h,m,s;

gettime(&t);

h=t.ti\_hour;

m=t.ti\_min;

s=t.ti\_sec;

float y=s+60\*m+3600\*h;

if((y-x)>=120) return 0;

setfillstyle(1,0);

bar(200,400,350,450);

settextstyle(7,0,2);

char kh[9];

int ph= 120-(y-x);

itoa(ph,kh,10);

outtextxy(200,400,kh);

return 1;

}

void grid()

{

int x=100,y=100;

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

{ line(x,y,x+200,y);

y=y+50;

}

x=100;

y=100;

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

{ line(x,y,x,y+200);

x=x+50;

}

}

void fill(char \*a)

{

char c[2];

setcolor(RED);

settextstyle(7,0,4);

c[1]=0;

for(int j=100,k=0;j<=250;j+=50)

{

for(int i=100;i<=250;i+=50)

{

c[0] = a[k];

outtextxy(i+12,j+12,c);

k++;

}

}

}

int get\_no(int x, int y)

{

int a=((x>100)&&(x<150));

int b=((x>150)&&(x<200));

int c=((x>200)&&(x<250));

int d=((x>250)&&(x<300));

int e=((y>100)&&(y<150));

int f=((y>150)&&(y<200));

int g=((y>200)&&(y<250));

int h=((y>250)&&(y<300));

int i = 90;

if(a&&e) i=0;

else if(b&&e) i=1;

else if(c&&e) i=2;

else if(d&&e) i=3;

else if(a&&f) i=4;

else if(b&&f) i=5;

else if(c&&f) i=6;

else if(d&&f) i=7;

else if(a&&g) i=8;

else if(b&&g) i=9;

else if(c&&g) i=10;

else if(d&&g) i=11;

else if(a&&h) i=12;

else if(b&&h) i=13;

else if(c&&h) i=14;

else if(d&&h) i=15;

else if(((x>175)&&(x<225))&&((y>350)&&(y<400))) i=16;

return i;

}

int check(int in, int &k, int &i)

{

if(k%4==0){ if((in==(k-4))||(in==(k-3))||(in==(k+1))||(in==(k+4))||(in==(k+5))){ word[i] = bog[in]; k=in;i++;return 1;}}

else if(k%4==3){ if((in==(k-5))||(in==(k-4))||(in==(k-1))||(in==(k+3))||(in==(k+4))){ word[i] = bog[in]; k=in; i++;return 1;}}

else if(!((in<(k-5))||(in>(k+5))||(in==(k-2))||(in==(k+2)))){ word[i] = bog[in]; k=in;i++;return 1;}

return 0;

}

void update\_score(char \*a)

{

int l=strlen(a);

if(l>1)

score=score+l\*10;

}

void split()

{

char s[100],a[100]= "A.txt";

fstream f("dict.txt",ios::in);

for(;a[0]>='A'&&a[0]<='Z';)

{

fstream g(a,ios::out);

f.getline(s,80);

s[0]=toupper(s[0]);

while(s[0]==a[0]&&!f.eof())

{

g<<s<<"\n";

f.getline(s,80);

s[0]=toupper(s[0]);

}

g.close();

a[0]++;

}

f.close();

}

int search(char a[])

{

char s[200];

char fn[] ="\*.txt";

fn[0] = a[0];

ifstream f(fn,ios::in);

long int flag=0;

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

f.getline(s,100,'\n');

while(!f.eof()&& flag==0)

{

if ( strcmpi(s,a) == 0) flag = 1;

f.getline(s,100,'\n');

}

f.close();

if(flag){ update\_score(a); return 2; }

else return 4;

}

void ret()

{

int i=0,k,flag=1;

while(flag)

{

do

{

delay(150);

m.get\_status();

cx = m.regs.x.cx;

cy = m.regs.x.dx;

}while(!(m.left\_click()));

int in=get\_no(cx,cy);

if(in==90) continue;

if(in!=16)

{

int p;

if(i==0){ word[i]=bog[in]; k=in; i++; }

else if(i!=0){ p= check(in,k,i);}

if(!p) continue;

settextstyle(7,0,2);

setcolor(WHITE);

outtextxy(402,yc,word);

}

else

{

flag = 0;

word[i]='\0';

int col=search(word);

settextstyle(7,0,2);

setcolor(col);

outtextxy(402,yc,word);

yc+=20;

for(int o=0;o<17;word[o]='\0',o++);

}

}

}

void disp\_score()

{

cleardevice();

char b[10];

itoa(score,b,10);

settextstyle(7,0,7);

outtextxy(100,100,"YOUR SCORE IS:");

outtextxy(100,300,b);

}

void main()

{

int a,b;

a=9;

b=2;

initgraph(&a,&b,"C:\\TC\\BGI");

cleardevice();

line(400,0,400,479);

setcolor(GREEN);

settextstyle(4,0,5);

outtextxy(120,0,"BOGGLE");

settextstyle(7,0,3);

setcolor(WHITE);

outtextxy(520,0,"WORDS");

initialise();

shuffle();

setboard();

convert();

split();

grid();

setfillstyle(1,4);

bar(175,350,225,400);

fill(bog);

score=0;

set\_timer();

yc=20;

while(timer())

{

m.show\_mouse();

ret();

m.hide\_mouse();

}

disp\_score();

getch();

cleardevice();

closegraph();

}

**GRAVITY**

**GRAVITY.CPP**

#include<iostream.h>

#include<dos.h>

#include<conio.h>

#include<math.h>

#include<fstream.h>

#include<stdio.h>

#include<string.h>

#include<graphics.h>

#include<stdlib.h>

#include "mouse.cpp"

float g=100,power;

const float dt=0.0025;

class vector

{

public:

double x,y;

};

float mod(vector a)

{

float m;

m=sqrt((a.x\*a.x)+(a.y\*a.y));

return m;

}

vector sunPos;

double sunMass=100000,earthMass=1,sunRadius=20;

class planet

{

public:

vector vel,acc,pos;

vector rad;

public:

planet()

{

pos.x=500;

pos.y=240;

vel.x=0;

vel.y=-150;

}

vector calcforce()

{

vector force;

rad.x=pos.x-sunPos.x;

rad.y=pos.y-sunPos.y;

force.x=((g\*sunMass\*earthMass)/pow(mod(rad),1-power))\*rad.x;

force.y=-((g\*sunMass\*earthMass)/pow(mod(rad),1-power))\*rad.y;

return force;

}

void updateVel()

{

acc.x=(calcforce().x)/earthMass;

acc.y=(calcforce().y)/earthMass;

vel.x=vel.x+acc.x\*dt;

vel.y=vel.y+acc.y\*dt;

}

void updatePos()

{

pos.x=pos.x-vel.x\*dt;

pos.y=pos.y+vel.y\*dt;

}

int ifEscape()

{

double te;

if(power!=-1)

te=(g\*sunMass\*earthMass/pow(mod(rad),1+power)/(1+power))-(0.5\*earthMass\*pow(mod(vel),2));

else

te=(g\*sunMass\*earthMass\*log(mod(rad)))-(0.5\*earthMass\*pow(mod(vel),2));

if((te>0)&&(pos.x>640||pos.x<0)&&(pos.y>480||pos.y<0))return 1;

else return 0;

}

void draw()

{

setcolor(BLUE);

setfillstyle(1,BLUE);

pieslice(pos.x,pos.y,0,360,7);

}

};

planet earth;

void arrow(int x,int y, vector arr,int color)

{ setcolor(color);

arr.x = arr.x/10.0 ;

arr.y = arr.y/10.0 ;

if(arr.x!=0)

{ line (x,y,x-arr.x,y+arr.y);

}

}

int check\_hit()

{

double dis ;

dis = sqrt((sunPos.x - earth.pos.x)\*(sunPos.x - earth.pos.x)+(sunPos.y - earth.pos.y)\*(sunPos.y - earth.pos.y)) ;

if ( dis >= 7+sunRadius ) return 1 ;

else return 0 ;

}

int check\_hit(int x, int y)

{

double dis ;

dis = sqrt((sunPos.x - x)\*(sunPos.x - x)+(sunPos.y - y)\*(sunPos.y - y)) ;

if ( dis >= 20 ) return 0 ;

else return 1 ;

}

void close\_button(int x, int y, int r)

{

setcolor(RED);

pieslice(x,y,0,360,r) ;

line(x-r/2,y-r/2,x+r/2,y+r/2);

line(x+r/2,y+r/2,x-r/2,y-r/2);

}

void update\_sunMass()

{

setcolor(11);

line(600,0,600,480) ;

line(620,200,620,300);

outtextxy(605,150,"Sun");

outtextxy(605,170,"Mass:");

if(m.left\_click())

{

if(cx>=600)

{

if(cy>=200 && cy<=300)

{

circle(620,cy,5);

sunMass=sunMass+float(sunMass)\*(250-cy)/1000.0 ;

// sunRadius=sunRadius-sunRadius\*(cy-250)/1000.0 ;

}

}

}

}

void inline draw\_sun(int color)

{

setcolor(color);

setfillstyle(1,color);

pieslice(sunPos.x,sunPos.y,0,360,sunRadius);

}

void main()

{

int gd=11,gm=2;

double days = 0 ;

char st[20],v[20];

sunPos.x=320;

sunPos.y=240;

vector force;

m.show\_mouse();

cx=100;cy=100;

cin >> power ;

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

//while (check\_hit())

while(1)

{

cx=100; cy=100;

arrow(earth.pos.x,earth.pos.y,earth.vel,RED);

arrow(earth.pos.x,earth.pos.y,earth.acc,GREEN);

close\_button(5,5,5);

draw\_sun(YELLOW);

earth.draw();

force= earth.calcforce();

delay(1);

setfillstyle(1,BLACK);

setcolor(BLACK);

circle(earth.pos.x,earth.pos.y,7);

arrow(earth.pos.x,earth.pos.y,earth.vel,BLACK);

arrow(earth.pos.x,earth.pos.y,earth.acc,BLACK);

earth.updatePos();

earth.updateVel();

m.show\_mouse();

m.get\_status();

cx = m.regs.x.cx;

cy = m.regs.x.dx;

update\_sunMass();

setcolor(BLACK) ;

if (m.left\_click()) // Left click

{

if(check\_hit(cx,cy))

{

m.hide\_mouse();

pieslice(sunPos.x, sunPos.y,0,360, sunRadius) ;

sunPos.x = cx ;

sunPos.y = cy ;

}

}

if (m.left\_click()&&(cy < 10) && (cx<10)) break ;

setcolor(WHITE);

itoa(days,st,10) ;

outtextxy(460,460,"Number of days:");

outtextxy(10,400,"Velocity");

itoa(mod(earth.vel),v,10);

outtextxy(80,400,v);

outtextxy(600,460,st);

days+=0.1;

delay(2);

setcolor(BLACK);

bar(600,460,640,480);

outtextxy(80,400,"ÛÛÛÛ");

setcolor(CYAN);

if(earth.ifEscape())outtextxy(10,20,"Earth has escaped from Sun's gravitational field");

}

closegraph();

//getch();

}

**OSU**

**GRAPHICS.CPP**

#include<iostream.h>

#include<stdio.h>

#include<graphics.h>

#include<conio.h>

#include<stdlib.h>

#include<dos.h>

#include<math.h>

struct REGS in,out;

int callmouse()

{

in.x.ax=1;

int86(51,&in,&out);

return 1;

}

void restrictmouseptr(int x1,int y1,int x2,int y2)

{

in.x.ax=7;

in.x.cx=x1;

in.x.dx=x2;

int86(51,&in,&out);

in.x.ax=8;

in.x.cx=y1;

in.x.dx=y2;

int86(51,&in,&out);

}

void mouseposi(int &xpos,int &ypos,int &click)

{

in.x.ax=3;

int86(51,&in,&out);

click=out.x.bx;

xpos=out.x.cx;

ypos=out.x.dx;

}

int mousehide()

{

in.x.ax=2;

int86(51,&in,&out);

return 1;

}

void initgraphics()

{

int gdriver = DETECT, gmode, errorcode;

/\* initialize graphics mode \*/

initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");

setcolor(GREEN);

setbkcolor(BLACK);

setfillstyle(SOLID\_FILL,GREEN);

restrictmouseptr(0,0,639,479);

callmouse();

}

void closegraphics()

{

closegraph();

}

void start\_menu()

{

int gdriver = DETECT, gmode, errorcode;

/\* initialize graphics mode \*/

initgraph(&gdriver, &gmode, "C:\\TURBOC3\\BGI");

circle(320,240,80);

outtextxy(320,240,"ZASU!");

outtextxy(320,260,"Click the Circles");

callmouse();

int x,y,c;

mouseposi(x,y,c);

double d;

d=pow(x-320,2)+pow(y-240,2);

//cout << d<< radius;

if(d<=pow(80,2))

{

closegraphics();

}

}

class circ

{

int centrex;

int centrey;

int radius;

public:

circ(int x1,int y1,int radius1)

{

centrex=x1;centrey=y1;radius=radius1;

}

int isInside(int x1,int y1)

{

//getch();

double d;

d=pow(x1-centrex,2)+pow(y1-centrey,2);

//cout << d<< radius;

if(d<=pow(radius,2))

{

return 1;

}

else return -1;

}

int getkey(int a)

{

a=getch();

if(a=='z'||a=='Z'||a=='x'||a=='X')

{

return 1;

}

else return 0;

}

void score(int click)

{

int scorex,scorey;

char clicks[20];

scorex=getmaxx();

scorey=getmaxy();

itoa(click,clicks,10);

outtextxy(scorex-20,scorey-20, clicks);

}

int print()

{

int x,y,cl,clicks=0;

randomize();

for(int i=0;i<60;i++)

{

score(clicks);

centrex=random(589);

centrey=random(429);

setcolor(random(15)+1);

circle(centrex,centrey,radius);

setfillstyle(EMPTY\_FILL,BLACK);

float n=0.9;

for(double j=0;j<682767/n;j++)

{

mouseposi(x,y,cl);

if(cl==1 && isInside(x,y)==1)

{

if(clicks>=5)

{

n=n+10.0;

}

if(clicks>=10)

{

n=n+1.0;

}

if(clicks>=15)

{

n=n+1.0;

}

if(clicks>=20)

{

n=n+0.5;

}

clicks++;

sound(1000);

score(clicks);

delay(30);

nosound();

break;

}

}

cleardevice();

}

mousehide();

closegraph();

return clicks;

}

};

void main()

{

clrscr();

randomize();

start\_menu();

cout<<""<<endl;

cout<<"MADE BY ADYANT.S"<<endl;

cout<<" INPUT y AND CLICK ENTER"<<endl;

int \*y;

char \*l;

cin>>l;

if(l=="y")clrscr; \*/

circ ob(150,30,20);

initgraphics();

int clicks=ob.print();

cout<<"Number of clicks is:"<<clicks;

getch();

clrscr();

getch();

closegraphics();

}

**MISSILES**

**MISSILES.CPP**

#include<iostream.h>

#include<math.h>

#include<dos.h>

#include<stdlib.h>

#include<conio.h>

#include<graphics.h>

#include "mouse.cpp"

float dt=0.3;

class vector

{

public:

double x,y ;

};

void unit\_vec(vector &a)

{

double mag;

mag = sqrt((a.x\*a.x)+(a.y\*a.y)) ;

a.x = a.x / mag ;

a.y = a.y / mag;

}

vector planePos ;

class planes

{

public:

vector pos,vel;

planes()

{

pos.x = 320 ;

pos.y = 240 ;

vel.x = 0 ;

vel.y = 10 ;

}

void updatepos2()

{

pos.x = pos.x + vel.x\*dt ;

pos.y = pos.y + vel.y\*dt ;

}

void updatevel2()

{

m.show\_mouse();

m.get\_status();

cx = m.regs.x.cx;

cy = m.regs.x.dx;

vector plane\_mouse;

plane\_mouse.x = cx - pos.x ;

plane\_mouse.y = cy - pos.y ;

unit\_vec( plane\_mouse ) ;

vel.x = 6\*plane\_mouse.x ;

vel.y = 6\*plane\_mouse.y ;

}

}plane;

class missile : public planes

{

public:

missile();

void update\_vel()

{

vector rpos ;

rpos.x = plane.pos.x - pos.x ;

rpos.y = plane.pos.y - pos.y ;

unit\_vec(rpos);

vel.x=5\*rpos.x+random(25)/10.0 ;

vel.y=5\*rpos.y+random(25)/10.0 ;

}

void update\_pos()

{

pos.x = pos.x + vel.x\*dt ;

pos.y = pos.y + vel.y\*dt ;

}

};

missile::missile()

{

randomize();

pos.x = random(640\*3)-640;

switch(random(2))

{

case 0: pos.y = -random(480) ; break;

case 1: pos.y = random(480)+480; break;

}

}

int check\_hit(missile ob)

{

double dis ;

dis = sqrt((plane.pos.x - ob.pos.x)\*(plane.pos.x - ob.pos.x)+(plane.pos.y - ob.pos.y)\*(plane.pos.y - ob.pos.y)) ;

if ( dis <= 15 ) return 1 ;

else return 0 ;

}

void main()

{

int gd=DETECT, gm=2 ;

int z=1;

planePos.x = 0 ;

planePos.y = 240 ;

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

randomize();

missile ob;

cx=320; cy=240;

while( 1 )

{

setcolor(YELLOW);

putpixel(cx,cy,5);

circle(plane.pos.x, plane.pos.y, 10);

s etcolor(RED);

circle(ob.pos.x, ob.pos.y, 6) ;

delay(10);

setcolor(BLACK) ;

circle(ob.pos.x, ob.pos.y, 6) ;

circle(plane.pos.x, plane.pos.y, 10);

setcolor(YELLOW) ;

ob.update\_vel();

ob.update\_pos();

plane.updatepos2();

plane.updatevel2();

if( check\_hit(ob) )

{ gotoxy(20,20);

cout<<" game over";

break;

}

}

getch();

closegraph();

}

**MASTERMIND**

**MAST.CPP**

#include<iostream.h>

#include<graphics.h>

#include<conio.h>

#include<stdlib.h>

#include"mouse.cpp";

int master[11][4], comb[16], user[16];

int col[8] = { 1,2,3,4,5,6,14,15 };

int mast[4];

char message[] = "0W,0R";

int page\_1()

{

settextstyle(7,0,6);

setcolor(YELLOW);

outtextxy(150,20,"MASTERMIND");

setfillstyle(1,YELLOW);

bar(256,230,384,262);

bar(256,300,384,332);

setcolor(BLACK);

settextstyle(12,0,8);

outtextxy(265,245,"SINGLE PLAYER");

outtextxy(265,315,"DOUBLE PLAYER");

int flag=0;

do

{

m.show\_mouse();

m.get\_status();

cx=m.regs.x.cx;

cy=m.regs.x.dx;

if(m.left\_click())

{

int c=(cx>256&&cx<384);

int d=(cy>230&&cy<262);

int e=(cy>300&&cy<332);

if(cx>=256&&d) flag =1;

else if(c&&e) flag=2;

delay(10);

}

}while(flag==0);

m.hide\_mouse();

if(flag==1) return 1;

else if(flag==2) return 2;

}

void board()

{

cleardevice();

setfillstyle(1,12);

setcolor(12);

pieslice(610,20,0,360,10);

setfillstyle(1,8);

bar(45,55,595,255);

setfillstyle(1,7);

bar(45,256,595,295);

bar(45,15,595,54);

setcolor(BLACK);

for(int j=55;j<=255;j+=50)

line(45,j,595,j);

for(int i=45;i<=595;i+=50)

line(i,15,i,295);

setfillstyle(1,BLACK);

for(j=80;j<=230;j+=50)

for(i=70;i<=570;i+=50)

pieslice(i,j,0,360,15);

char s[2]="1";

for(i=1,j=70;i<=9;i++)

{

settextstyle(12,0,5);

outtextxy(j,30,s);

s[0]++;

j+=50;

}

outtextxy(520,30,"10");

setfillstyle(1,1);

setcolor(1);

pieslice(22,40,0,360,15);

setfillstyle(1,2);

setcolor(2);

pieslice(22,75,0,360,15);

setfillstyle(1,3);

setcolor(3);

pieslice(22,110,0,360,15);

setfillstyle(1,4);

setcolor(4);

pieslice(22,145,0,360,15);

setfillstyle(1,5);

setcolor(5);

pieslice(22,180,0,360,15);

setfillstyle(1,6);

setcolor(6);

pieslice(22,215,0,360,15);

setfillstyle(1,14);

setcolor(14);

pieslice(22,250,0,360,15);

setfillstyle(1,15);

setcolor(15);

pieslice(22,285,0,360,15);

settextstyle(7,0,4);

setcolor(RED);

outtextxy(210,360,"MASTERMIND");

}

void choose\_one()

{

for(int ik=0;ik<16;comb[ik]=0,ik++);

randomize();

for(int i=0; i<4; i++)

{

int k= random(8);

master[10][i] = col[k] ;

comb[col[k]]++;

}

}

void choose\_two()

{

cleardevice();

for(int ik=0;ik<16;comb[ik]=0,ik++);

setfillstyle(1,1);

setcolor(1);

pieslice(22,40,0,360,15);

setfillstyle(1,2);

setcolor(2);

pieslice(22,75,0,360,15);

setfillstyle(1,3);

setcolor(3);

pieslice(22,110,0,360,15);

setfillstyle(1,4);

setcolor(4);

pieslice(22,145,0,360,15);

setfillstyle(1,5);

setcolor(5);

pieslice(22,180,0,360,15);

setfillstyle(1,6);

setcolor(6);

pieslice(22,215,0,360,15);

setfillstyle(1,14);

setcolor(14);

pieslice(22,250,0,360,15);

setfillstyle(1,15);

setcolor(15);

pieslice(22,285,0,360,15);

settextstyle(7,0,2);

setcolor(WHITE);

outtextxy(210,120,"CHOOSE YOUR COMBINATION");

int flag,k=0;

for(int i=250;i<=355;i+=35,k++)

{

flag=0;

do

{

m.show\_mouse();

m.get\_status();

cx=m.regs.x.cx;

cy=m.regs.x.dx;

if(m.left\_click())

{

if(getpixel(cx+2,cy)) flag=getpixel(cx+2,cy);

}

}while(flag==0);

comb[flag]++;

master[10][k] = flag;

setcolor(flag);

setfillstyle(1,flag);

pieslice(220,i,0,360,15);

delay(800);

}

m.hide\_mouse();

getch();

}

int check(int i,int x)

{

message[0] = message[3] = 0;

for( int j = 0 ; j < 16 ; j++ )

{

if(user[j]!=0&&comb[j]!=0)

{

int t;

if(user[j]<=comb[j]) t = user[j];

else t = comb[j];

message[0]+=t;

}

}

for( j = 0 ; j < 4 ; j++ )

{

if(master[i][j] == master[10][j]) { message[0]--; message[3]++; }

}

message[0]+=48;

message[3]+=48;

settextstyle(13,0,3);

setcolor(BLACK);

outtextxy(x+5,260,message);

if(message[3]=='4') return 1;

return 0;

}

void disp\_page(int c)

{

cleardevice();

if(c==1)

{

settextstyle(7,0,3);

setcolor(GREEN);

outtextxy(40,150,"CONGRATULATIONS!");

outtextxy(100,200,"YOU HAVE GOT THE RIGHT COMBINATION!");

for(int i=250,j=0;i<=355;i+=35,j++)

{

setcolor(master[10][j]);

setfillstyle(1,master[10][j]);

pieslice(220,i,0,360,15);

}

}

else if(c==0)

{

settextstyle(7,0,3);

setcolor(RED);

outtextxy(40,150,"OOPS!");

outtextxy(60,200,"YOU DID NOT FIND THE RIGHT COMBINATION");

outtextxy(20,250,"THE RIGHT COMBINATION IS:");

for(int i=265,j=0;i<=370;i+=35,j++)

{

setcolor(master[10][j]);

setfillstyle(1,master[10][j]);

pieslice(440,i,0,360,15);

}

}

getch();

closegraph();

exit(0);

}

int round(int x,int i)

{

int flag,k=0;

for(int p=0;p<16;user[p]=0,p++);

for(int j=80;j<=230;j+=50,k++)

{

flag=0;

do

{

m.show\_mouse();

m.get\_status();

cx=m.regs.x.cx;

cy=m.regs.x.dx;

if(m.left\_click())

{

if(getpixel(cx+2,cy)) flag=getpixel(cx+2,cy);

if(flag==12) disp\_page(0);

if(flag==8||flag==7) flag=0;

}

}while(flag==0);

user[flag]++;

master[i][k] = flag;

setcolor(flag);

setfillstyle(1,flag);

pieslice(x,j,0,360,15);

delay(800);

}

getch();

if(check(i,x-25)) { disp\_page(1); return 0;}

else if(i==9) { disp\_page(0); return 0;}

return 1;

}

void play()

{

m.show\_mouse();

int x=70,i=0;

while(round(x,i))

{

x+=50;

i++;

}

}

void one()

{

choose\_one();

board();

play();

}

void two()

{

choose\_two();

board();

play();

}

void main()

{

int a,b;

a=9;

b=2;

initgraph(&a,&b,"C:\\TC\\BGI");

cleardevice();

if(page\_1()==1) one();

else if(page\_1()==2) two();

getch();

closegraph();

}

**ATARI BREAKOUT**

**ATARI.CPP**

#include<graphics.h>

#include<alloc.h>

#include<math.h>

#include<dos.h>

#include<conio.h>

#include<stdlib.h>

#include"c:\tc\bin\mouse.cpp";

struct coord

{

int x1,y1,x2,y2;

};

class vector

{

public:

double x,y;

double mod(vector v);

void unit(vector v);

}acc={10,15};

double vector::mod(vector v)

{

double mod;

mod = sqrt(v.x\*v.x + v.y\*v.y);

return mod;

}

void vector::unit(vector v)

{

x = v.x / mod(v);

y = v.y / mod(v);

}

vector vel,pos,hit,barr,circ,normal;

coord check[400];

int number, score;

double dt=0.005;

void \*im;

void place\_bar()

{

setfillstyle(4,4);

setcolor(RED);

bar(273,468,369,480);

barr.x=273;

barr.y=468;

}

void placeball\_bar()

{

place\_bar();

setcolor(RED);

setfillstyle(1,RED);

circle(321,462,5);

floodfill(321,462,RED);

pos.x=321;

pos.y=462;

int size;

size = imagesize(316,457,326,467);

im = (char\*)malloc(size);

getimage(316,457,326,467,im);

}

void place\_circpad()

{

setfillstyle(4,RED);

setcolor(RED);

circle(320,459,20);

circ.x=320;

circ.y=459;

}

void placeball\_circpad()

{

place\_circpad();

setcolor(RED);

setfillstyle(1,RED);

circle(320,433,5);

floodfill(320,433,RED);

pos.x=320;

pos.y=433;

int size;

size = imagesize(315,428,325,438);

im = (char\*)malloc(size);

getimage(315,428,325,438,im);

}

void layout\_1()

{

int s=0;

number=80; setfillstyle(4,4);

for(int y=0;y<=144;y+=24)

{

for(int x=0,i=s;x<=576;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s=s+3;

}

}

void layout\_2()

{

int s= 4;

number=170; setfillstyle(4,4);

for(int y=0;y<=120;y+=24)

{

for(int x=0,i=s;x<=576;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

}

int a=64;

for(y=144;y<=216;y+=24)

{

for(int x=0,i=s;x<=256-a;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=8;

a+=64;

}

a=64;

for(y=144;y<=216;y+=24)

{

for(int x=640,i=s;x>=384+a;x-=64)

{

if(i%16==0)i=11;

setfillstyle(1,(i%16));

bar(x-64,y,x,y+24);

number++;

i+=5;

}

s+=4;

a+=64;

}

}

void layout\_3()

{

int s= 3;

number=251; setfillstyle(4,4);

for(int y=0;y<=108;y+=24)

{

for(int x=0,i=s;x<=576;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

}

int a=64;

for(y=120;y<=192;y+=24)

{

for(int x=a,i=s;x<=576-a;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

a+=64;

s+=7;

}

setfillstyle(1,2);

bar(289,217,353,241);

}

void layout\_4()

{

int s= 2;

number=345; setfillstyle(4,4);

for(int y=0;y<=120;y+=24)

{

for(int x=0,i=s;x<=576;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

}

for(y=144;y<=216;y+=24)

{

for(int x=0,i=s;x<=256;x+=128)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=6;

}

for(y=144;y<=216;y+=24)

{

for(int x=640,i=s;x>=384;x-=128)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x-64,y,x,y+24);

number++;

i+=6;

}

s+=6;

}

}

void layout\_5()

{

int s=2,a=0;

number=405; setfillstyle(4,4);

for(int y=0;y<=216;y+=24)

{

for(int x=0,i=s;x<=576-a;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

a+=64;

}

}

void layout\_6()

{

int s=2,a=64;

number=465; setfillstyle(4,4);

for(int y=0;y<=216;y+=24)

{

for(int x=640,i=s;x>=a;x-=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x-64,y,x,y+24);

number++;

i+=6;

}

s+=7;

a+=64;

}

}

void layout\_7()

{

int s=3,a=320;

number=510; setfillstyle(4,4);

for(int y=48;y<=120;y+=24)

{

for(int x=320,i=s;x>=a;x-=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x-64,y,x,y+24);

number++;

i+=6;

}

s+=7;

a-=64;

}

for( y=144,a=192;y<=192;y+=24)

{

for(int x=320,i=s;x>=a;x-=64)

{

if(i%16==0)i=11;

setfillstyle(1,(i%16));

bar(x-64,y,x,y+24);

number++;

i+=6;

}

s+=7;

a+=64;

}

for(y=48,a=320;y<=120;y+=24)

{

for(int x=320,i=s;x<=a;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

a+=64;

}

for( y=144,a=448;y<=192;y+=24)

{

for(int x=320,i=s;x<=a;x+=64)

{

if(i%16==0)i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

a-=64;

}

setfillstyle(1,2);

bar(289,217,353,241);

setfillstyle(1,11);

bar(289,24,353,47);

}

void layout\_8()

{

int s=3,a=0;

number=590; setfillstyle(4,4);

for(int y=0;y<=96;y+=24)

{

for(int x=0,i=s;x<=a;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

a+=64;

}

for(y=0,a=576;y<=96;y+=24)

{

for(int x=576,i=s;x>=a;x-=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s+=7;

a-=64;

}

for( y=120;y<=192;y+=24)

{

for(int x=0,i=s;x<=576;x+=64)

{

if(i%16==0) i=11;

setfillstyle(1,(i%16));

bar(x,y,x+64,y+24);

number++;

i+=6;

}

s=s+3;

}

}

void start\_1()

{

mouse m;

do

{

m.show\_mouse();

m.get\_status();

cx=m.regs.x.cx;

cy=m.regs.x.dx;

}while(!(m.left\_click()));

{

vel.x= cx - 321;

vel.y= cy - 462;

vel.unit(vel);

vel.x\*=500;

vel.y\*=500;

m.hide\_mouse();

}

}

void start\_2()

{

mouse m;

do

{

m.show\_mouse();

m.get\_status();

cx=m.regs.x.cx;

cy=m.regs.x.dx;

}while(!(m.left\_click()));

{

vel.x= cx - 320;

vel.y= cy - 433;

vel.unit(vel);

vel.x\*=500;

vel.y\*=500;

m.hide\_mouse();

}

}

void update\_pos()

{

pos.x = pos.x + vel.x\*dt;

pos.y = pos.y + vel.y\*dt;

}

void update\_vel()

{

vel.x = vel.x + acc.x\*dt;

vel.y = vel.y + acc.y\*dt;

}

int check\_hit\_block()

{

int x=pos.x, y=pos.y;

if(getpixel(x+6,y)!=0) { hit.x = x+6; hit.y = y; return 1; }

else if(getpixel(x-6,y)!=0) { hit.x = x-6; hit.y = y; return 1; }

else if(getpixel(x,y+6)!=0) { hit.x = x; hit.y = y+6; return 1; }

else if(getpixel(x,y-6)!=0) { hit.x = x; hit.y = y-6; return 1; }

else if(getpixel(x-6,y-6)!=0) { hit.x = x-6; hit.y = y-6; return 1; }

else if(getpixel(x+6,y-6)!=0) { hit.x = x+6; hit.y = y-6; return 1; }

else if(getpixel(x-6,y+6)!=0) { hit.x = x-6; hit.y = y+6; return 1; }

else if(getpixel(x+6,y+6)!=0) { hit.x = x+6; hit.y = y+6; return 1; }

return 0;

}

void update\_vel\_coll(int i)

{

if(i==1) vel.y = -vel.y;

else if(i==2) vel.x = -vel.x;

}

void delete\_block(int x, int y)

{

score+=50;

int x1 = x, x2 = x, y1 = y, y2 = y;

while((x1%64)!=0&&(x2%64)!=0)

{

x1++;

x2--;

}

while((y1%24)!=0&&(y2%24)!=0)

{

y1++;

y2--;

}

int a,b,c,d;

if(x1==x2)

{

if(vel.x>0) { a = x1 ; c = x1 ; }

else { a = x1 - 64 ; c = x1 ; }

}

else

{

if((x1%64)==0) { a = x1 - 64 ; c = x1 ; }

else { a = x2 ; c = x2 ; }

}

if(y1==y2)

{

if(vel.y>0) { b = y1 ; d = y1 ; }

else { b = y1 - 24 ; d = y1 ; }

}

else

{

if((y1%24)==0) { b = y1 - 24 ; d = y1 ; }

else { b = y2 ; d = y2 ; }

}

setfillstyle(1,0);

bar(a,b,a+64,b+24);

if(fabs(x-c) > fabs(y-d)) update\_vel\_coll(1);

else if(fabs(x-c) < fabs(y-d)) update\_vel\_coll(2);

else

{

if(fabs(x-pos.x) > fabs(y-pos.y)) update\_vel\_coll(2);

else update\_vel\_coll(1);

}

}

void controls\_1()

{

m.get\_status();

cx = m.regs.x.cx ;

setcolor(RED);

setfillstyle(4,4);

bar(cx,468,cx+96,480);

barr.x = cx;

barr.y = 468 ;

}

void controls\_2()

{

m.get\_status();

cx = m.regs.x.cx ;

setfillstyle(8,RED);

setcolor(RED);

circle(cx,459,20);

circ.x = cx;

circ.y = 459 ;

}

void check\_corner()

{

int x1 = pos.x + 5, x2 = pos.x - 5, y1 = pos.y - 5;

if(x1>=639||x2<=1) update\_vel\_coll(2);

else if(y1<=1) update\_vel\_coll(1);

}

int not\_out()

{

if(pos.y>=480) return 0;

return 1;

}

void display\_score()

{

cleardevice();

settextstyle(7,0,3);

outtextxy(300,240, "Your Score Is: ");

char a[10];

itoa(score,a,10);

outtextxy(540,240,a);

outtextxy(200,360,"Press Any Key To Continue");

getch();

}

void check\_hit\_bar()

{

int x=pos.x,y=pos.y;

if((x>barr.x-5)&&(x<(barr.x+101))&&(y>462)) vel.y = -vel.y;

}

double dot(vector a, vector b)

{

double pdt;

pdt = a.x\*b.x + a.y\*b.y ;

return pdt;

}

void check\_hit\_circpad()

{

normal.x = pos.x-circ.x;

normal.y = pos.y-circ.y;

double dist= normal.mod(normal);

if(dist<=25)

{

normal.unit(normal);

double mag = vel.mod(vel);

vel.unit(vel);

double pdt = dot(vel,normal);

vel.x = vel.x - (2\*pdt\*normal.x);

vel.y = vel.y - (2\*pdt\*normal.y);

vel.x\*=mag;

vel.y\*=mag;

delay(10);

}

}

void play\_1()

{

placeball\_bar();

start\_1();

while(not\_out()&&(score/50 <= number))

{

setcolor(BLACK);

setfillstyle(1,0);

delay(8);

bar(pos.x-5,pos.y-5,pos.x+5,pos.y+5);

bar(barr.x-2,barr.y-2,barr.x+98,barr.y+14);

update\_vel();

if(pos.y>461) check\_hit\_bar();

if(pos.y<300)

{

if(check\_hit\_block())

{

delete\_block(hit.x,hit.y);

}

}

controls\_1();

check\_corner();

update\_pos();

putimage(pos.x-5,pos.y-5,im,0);

}

if(score/50 < number) { display\_score(); exit(0); }

else display\_score();

}

void play\_2()

{

placeball\_circpad();

start\_2();

while(not\_out()&&(score/50 != number))

{

setcolor(BLACK);

setfillstyle(1,0);

delay(8);

bar(pos.x-5,pos.y-5,pos.x+5,pos.y+5);

bar(circ.x-21,circ.y-21,circ.x+21,circ.y+21);

update\_vel();

if(pos.y>430)check\_hit\_circpad();

if(pos.y<400)

{

if(check\_hit\_block())

{

delete\_block(hit.x,hit.y);

}

}

controls\_2();

check\_corner();

update\_pos();

putimage(pos.x-5,pos.y-5,im,0);

}

if(score/50 < number) { display\_score(); exit(0); }

else display\_score();

}

void main()

{

int gdriver= 9, gmode=2;

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

cleardevice();

setfillstyle(1,0);

bar(640,0,700,480);

layout\_1();

play\_1();

getch();

cleardevice();

layout\_2();

play\_1();

getch();

cleardevice();

layout\_3();

play\_1();

getch();

cleardevice();

layout\_4();

play\_1();

getch();

cleardevice();

layout\_5();

play\_1();

getch();

cleardevice();

layout\_6();

play\_1();

getch();

cleardevice();

layout\_7();

play\_1();

getch();

cleardevice();

layout\_8();

play\_1();

getch();

score = 0;

cleardevice();

setfillstyle(1,0);

bar(640,0,700,480);

layout\_1();

play\_2();

getch();

cleardevice();

layout\_2();

play\_2();

getch();

cleardevice();

layout\_3();

play\_2();

getch();

cleardevice();

layout\_4();

play\_2();

getch();

cleardevice();

layout\_5();

play\_2();

getch();

cleardevice();

layout\_6();

play\_2();

getch();

cleardevice();

layout\_7();

play\_2();

getch();

cleardevice();

layout\_8();

play\_2();

getch();

cleardevice();

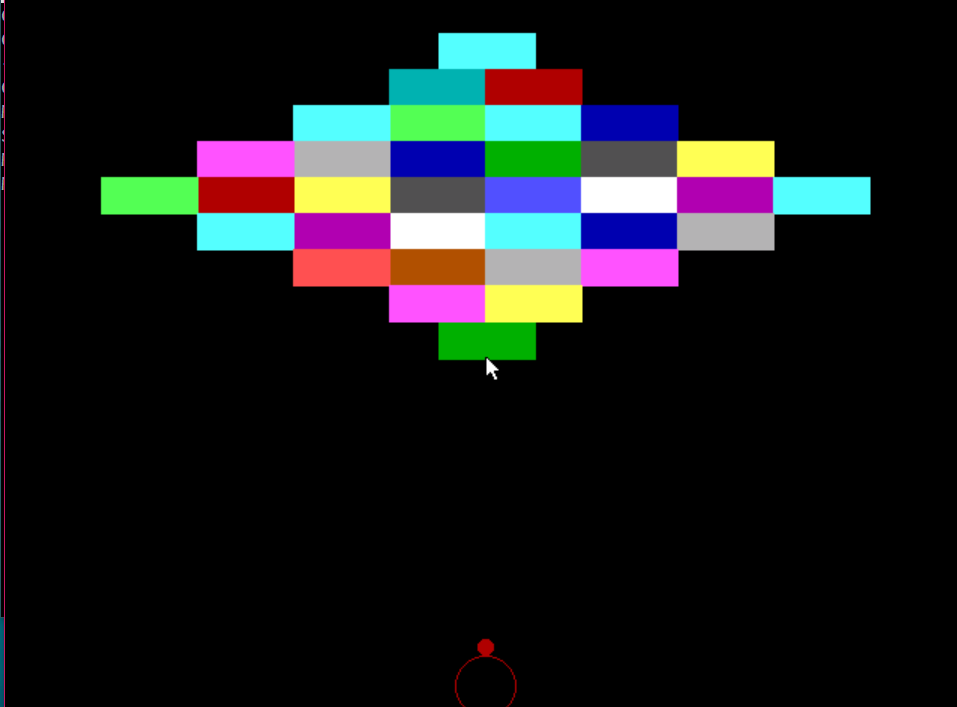
closegraph();

}

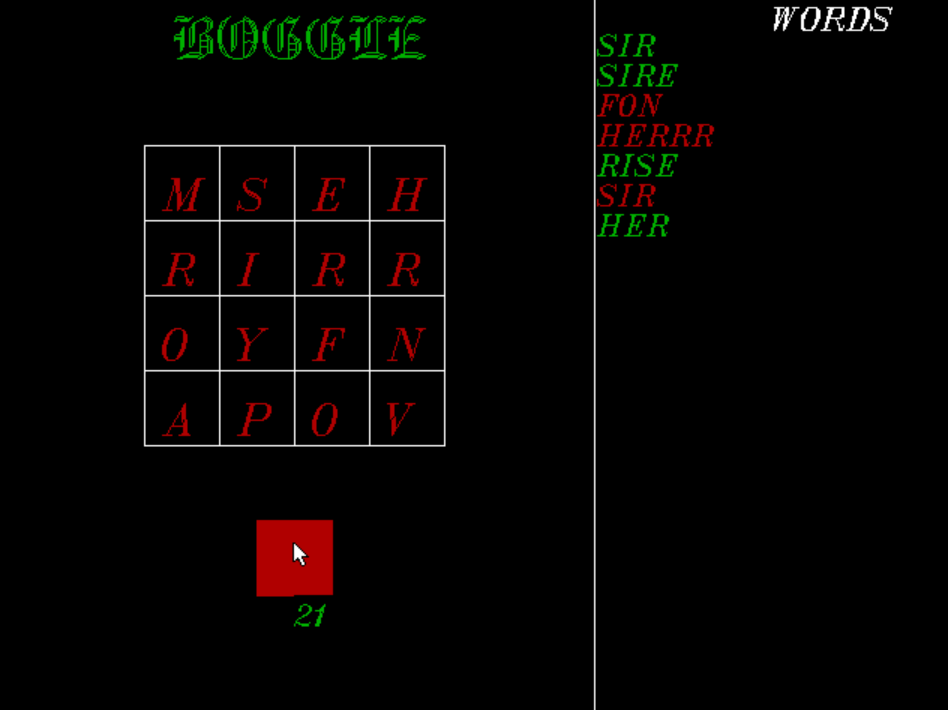
--------------------------THANK YOU-----------------------------

**SAMPLE OUTPUT**

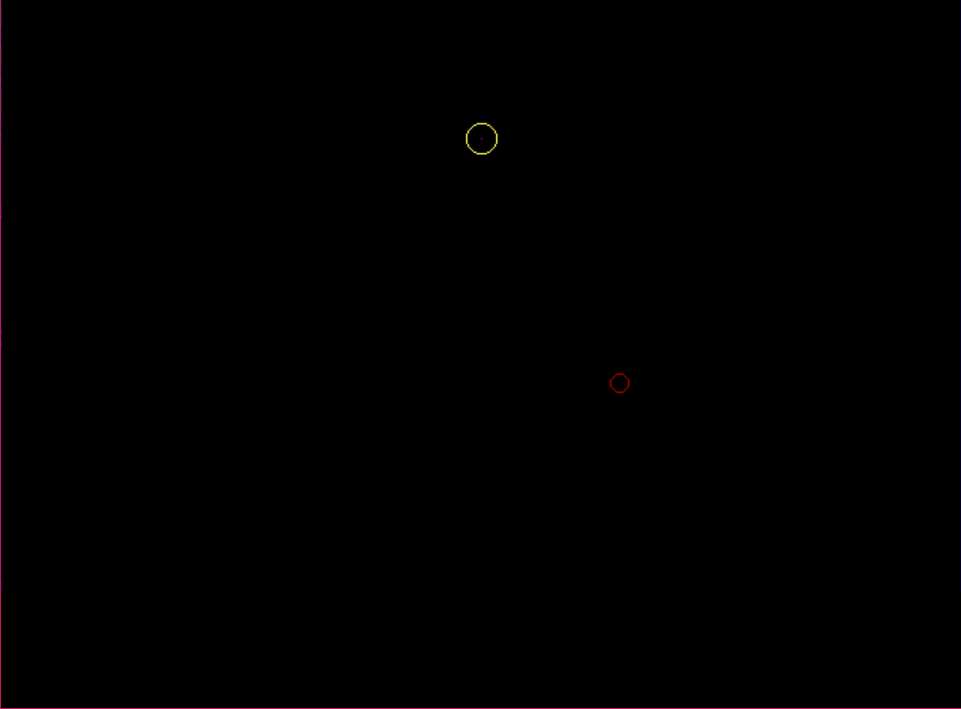
**ATARI BREAKOUT**

****

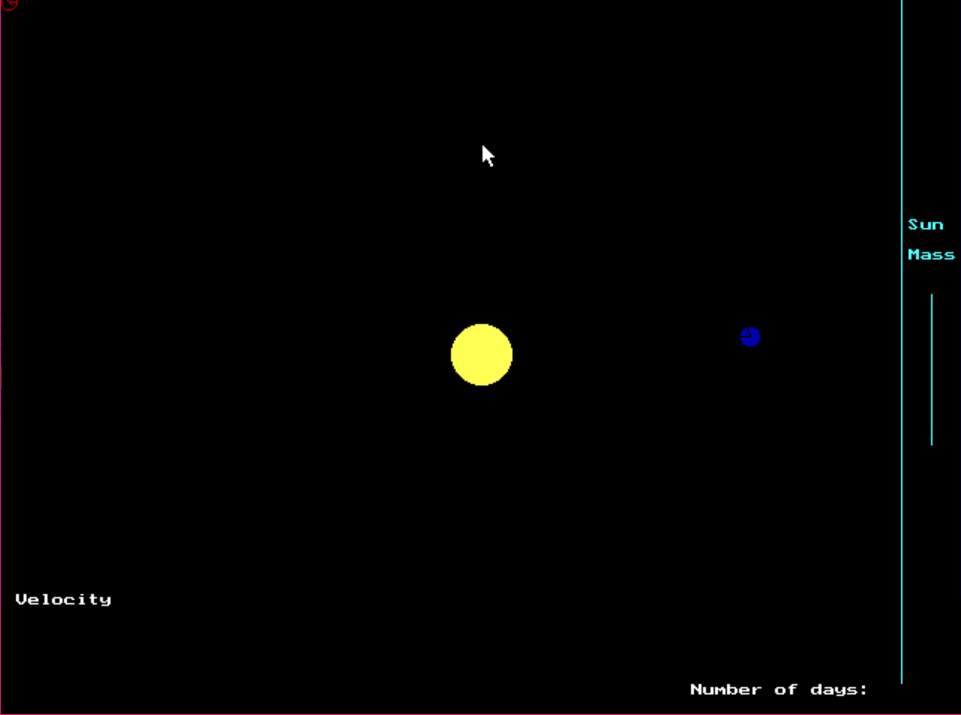
**BOGGLE**

****

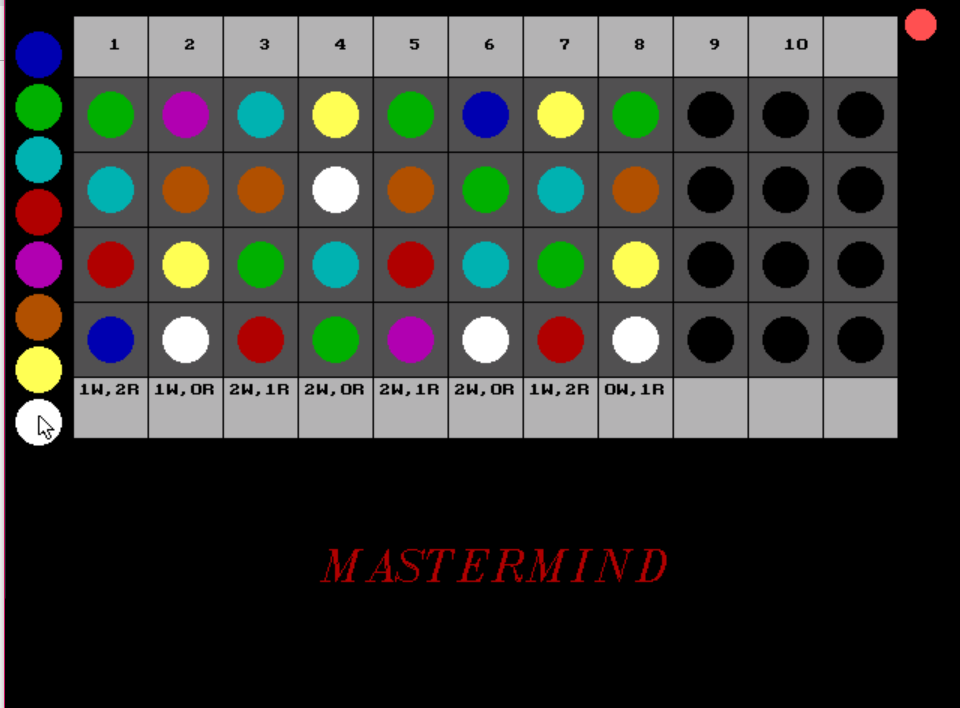
**MISSILES**

****

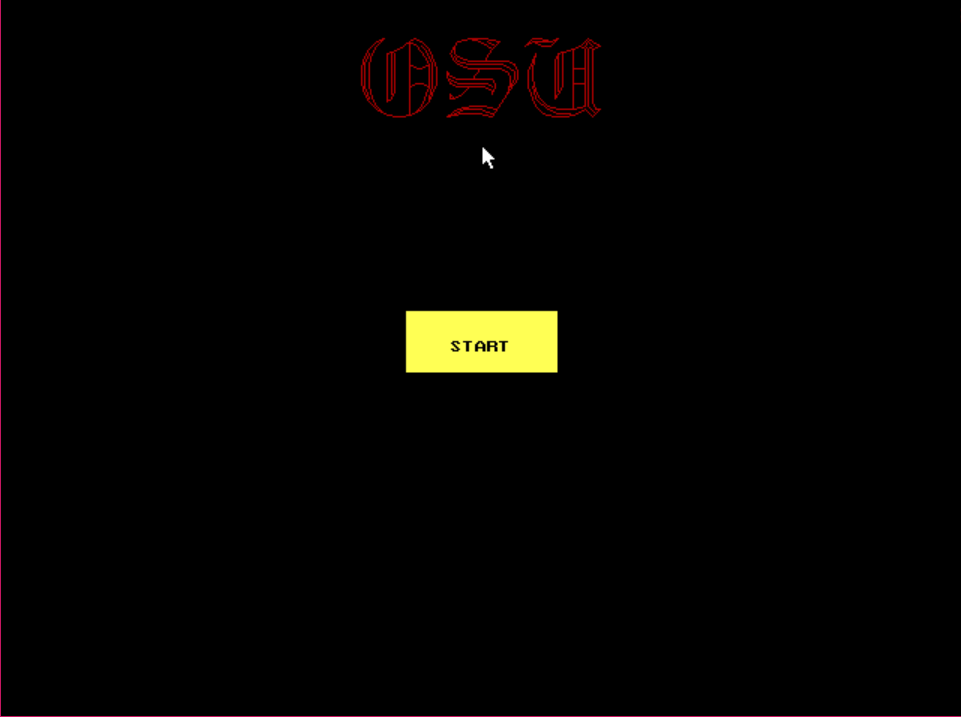
**GRAVITY**

****

**MASTERMIND**

****

**OSU**

****

**BIBLIOGRAPHY**

<https://elliptigon.com/energy-derivation-for-a-body-moving-under-gravity/>

[**https://sites.google.com/a/vhhscougars.org/johnsearch/searchindex/oxford-english-dictionary**](https://sites.google.com/a/vhhscougars.org/johnsearch/searchindex/oxford-english-dictionary)

**THANK YOU**