#include <GL/glut.h>

#include <stdio.h>

#include <GL/gl.h>

#include <GL/glu.h>

#include<string.h>

#define bool int

#define true 1

#define false 0

#define FROM\_RIGHT 1

#define FROM\_LEFT 2

#define FROM\_TOP 3 #define FROM\_BOTTOM 4

int game=0; int p=0; int a,b,c,d; static mouse\_x=0; static int value = 0; static int submenu\_id; static int bmenu\_id; static int menu\_id; static int nu\_id; static int id; static int window;

double r1=1,g1=1,b1=1,r2=1,g2=1,b2=1.0; static int z=0;

static int WINDOW\_WIDTH ,WINDOW\_HEIGHT; int playerResult=0; int pcResult =0;

static float Xspeed=1,Yspeed=1; //for moving ball static float delta=1; //ball movements in steps char string [100]; static int sizeb=0;

//structure for drawing ball and bat typedef struct RECTA

{

float left,top,right,bottom;

}RECTA;

RECTA ball={10,10,20,20};

RECTA wall ;

RECTA player\_1 ={100,490,40,500};

//increase ball size void incsize(RECTA rect)

{

ball.right+=10; ball.bottom+=10;

}

//decrease ball size void decsize(RECTA rect) {

if(ball.left<ball.right)

{

ball.left+=10; ball.top+=10;

}

}

//drawing ball

void DrawBall(RECTA rect,double r,double g,double b)

{

glColor3f(r,g,b); glBegin(GL\_QUADS);

glVertex2f(rect.left,rect.bottom); glVertex2f(rect.right,rect.bottom); glVertex2f(rect.right,rect.top); glVertex2f(rect.left,rect.top); glEnd();

}

//drawing bat

void DrawBat(RECTA rect,double r,double g,double b)

{

glColor3f(r,g,b); glBegin(GL\_QUADS);

glVertex2f(rect.left,rect.bottom); glVertex2f(rect.right,rect.bottom); glVertex2f(rect.right,rect.top); glVertex2f(rect.left,rect.top); glEnd();

}

//timer funtion for moving ball void Timer(int v)

{

ball.left+=Xspeed; ball.right+=Xspeed; ball.top+=Yspeed; ball.bottom+=Yspeed; glutTimerFunc(1,Timer,1);

}

//drawing text

void drawText(char\* string,int x, int y)

{ int len, i; glRasterPos2f(x,y); len=(int) strlen(string); for(i = 0; i < len; i++)

{

glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24,string[i]); }

}

//test collision between ball and wall int Test\_Ball\_Wall(RECTA ball , RECTA wall)

{

if(ball.right >=wall.right) return FROM\_RIGHT; if(ball.left <=wall.left) return FROM\_LEFT; if(ball.top <=wall.top) return FROM\_TOP; if(ball.bottom >=wall.bottom) return FROM\_BOTTOM; else return 0 ;

}

//calculating score

bool Test\_Ball\_Player(RECTA ball,RECTA player)

{

if(ball.bottom >= player.top && ball.left>= player.left && ball.right <=player.right )

{

playerResult++; return true;

}

return false;

}

//repositioning ball void rend(RECTA rect)

{ ball.left=10; ball.top=10; ball.right=20; ball.bottom=20;

}

void rendp(RECTA rect)

{ a=ball.left; b=ball.top; c=ball.right; d=ball.bottom;

} void rendr(RECTA rect)

{ ball.left=a; ball.top=b; ball.right=c; ball.bottom=d;

}

//key Board Messages

void keyboard(unsigned char key, int x, int y)

{

switch (key)

{

case 'e':exit(0);break; case 'n'|'N':startscreenn();break; case '1':choice1();break; case '2':renderr2();break; case '3':exit(0);break; case '4':winscreenn();break; case 'p':rendp(ball);pause11();break; case 'r':rendr(ball);renderr1();break; case

'c':rend(ball);delta=1;pcResult=0;playerResult=0;game=0;renderr1();bre ak; case

'a':rend(ball);delta=1;pcResult=0;playerResult=0;game=1;renderr1();bre ak; }

}

//key Board Message

void inputKey(int key, int x, int y)

{

switch (key)

{

case GLUT\_KEY\_LEFT :decsize(ball);break; case GLUT\_KEY\_RIGHT:incsize(ball);break; case GLUT\_KEY\_UP :delta++;break ;

case GLUT\_KEY\_DOWN :if(delta>1) delta--;break;

}

}

//moving bat on x-axis void MouseMotion(int x,int y)

{

mouse\_x=x;

}

//openGL Setting

void Setting(double r,double g,double b,double alpha)

{

glClearColor (r, g, b, alpha);

glHint(GL\_PERSPECTIVE\_CORRECTION\_HINT, GL\_NICEST);

}

//windowResize void reshape (int w, int h) {

WINDOW\_WIDTH =w ; WINDOW\_HEIGHT =h ;

glViewport (0, 0, (GLsizei) w, (GLsizei) h); glMatrixMode (GL\_PROJECTION); glLoadIdentity (); gluOrtho2D (0, w, h, 0); glMatrixMode (GL\_MODELVIEW); glLoadIdentity ();

}

//putting all things together void Renderc(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(1,1,1); sprintf(string,"PC : %d ",pcResult); drawText(string,10,40); sprintf(string,"Player : %d ",playerResult); drawText(string,10,60); drawText("PRESS '4' TO EXIT",10,80); drawText("PRESS 'P' TO PAUSE",10,100); wall.left=wall.top=0; wall.right=WINDOW\_WIDTH; wall.bottom=WINDOW\_HEIGHT; DrawBall(ball,r1,g1,b1);

if(Test\_Ball\_Wall(ball,wall)== FROM\_RIGHT) Xspeed=-delta;

if(Test\_Ball\_Wall(ball,wall)== FROM\_LEFT) Xspeed=delta;

if(Test\_Ball\_Wall(ball,wall)== FROM\_TOP) Yspeed=delta;

if(Test\_Ball\_Wall(ball,wall)== FROM\_BOTTOM)

{

Yspeed=-delta; pcResult +=1;

}

DrawBat(player\_1,r2,g2,b2); player\_1.left=mouse\_x-20; player\_1.right=mouse\_x+40;

if(Test\_Ball\_Player(ball,player\_1)==true) Yspeed=-delta; glutSwapBuffers(); if(p==0) { sleep(1); p++;

}

}

void Rendera(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(1,1,1); sprintf(string,"PC : %d ",pcResult); drawText(string,10,40); sprintf(string,"Player : %d ",playerResult); drawText(string,10,60); drawText("PRESS '4' TO EXIT",10,80); drawText("PRESS 'P' TO PAUSE",10,100); wall.left=wall.top=0; wall.right=WINDOW\_WIDTH; wall.bottom=WINDOW\_HEIGHT; DrawBall(ball,r1,g1,b1);

if(Test\_Ball\_Wall(ball,wall)== FROM\_RIGHT) Xspeed=-delta;

if(Test\_Ball\_Wall(ball,wall)== FROM\_LEFT) Xspeed=delta;

if(Test\_Ball\_Wall(ball,wall)== FROM\_TOP) Yspeed=delta;

if(Test\_Ball\_Wall(ball,wall)== FROM\_BOTTOM)

{

winscreenn();

}

DrawBat(player\_1,r2,g2,b2); player\_1.left=mouse\_x-20; player\_1.right=mouse\_x+40;

if(Test\_Ball\_Player(ball,player\_1)==true) Yspeed=-delta; glutSwapBuffers(); if(p==0) { sleep(1); p++;

}

} void disp() { if(game==0) Renderc(); else if(game==1)

Rendera();

}

//instructions void Render2(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(0,1,0);

drawText("e : exit",20,40); drawText("right click : options",20,100); drawText("up arrow : increase ball speed",20,160); drawText("down arrow : decrease ball speed",20,220); drawText("left arrow : decrease ball size",20,280); drawText("right arrow : increase ball size",20,340); drawText("mouse : pad movements",20,400); glColor3f(1,0,0);

drawText("\"touching the ball to the corner of pad will fetch more points\"",80,440); glColor3f(0,1,1);

drawText("P R E S S N T O G O B A C K",200,500); glutSwapBuffers();

}

//menus options void menu(int num)

{

if(num == 0)

{

glutDestroyWindow(window); exit(0); } else {

switch(num) {

case 1:r2=1.0,g2=1.0,b2=1.0;break; case 2:r2=1.0,g2=1.0,b2=0.0;break; case 3:r2=0.0,g2=1.0,b2=1.0;break; case 4:r2=0.5,g2=1.0,b2=0.5;break; case 5:r1=1.0,g1=1.0,b1=1.0;break; case 6:r1=1.0,g1=1.0,b1=0.0;break; case 7:r1=0.0,g1=1.0,b1=1.0;break; case 8:r1=0.5,g1=1.0,b1=0.5;break; case 9:Setting(0,0,1,0.5);break; case 10:Setting(1,0.5,0,0);break; case 11:Setting(0.5,0,1,0);break; case 12:Setting(0.0,0,0.0,0);break;

} }

glutPostRedisplay();

}

//creating menus void createMenu(void)

{

//sub menu entry

submenu\_id = glutCreateMenu(menu); glutAddMenuEntry("White",1); glutAddMenuEntry("Yellow",2); glutAddMenuEntry("Cyan",3);

glutAddMenuEntry("Green",4); //submenu entry bmenu\_id = glutCreateMenu(menu); glutAddMenuEntry("White",5); glutAddMenuEntry("Yellow",6); glutAddMenuEntry("Cyan",7); glutAddMenuEntry("Green",8);

//sub menu entry

menu\_id = glutCreateMenu(menu); glutAddMenuEntry("Blue",9); glutAddMenuEntry("Orange",10); glutAddMenuEntry("Purple",11); glutAddMenuEntry("Black",12);

//main menu

nu\_id = glutCreateMenu(menu); glutAddSubMenu("BAT COLOR", submenu\_id); glutAddSubMenu("BALL COLOR", bmenu\_id); glutAddSubMenu("BACKGROUND", menu\_id); glutAddMenuEntry("exit",0); glutAttachMenu(GLUT\_RIGHT\_BUTTON);

}

//front screen void frontscreen(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(0,0,1);

drawText("JSS ACADEMY OF TECHNICAL EDUCATION, Bengaluru",180,30); drawText("DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING",150,70); glColor3f(0,1,0);

drawText("A Mini Project On:",380,150); drawText("\"2D-GAME\"",400,180); drawText("\"USING OPENGL\"",370,210);

glColor3f(1,0,1);

drawText("Umang Agarwal",10,270); drawText("1js10cs087",10,300);

drawText("computer science and engineering",10,330); drawText("Swathi Phatak",600,270); drawText("1js10cs083",600,300);

drawText("computer science and engineering",600,330); glColor3f(0,1,1);

drawText("UNDER THE GUIDANCE OF:",320,380); drawText("1.Sharana Basavana Gowda(B.E.)",10,430); drawText("Professor,Dept.of CSE",10,460); drawText("2.Savitha S(B.E.)",10,490); drawText("Professor,Dept. of CSE",10,520); glColor3f(1,1,1); drawText("PRESS N TO GO TO NEXT SCREEN",285,550); glColor3f(0,0,0); glutSwapBuffers();

}

void pause1()

{ glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(1,1,0); drawText("PAUSE",200,150); //glColor3f(0,1,0);

drawText("PRESS R TO RESUME",140,300); glColor3f(0,0,0); glutSwapBuffers();

}

void choice() {

glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(0,1,0);

drawText("PLAYING AGAINST COMPUTER",200,150); drawText("P R E S S C",200,180); glColor3f(0,0,1);

drawText("PLAYING ALONE",200,300); drawText("P R E S S A",200,330); glColor3f(1,1,1);

drawText("P R E S S E T O E X I T ",200,430); glutSwapBuffers();

}

//start screen void startscreen()

{

glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(1,1,1);

drawText("WELCOME TO 2D GAME",250,150); glColor3f(0,1,0); drawText("1.NEW GAME",250,200); glColor3f(1,1,0);

drawText("2.INSTRUCTIONS",250,250); glColor3f(1,0.5,0); drawText("3.QUIT",250,300); glColor3f(0,0,0); glutSwapBuffers();

}

//last screen void winscreen()

{

glClear(GL\_COLOR\_BUFFER\_BIT ); glMatrixMode(GL\_MODELVIEW); glLoadIdentity(); glColor3f(0,1,0);

drawText("!!! C O N G R A T S !!!",270,60); glColor3f(1,0.5,0);

drawText("P O I N T S A R E",290,200); sprintf(string,"PC : %d",pcResult); drawText(string,100,300);

sprintf(string,"PLAYER : %d",playerResult); drawText(string,600,300); glColor3f(1,1,0);

drawText("\*\*\*PRESS \"n\" TO GO TO MAIN MENU\*\*\*",180,420); drawText("\*\*\*PRESS \"1\" TO RESTART THE GAME\*\*\*",170,460); drawText("\*\*\*PRESS \"e\" TO EXIT FROM THE GAME\*\*\*",160,500); glutSwapBuffers();

}

int pause11()

{

glutInitDisplayMode ( GLUT\_DOUBLE | GLUT\_RGB); glutInitWindowSize (600, 500); glutInitWindowPosition (250,70); window=glutCreateWindow("PAUSE"); glutDisplayFunc(pause1); glutIdleFunc(pause1); glutReshapeFunc(reshape); glutKeyboardFunc(keyboard); Setting (0,0,0,0); glutSpecialFunc(inputKey); glutMainLoop(); return 0;

}

int choice1()

{

glutDestroyWindow(window);

glutInitDisplayMode ( GLUT\_DOUBLE | GLUT\_RGB); glutInitWindowSize (800, 600); glutInitWindowPosition (250,70); window=glutCreateWindow("CHOICE"); glutDisplayFunc(choice); glutIdleFunc(choice); glutReshapeFunc(reshape); glutKeyboardFunc(keyboard); Setting (0,0,0,0); glutSpecialFunc(inputKey); glutMainLoop();

return 0;

}

//showing instructions int renderr2() {

glutDestroyWindow(window);

glutInitDisplayMode ( GLUT\_DOUBLE | GLUT\_RGB); glutInitWindowSize (800, 600); glutInitWindowPosition (250,70); window=glutCreateWindow("INSTRUCTIONS"); glutDisplayFunc(Render2); glutIdleFunc(Render2); glutReshapeFunc(reshape); glutKeyboardFunc(keyboard); Setting (0,0,0,0); glutSpecialFunc(inputKey); glutMainLoop(); return 0;

}

//showing start screen int startscreenn()

{

glutDestroyWindow(window);

glutInitDisplayMode ( GLUT\_DOUBLE | GLUT\_RGB); glutInitWindowSize (800, 600); glutInitWindowPosition (250, 70); window=glutCreateWindow("START"); glutDisplayFunc(startscreen); glutIdleFunc(startscreen); glutReshapeFunc(reshape); glutKeyboardFunc(keyboard); Setting (0,0,0,0); glutSpecialFunc(inputKey); glutMainLoop(); return 0;

}

//game int renderr1() {

glutDestroyWindow(window);

glutInitDisplayMode ( GLUT\_DOUBLE | GLUT\_RGB); glutInitWindowSize (600, 500); glutInitWindowPosition (250, 70); window=glutCreateWindow("GAME"); glutDisplayFunc(disp); glutIdleFunc(disp); glutTimerFunc(1,Timer,1); glutReshapeFunc(reshape); glutKeyboardFunc(keyboard); glutPassiveMotionFunc(MouseMotion);

Setting (0,0,0,0); createMenu(); glutSpecialFunc(inputKey); glutMainLoop(); return 0;

}

//showing last screen int winscreenn()

{

glutDestroyWindow(window);

glutInitDisplayMode ( GLUT\_DOUBLE | GLUT\_RGB); glutInitWindowSize (800, 600); glutInitWindowPosition (250,70); window=glutCreateWindow("RESULT"); glutDisplayFunc(winscreen); glutIdleFunc(winscreen); glutReshapeFunc(reshape); glutKeyboardFunc(keyboard);

Setting (0,0,0,0); glutMainLoop(); return 0;

}

//showing welcome screen int main(int argc, char\*\* argv)

{

glutInit(&argc, argv);

glutInitDisplayMode ( GLUT\_DOUBLE | GLUT\_RGB ); glutInitWindowSize (1000, 600); glutInitWindowPosition (150, 70); window=glutCreateWindow("WELCOME"); glutDisplayFunc(frontscreen); glutIdleFunc(frontscreen); glutReshapeFunc(reshape); glutKeyboardFunc(keyboard); Setting (0,0,0,0); glutMainLoop(); return 0;

}