

Projet I3333

Sujet : C++ Animation

Etudiant : Michel Sleiman Rahme

Numéro de dossier : 56312

|  |  |  |
| --- | --- | --- |
| **Contents:** |  | |
| Contents | . . . . . . . . . . . . . . . . . . . . . . . . . | 2 |
| Introduction | . . . . . . . . . . . . . . . . . . . . . . . . . | 5 |
| Objective | . . . . . . . . . . . . . . . . . . . . . . . . . | 7 |
| Methodology | . . . . . . . . . . . . . . . . . . . . . . . . . | 7 |
| System Specification | . . . . . . . . . . . . . . . . . . . .. ….. | 8 |
| Works Done | . . . . . . . . . . . . . . . . . . . . . . . . . | 9 |
| Result and Discussion | . . . . . . . . . . . . . . . . . . . .. ….. | 12 |
| Conclusion | . . . . . . . . . . . . . . . . . . . . . . . . . | 12 |
| Appendix (code) | . . . . . . . . . . . . . . . . . . . .. ….. | 13 |

**Introduction:**

In the project “COIN COLLECTING 2D GAME”, the project is programmed using c/c++ . the project involves the player and coins on which basis game is built. In this game there will be one player. USER is the player of the game. This program contains Functions to col- lect the ball.

This project includes the multiple windows, menus and submenus using which color of the player & coins, screen color, coin size will be changed. These all actions are assigned to keyboard.

User-interface is provided by means of Keyboard. By using arrow keys player can be moved. By using keyboard user can handle menu and submenu.

## OpenGL Interface:

OpenGL is an application program interface (API) offering various functions to implement primitives, models and images. This offers functions to create and manipulate render light- ing, coloring, viewing the models. OpenGL offers different coordinate system and frames. OpenGL offers translation, rotation and scaling of objects.

Most of our applications will be designed to access OpenGL directly through functions in three libraries. They are:

1. **Main GL:** Library has names that begin with the letter gl and are stored in a library usu- ally referred to as GL.
2. **OpenGL Utility Library (GLU):** This library uses only GL functions but contains code for creating common objects and simplifying viewing.
3. **OpenGL Utility Toolkit (GLUT):** This provides the minimum functionality that should be accepted in any modern windowing system.

## OpenGL Overview:

* + OpenGL(Open Graphics Library) is the interface between a graphic program and graphics hardware. ***It is streamlined*.** In other words, it provides low-level functional- ity. For example, all objects are built from points, lines and convex polygons. Higher level objects like cubes are implemented as six four-sided polygons.
  + OpenGL supports features like 3-dimensions, lighting, anti-aliasing, shadows, tex- tures, depth effects, etc.
  + ***It is a state machine*.** At any moment during the execution of a program there is a current model transformation
  + ***It is a rendering pipeline*.** The rendering pipeline consists of the following steps:
    - Defines objects mathematically.
    - Arranges objects in space relative to a viewpoint.
    - Calculates the color of the objects.

# Objective:

The aim of this project is to develop a graphics package which supports basic operations which include building a 2D GAME using Open GL. The package must also has a user- friendly interface. The objective of developing this model was to design and apply the skills I learnt in class.

# Methodology:

To implement the above “2D Game” ,the following methodology needs to be followed :

* The game consists four cpp file: declaration.cpp, drawing.cpp,move.cpp and main.cpp..
* Specifying the Application and various components of the Architecture.
* First including the all header file related to project in Declaration file.
* Declaration of player and coins related structure.
* Declaration of the function to draw Rectangle, Colored Circle, Line, Quads etc.
* Drawing the player and coins using structure and declared shape’s function. And de- cide the initial position of all player and coins.
* Applying the movement function on player and coins.
* Collision detection between player-coins, player-walls, coins-walls. After detecting collision update the parameter accordingly.
* Showing the menus, submenus and score on windows screen.

# System Specification:

### HARDWARE REQUIREMENTS :

* + Keyboard

### SOFTWARE REQUIREMENTS :

* + Programming language – C/C++ using OpenGL
  + Operating system – Windows operating system
  + Compiler – GCC Compiler
  + Graphics library – GL/glut.h

### FUNCTIONAL REQUIREMENTS :

* + **OpenGL APIs:**

If we want to have a control on the flow of program and if we want to interact with the window system then we use OpenGL API’S. Vertices are represented in the same manner internally, whether they are specified as two-dimensional or three-dimen- sional entities, everything that we do are here will be equally valid in three dimen- sions. Although OpenGL is easy to learn, compared with other APIs, it is neverthe- less powerful. It supports the simple three dimensional programs and also supports the advanced rendering techniques.

### GL/glut.h :

We use a readily available library called the OpenGL Utility Toolkit (GLUT), which provides the minimum functionality that should be expected in any modern window- ing system.

The application program uses only GLUT functions and can be recompiled with the GLUT library for other window system. OpenGL makes a heavy use of

macros to increase code readability and avoid the use of magic numbers. In most im- plementation, one of the include line.

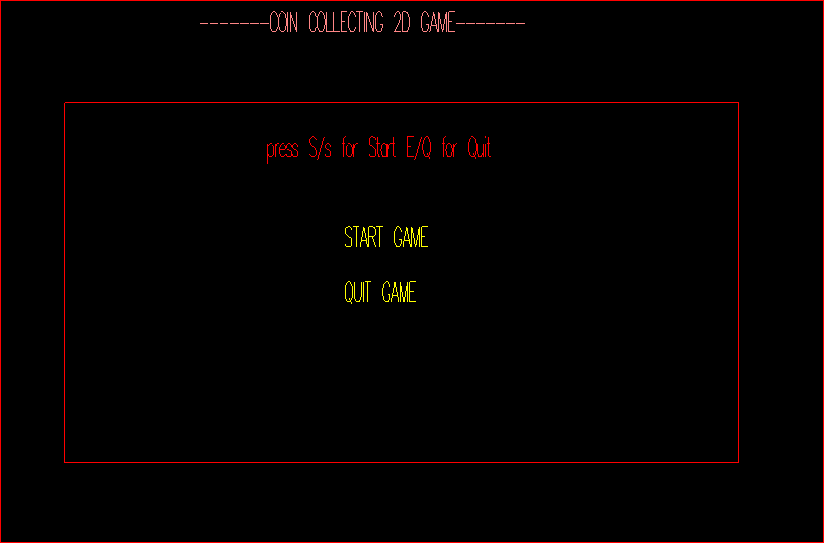
# Works done:

In ‘coin collecting game 2D game’ different state are created which defines the Game-state.

* Start Menu
* Game Mode
* Pause
* Game Over

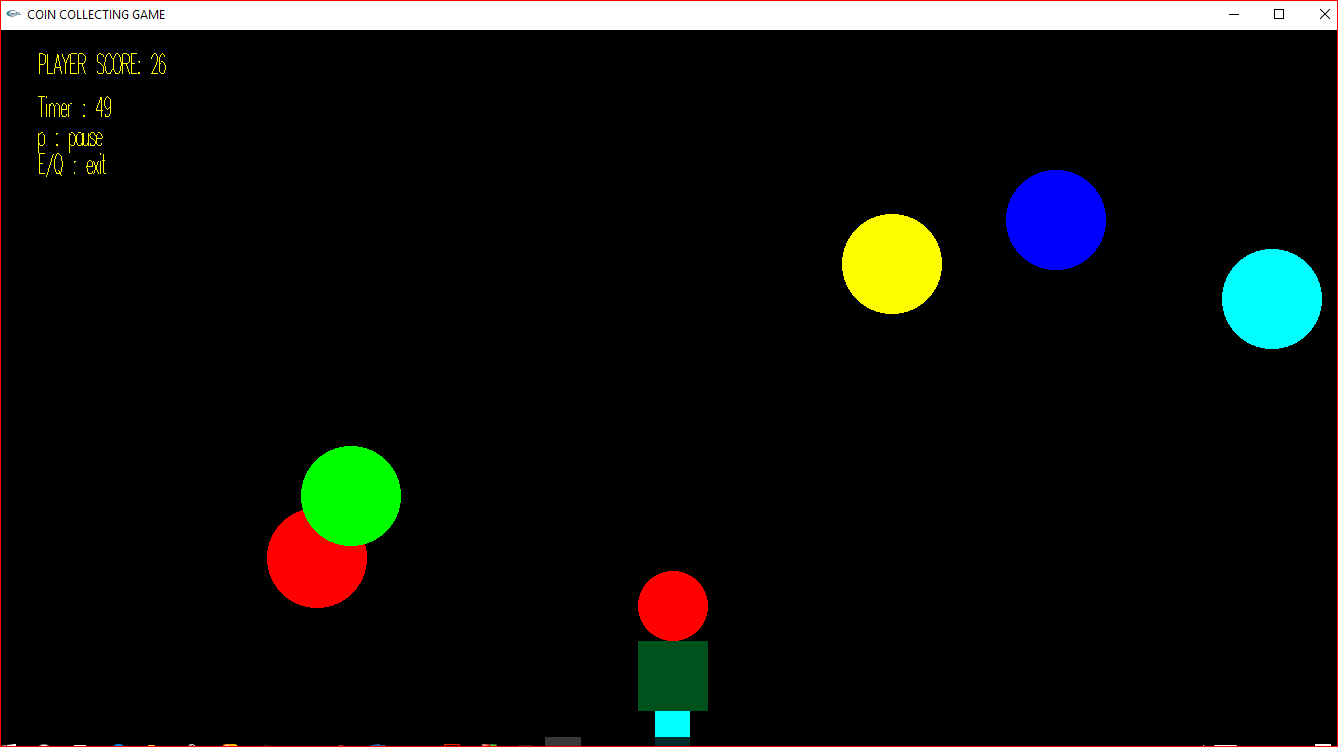
### Start Menu:

This is the first screen of the game. In this state user can play or exit the game using keyboard.



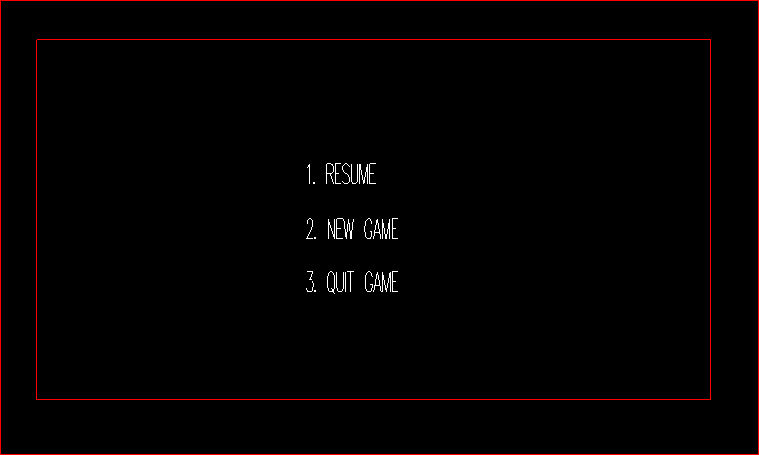
### Play Mode:

In this state user can play the game using keyboard. User can handle the movement of player using arrows key.



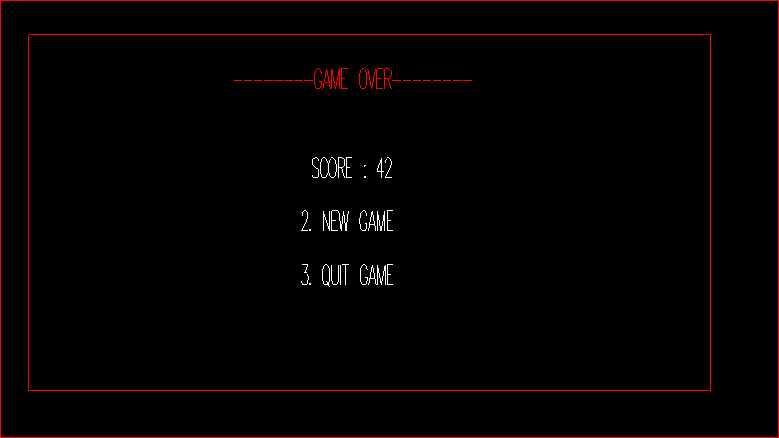
### Pause Mode:

In pause mode the options are available which can be choose by user.



### Game Over:

This is game over state. In this state final score are showed and new game and quit option are available.



# Result and discussion:

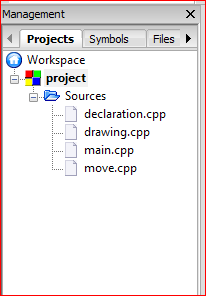
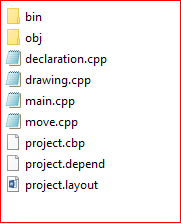
* + Even though we had no prior knowledge of how to proceed with this project, we put our ideas and creativity at work and succeeded in completing this project.
  + It was quite a learning experience and made me familiar as to how open gl and c++ works together.
  + This project also helped me create an interest in the field of animations.
  + The whole group played an active part of the discussion while making the project.
  + A lot of discussion was done in the group before finally deciding on this project.
  + As a result, now we have a game which has solely been made by me.

# Conclusion:

The project was started with modest aim with no prior experience in any programming projects as this, but ended up in learning many things, fine tuning the programming skills and getting into the real world of software development with an exposure to corporate envi- ronment. During the development of any software of significant utility, we are forced with the tradeoff between speed of execution and amount of memory consumed. This is simple interactive application. It is extremely user friendly and has the features, which makes sim- ple graphics project. It has an open source and no security features has been included. The user is free to alter the code for feature enhancement. Checking and verification of all possi- ble types of the functions are taken care. Care was taken to avoid bugs. Bugs may be re- ported to creator as the need.

# Appendix(Code):

### Code:



**Appendix a:**

**Declaration.cpp file:**

In this file Some function and variable are defined which is used in another function**.** In it all required header file are included.

//header file

#include <time.h> #include <windows.h> #include <iostream> #include<math.h> #ifdef APPLE

#include <GLUT/glut.h> #else

#include <GL/glut.h> #endif

#include <stdlib.h> #include <string.h> #include <stdio.h> #define PI 3.14

int gameTime=60;

//different gameState of game

int menu=1,play=2,pause=3,over=4,gameexit=5;

GLfloat headRadius=35; //player head radius

GLfloat coinRadius=50; //coin radius char str [100];

int game\_state=menu; //store the game state

float c1Speed=6,c2Speed=8,c3Speed=10,c4Speed=12,c5Speed=13;//Speed of Five balls float ttime; //store time passes

int t=0; //storing previous time

int i=0,j=0,k=0,l=0,m=0; // use to set random position int score=0; //store score

int w=1350,h=720; //World Size

//double Yspeed=10.0f; //speed of Coins

double Xspeed=60.0f; //speed of player

//float y=0; struct Cir

{

float Xc,Yc,radius;

};

struct Rect

{

float left,right,top,bottom;

};

//structer of player struct Pl

{

Pl()

{

head={0.5\*w,0.20\*h,headRadius};

}

struct Cir head;

struct Rect body;//={625,725,100,50}; struct Rect leg;

};

struct Cir c1={75,720,coinRadius}; //first ball struct Cir c2={300,720,coinRadius}; //second balls struct Cir c3={550,720,coinRadius}; //third balls struct Cir c4={800,720,coinRadius}; //second balls struct Cir c5={1250,720,coinRadius}; //third balls struct Pl player;

struct Rect wall={0,w,h,0};

//timer function

void Timer(int value)

{

c1.Yc-=c1Speed; c2.Yc-=c2Speed; c3.Yc-=c3Speed; c4.Yc-=c4Speed; c5.Yc-=c5Speed;

glutPostRedisplay(); glutTimerFunc(16,Timer,0); ttime+=0.016;

}

//check Collision between coin and player int coin\_player(struct Cir c)

{

float d1=(c.Xc-player.head.Xc); float d2=(c.Yc-player.head.Yc); float d=sqrt(d1\*d1+d2\*d2); if(d<=c.radius+player.head.radius)

return 1;

return 0;

}

//Check collision between Coins and balls int coin\_wall(struct Cir c)

{

if(c.Yc<=wall.bottom) return 1;

return 0;

}

//Fuction to Draw Text

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

{

char \*c; c=string; glPushMatrix();

glTranslatef(x,y,0); glScalef(.1,.2,.1);

for(int i=0;c[i]!='\0';i++)

{

glutStrokeCharacter(GLUT\_STROKE\_ROMAN,c[i]);

}

glPopMatrix();

}

//function to draw FilledCircle

void filledcircle(GLfloat x,GLfloat y,GLfloat r)

{

glBegin(GL\_TRIANGLE\_FAN); glVertex2f(x,y);

int seg=1000;

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

{

double angle=i\*2\*3.14/seg; glVertex2f(x+r\*cos(angle),y+r\*sin(angle));

}

glEnd();

}

//Function to Draw Circle

void Circle(GLfloat x, GLfloat y, GLfloat radius){ int i;

int lineAmount = 100; //# of triangles used to draw circle

//GLfloat radius = 0.8f; //radius GLfloat twicePi = 2.0f \* PI;

glBegin(GL\_LINE\_LOOP);

for(i = 0; i <= lineAmount;i++) { glVertex2f(

x + (radius \* cos(i \* twicePi / lineAmount)), y + (radius\* sin(i \* twicePi / lineAmount))

);

}

glEnd();

}

//function to draw Line

void line(GLfloat x1,GLfloat y1,GLfloat x2,GLfloat y2)

{

glBegin(GL\_LINES); glVertex3f(x1,y1,0.0); glVertex3f(x2,y2,0.0); glEnd();

}

//function to draw rectangle

void rectang(GLfloat x1,GLfloat x2,GLfloat y1,GLfloat y2)

{

glBegin(GL\_QUADS); glVertex2f(x1,y1); glVertex2f(x1,y2);

glVertex2f(x2,y2); glVertex2f(x2,y1); glEnd();

}

//display startMenu void startMenu()

{

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

glPushMatrix(); glLoadIdentity(); glColor3f(0,0,0); glBegin(GL\_QUADS); glVertex2f(0,0); glVertex2f(w,0); glVertex2f(w,h); glVertex2f(0,h); glEnd(); glPopMatrix(); glPushMatrix(); glColor3f(1,0,0);

int wx=w/4; int hy=h/4;

line(wx,3\*hy,3\*wx,3\*hy); line(wx,hy,3\*wx,hy); line(wx,3\*hy,wx,hy); line(3\*wx,hy,3\*wx,3\*hy); glPopMatrix();

char str[100]; glColor3f(1,.5,.5);

sprintf(str,"-------COIN COLLECTING 2D GAME ");

drawText(str,1.4\*wx,3.4\*hy); glColor3f(1,0,0);

sprintf(str,"press S/s for Start E/Q for Quit"); drawText(str,1.6\*wx,2.7\*hy); glColor3f(1,1,0);

sprintf(str," START GAME"); drawText(str,1.8\*wx,2.2\*hy); sprintf(str," QUIT GAME"); drawText(str,1.8\*wx,1.9\*hy); glutSwapBuffers();

}

// display pauseMenu void pauseMenu()

{

glClear(GL\_COLOR\_BUFFER\_BIT);

glMatrixMode(GL\_MODELVIEW); glLoadIdentity();

glPushMatrix(); glLoadIdentity(); glColor3f(0,0,0); glBegin(GL\_QUADS); glVertex2f(0,0); glVertex2f(w,0); glVertex2f(w,h); glVertex2f(0,h); glEnd(); glPopMatrix(); glPushMatrix(); glColor3f(1,0,0);

int wx=w/4; int hy=h/4;

line(wx,3\*hy,3\*wx,3\*hy); line(wx,hy,3\*wx,hy); line(wx,3\*hy,wx,hy); line(3\*wx,hy,3\*wx,3\*hy); glPopMatrix();

char str[100]; glColor3f(1,0,0);

//sprintf(str,"press /s for Start E/e for Quit");

//drawText(str,635,500); glColor3f(1,1,1); sprintf(str,"1. RESUME"); drawText(str,1.8\*wx,2.2\*hy); sprintf(str,"2. NEW GAME"); drawText(str,1.8\*wx,1.9\*hy); sprintf(str,"3. QUIT GAME"); drawText(str,1.8\*wx,1.6\*hy); glutSwapBuffers();

}

//display gameoverMenu void gameoverMenu()

{

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

glPushMatrix(); glLoadIdentity(); glColor3f(0,0,0); glBegin(GL\_QUADS); glVertex2f(0,0); glVertex2f(w,0); glVertex2f(w,h);

glVertex2f(0,h); glEnd(); glPopMatrix(); glPushMatrix(); glColor3f(1,0,0); int wx=w/4;

int hy=h/4; line(wx,3\*hy,3\*wx,3\*hy); line(wx,hy,3\*wx,hy); line(wx,3\*hy,wx,hy); line(3\*wx,hy,3\*wx,3\*hy); glPopMatrix();

char str[100]; glColor3f(1,0,0);

sprintf(str,"--------GAME OVER ");

drawText(str,1.6\*wx,2.7\*hy); glColor3f(1,1,1);

sprintf(str," SCORE : %d",score); drawText(str,1.8\*wx,2.2\*hy); sprintf(str,"2. NEW GAME"); drawText(str,1.8\*wx,1.9\*hy); sprintf(str,"3. QUIT GAME"); drawText(str,1.8\*wx,1.6\*hy); glutSwapBuffers();

}

### Drawing.cpp file:

This function draw Player and Coins using some shape function which is declared in declaration function.

//include declaration file #include "declaration.cpp"

// drawing player void drawman()

{

glColor3f(1,0,0); //set the color of Head float xc=player.head.Xc; //set the center of head float yc=player.head.Yc;

float r=player.head.radius; // set the radius of head filledcircle(xc,yc,r); //drawing head

//set coordinate of player body player.body.left=xc-r; player.body.right=xc+r; player.body.top=yc-r; player.body.bottom=yc-3\*r;

glColor3f(0,0.3,0.1); // set color of player body part rectang(player.body.left,player.body.right,player.body.top,player.body.bottom);

// set leg part of body player.leg.left=xc-r/2;

player.leg.right=xc+r/2; player.leg.top=yc-3\*r; player.leg.bottom=yc-4\*r; glColor3f(0,1,1); // set leg color

rectang(player.leg.left,player.leg.right,player.leg.top,player.leg.bottom);

}

// drawing different coins

void drawcoin1()

{

glColor3f(1,0,0); filledcircle(c1.Xc,c1.Yc,c1.radius); glEnd();

}

void drawcoin2()

{

glColor3f(0,1,0); filledcircle(c2.Xc,c2.Yc,c2.radius);

glEnd();

}

void drawcoin3()

{

glColor3f(0,0,1); filledcircle(c3.Xc,c3.Yc,c3.radius);

glEnd();

}

void drawcoin4()

{

glColor3f(1,1,0); filledcircle(c4.Xc,c4.Yc,c4.radius); glEnd();

}

void drawcoin5()

{

glColor3f(0,1,1); filledcircle(c5.Xc,c5.Yc,c5.radius);

glEnd();

}

### Move.cpp file:

This file handle all movement of player and coins and also check collision between player- coins, player-wall and coins-walls and perform some action after collision.

//include drawing cpp file #include "drawing.cpp" void backGround()

{

glClearColor(0,0,0,0);

}

//Called when the window is resized void reshape(int width, int height)

{

w=(GLfloat)width; h=(GLfloat)height; glViewport(0, 0, w, h);

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

}

void drawScene()

{

glMatrixMode(GL\_MODELVIEW); glLoadIdentity();

drawcoin1(); drawcoin2(); drawcoin3(); drawcoin4(); drawcoin5();

//draw(); glutSwapBuffers();

}

//check collision of coins from wall or player void check\_player\_c1()

{

if(coin\_wall(c1))

{

c1.Yc=750;

i=c1.radius+rand()%(int)(int)(w-2\*c1.radius); c1.Xc=i;

drawcoin2();

}

if(coin\_player(c1))

{

c1.Yc=750;

i=c1.radius+rand()%(int)(w-2\*c1.radius); c1.Xc=i;

drawcoin1(); score++;

}

}

void check\_player\_c2()

{

if(coin\_wall(c2))

{

c2.Yc=750;

j=c2.radius+rand()%(int)(w-2\*c2.radius); c2.Xc=j;

drawcoin1();

}

if(coin\_player(c2))

{

c2.Yc=750;

j=c2.radius+rand()%(int)(w-2\*c2.radius); c2.Xc=j;

drawcoin2(); score++;

}

}

void check\_player\_c3()

{

if(coin\_wall(c3))

{

c3.Yc=750;

k=c3.radius+rand()%(int)(w-2\*c3.radius); c3.Xc=k;

drawcoin4();

}

if(coin\_player(c3))

{

c3.Yc=750;

k=c3.radius+rand()%(int)(w-2\*c3.radius); c3.Xc=k;

drawcoin3(); score++;

}

}

void check\_player\_c4()

{

if(coin\_wall(c4))

{

c4.Yc=750;

l=c4.radius+rand()%(int)(w-2\*c4.radius); c4.Xc=l;

drawcoin3();

}

if(coin\_player(c4))

{

c4.Yc=750;

l=c4.radius+rand()%(int)(w-2\*c4.radius); c4.Xc=l;

drawcoin4(); score++;

}

}

void check\_player\_c5()

{

if(coin\_wall(c5))

{

c5.Yc=750;

m=c5.radius+rand()%(int)(w-2\*c5.radius); c5.Xc=m;

drawcoin5();

}

if(coin\_player(c5))

{

c5.Yc=750;

m=c5.radius+rand()%(int)(w-2\*c5.radius); c5.Xc=m;

drawcoin5(); score++;

}

}

//call all collision function to detect the collision void checkCollision()

{

check\_player\_c1(); check\_player\_c2(); check\_player\_c3(); check\_player\_c4(); check\_player\_c5();

}

// main display function void display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT); //clear the buffer

glLoadIdentity(); if(game\_state==menu)

{

score=0; ttime=0; startMenu();

}

if(game\_state==play)

{

t=ttime; glPushMatrix(); glLoadIdentity(); glColor3f(1,1,0);

sprintf(str,"PLAYER SCORE: %d",score); drawText(str,0.03\*w,0.94\*h); sprintf(str,"Timer : %d",(int )ttime); drawText(str,0.03\*w,0.88\*h); sprintf(str,"p : pause"); drawText(str,0.03\*w,0.84\*h); sprintf(str,"E/Q : exit"); drawText(str,0.03\*w,0.80\*h); glPopMatrix();

drawman(); drawScene(); checkCollision();

}

if((int)ttime==gameTime)

{

ttime=0; //reset the timepassed game\_state=over;

}

if(game\_state==pause)

{

ttime=t; //pause the timer pauseMenu();

}

if(game\_state==over)

{

gameoverMenu();

}

if(game\_state==gameexit)

{

exit(0);

}

glFlush();

}

### Main.cpp file:

This is main cpp file of the programme. In this file some key handling function are defined.

#include "move.cpp" void mainDisplay()

{

backGround(); display();

}

//move player using keyboard void key(int key,int x,int y)

{

if(key==GLUT\_KEY\_RIGHT && player.head.Xc<(w-headRadius))

{

player.head.Xc += 0.26\*Xspeed;

}

if(key==GLUT\_KEY\_LEFT && player.head.Xc>(headRadius))

{

player.head.Xc -= 0.26\*Xspeed;

}

glutPostRedisplay();

}

//Changing of gamestate using keyboard void keyboard(unsigned char key,int x,int y)

{

switch(key)

{

case 'S':

case 's': if(game\_state==menu){game\_state=play;} break;

case 'q':

case 'Q':

case 'E':

case 'e': if(game\_state==menu || game\_state==play){game\_state=gameexit;} break;

case 'P':

case 'p':if(game\_state==play){game\_state=pause;} break;

case '1': if(game\_state==pause){game\_state=play;} break;

case '2': if(game\_state==pause || game\_state==over){game\_state=menu;} break;

case '3': if(game\_state==pause || game\_state==over){game\_state=gameexit;} break;

}

}

int main(int argc, char \*argv[])

{

glutInit(&argc, argv); glutInitWindowSize(1350,720); glutInitWindowPosition(0,0);

glutInitDisplayMode(GLUT\_RGB | GLUT\_DOUBLE); glutCreateWindow("COIN COLLECTING GAME"); glutDisplayFunc(mainDisplay);

glutIdleFunc(display); glutReshapeFunc(reshape); glutTimerFunc(16,Timer,0); glutKeyboardFunc(keyboard); glutSpecialFunc(key); glutMainLoop();

return EXIT\_SUCCESS;

}