```
#include<bits/stdc++.h>
#include <iostream>
#include <string>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <cmath>
#include <sstream>
#include <GL/gl.h>
#include <GL/glut.h>
#include <GL/glu.h>

static int animationPeriod = 4;
static int isAnimate = 0;
int score=0;

const int fact = 3;
const int x = 80;
const double DEG2RAD = 3.14159265358979323384/180;

static double w = 200;
static int flag = 0;
static int walk = 0;
static int x_ = 2500;
using namespace std;

/// dinosaur animation
void animate(int value){
    if(isAnimate){
            /// display refresh
        glutPostRedisplay();
    /// animation detect in each time frame
        glutTimerFunc(animationPeriod, animate, 1);
    }
}

/// keyboard function
void keyInput(unsigned char key , int x, int y){
    /// exit the main screen
    switch(key){
    case 'q':
        exit(0);
         /// otherwise animate
    case ' ':
        if(isAnimate) isAnimate = 0; /// if dinosaur not move
        else{
            isAnimate = 1;
            animate(1);
        }
        break;
    }
}
///check collision
bool collision(double len){
    if(abs(157 + x - (x_ + x + 50)) <= 100 + x){
        if(5 * fact + w <= 650 * len)return 1; /// if collision make then 1
        return 0; /// otherwise 0
    }
    return 0;
}
///special key to jump using up key
void specialKeyInput(int key , int x , int y ){
    if( key == GLUT_KEY_UP && flag==0 && w <= 200.0){
        flag  = 1;
    }
    glutPostRedisplay();
}
///Circle drawing function using cos and sin
void draw_circle(double theta, double inner_radius, double outer_radius, int x, int y, int
```

```
sin_sign = 1, int cos_sign = 1){
   glBegin(GL_POINTS);
   glColor3f((40) / 255.0, (120) / 255.0, (10) / 255.0);
   for(double r = outer_radius; r >= inner_radius; r -= 3.0){
        for(double i = 0; i < theta ; i++){
          double degInRad = i * DEG2RAD;
          glVertex2f( cos_sign * cos(degInRad) * r + x , sin_sign * sin(degInRad) * r + y  );
       }
   }
   glEnd();
}
/// Generate Tree function
void generate_tree(int x_, double len){
/// suitable for tree we take x
    int x = 30;
/// Tree Color
    glColor3f((40) / 255.0, (120) / 255.0, (10) / 255.0);
/// Tree polygon part
/// first polygon (Large)
    glBegin(GL_POLYGON);
        glVertex2f(x_, 250 * len);
        glVertex2f(x_ + x, 250 * len);
        glVertex2f(x_ + x, 650 * len);
        glVertex2f(x_, 650 * len);
    glEnd();

/// first polygon cap
    draw_circle(180.0, 0.0, x / 2, x_ + x / 2, 650 * len); /// For half circle, angle=180


glColor3f((40) / 255.0, (120) / 255.0, (10) / 255.0);
    glBegin(GL_POLYGON); /// second polygon (Small)
        glVertex2f(x_ + x + 25, 400 * len);
        glVertex2f(x_ + x + 50, 400 * len);
        glVertex2f(x_ + x + 50, 600 * len);
        glVertex2f(x_ + x + 25, 600 * len);
    glEnd();
    draw_circle(180.0, 0.0, 25.0 / 2, x_ + x + 75.0 / 2, 600 * len); /// For half circle,
    angle=180


glColor3f((40) / 255.0, (120) / 255.0, (10) / 255.0);
    glBegin(GL_POLYGON); /// Third polygon (small)
        glVertex2f(x_ - 25, 400 * len);
        glVertex2f(x_ - 50, 400 * len);
        glVertex2f(x_ - 50, 600 * len);
        glVertex2f(x_ - 25, 600 * len);
    glEnd();
    draw_circle(180.0, 0.0, 25.0 / 2, x_ - 75.0 / 2, 600 * len);/// For half circle, angle=180


    draw_circle(90.0, 25, 50, x_ + x, 400 * len, -1);/// first Circle between small and large
    polygon
    draw_circle(90.0, 25, 50, x_, 400 * len, -1, -1);/// second Circle between small and large
    polygon
}
/// screen become reset
void reset(){
///initial position of dinosaur
    w = 200;
    flag = 0;
    walk = 0;
    x_ = 2500;
    animationPeriod = 4;
    isAnimate = 0;
}
/// Initial drawing function
void render( void ){
    glClear(GL_COLOR_BUFFER_BIT); /// screen clear and previous data or pixel remove
```

```
    /// starting all screen string print
   glColor3f (0.0, 0.0, 0.0); /// string color
        glRasterPos2f(800, 1700); ///define position on the screen string
        char *string = "Dinosaur Game!!"; /// define string

      while(*string){
        glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24, *string++);/// pick up each character
        /// Font:TIMES_ROMAN and Size: 24
        /// loop exit when we get null string
      }

       glColor3f (0.0, 0.0, 1.0);
        glRasterPos2f(1700, 1700); ///define position on the screen
        char *string1 = "Score:";
/// change to projection status
      while(*string1){
        glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24, *string1++);
      }
       glColor3f (0.0, 0.0, 1.0);
        glRasterPos2f(1800, 1700); ///define position on the screen
            std::stringstream t;
    t << score; ///push int
    char const *string2 =
         t.str().c_str(); /// c library converts
         ///convert int to string
      while(*string2){
        glutBitmapCharacter(GLUT_BITMAP_TIMES_ROMAN_24, *string2++);
      }
 /// ending all screen string print
 /// player ground point size
    glPointSize(2);
/// player ground point draw
    glBegin(GL_POINTS);
/// player ground point color
        glColor3f((0) / 255.0, (0) / 255.0, (0) / 255.0);
        for(int i = 0; i < 100; i++){
/// player ground point randomly
            glVertex2f(rand() % 2000, 200);
            glVertex2f((rand() + 31) % 2000, 150);
        }
    glEnd();
/// Game tree generate
    generate_tree(x_, 1.0);

/// Game tree generate position randomize to give challenge
    if(x_>= 0)
        x_ -= 5;
    else{
        x_ = 2000 + rand()%400;
    }
/// Game tree generate position randomize to give challenge
    glLineWidth(2);
/// Line draw start
    glBegin(GL_LINES);
    /// color
        glColor3f((40) / 255.0, (120) / 255.0, (10) / 255.0);
        glVertex2f(0, 250);
        glVertex2f(2000, 250);
    glEnd();
/// Line draw end

/// Dinosaur drawing
    glLineWidth(10);
    glBegin(GL_LINES);
    /// Dinosaur color
        glColor3f(200 / 255.0, 18 / 255.0, 20 / 255.0);

        glVertex2f(10 + x, 75 * fact + w);
```

```
glVertex2f(10 + x, 45 * fact + w);
glVertex2f(15 + x, 65 * fact + w);
glVertex2f(15 + x, 40 * fact + w);
glVertex2f(20 + x, 60 * fact + w);
glVertex2f(20 + x, 35 * fact + w);
glVertex2f(25 + x, 55 * fact + w);
glVertex2f(25 + x, 35 * fact + w);
glVertex2f(30 + x, 55 * fact + w);
glVertex2f(30 + x, 35 * fact + w);
glVertex2f(35 + x, 55 * fact + w);
glVertex2f(35 + x, 25 * fact + w);
glVertex2f(40 + x, 60 * fact + w);
glVertex2f(40 + x, 5 * fact + w-walk); /// Dinosaur walk point down
glVertex2f(45 + x, 65 * fact + w);
glVertex2f(45 + x, 15 * fact + w);
glVertex2f(45 + x, 10 * fact + w-walk); /// Dinosaur walk point down
glVertex2f(45 + x, 5 * fact + w-walk); /// Dinosaur walk point down
glVertex2f(50 + x, 10 * fact + w-walk); /// Dinosaur walk point down
glVertex2f(50 + x, 5 * fact + w-walk); /// Dinosaur walk point down
glVertex2f(55 + x, 10 * fact + w-walk); /// Dinosaur walk point down
glVertex2f(55 + x, 5 * fact + w-walk); /// Dinosaur walk point down
glVertex2f(50 + x, 65 * fact + w);
glVertex2f(50 + x, 20 * fact + w);
glVertex2f(55 + x, 70 * fact + w);
glVertex2f(55 + x, 25 * fact + w);
glVertex2f(63 + x, 75 * fact + w);
glVertex2f(63 + x, 20 * fact + w);
glVertex2f(70 + x, 115 * fact + w);
glVertex2f(70 + x, 5 * fact + w+walk);  /// Dinosaur walk point up
glVertex2f(78 + x, 120 * fact + w);
glVertex2f(78 + x, 25 * fact + w);
glVertex2f(78 + x, 10 * fact + w+walk);  /// Dinosaur walk point up
glVertex2f(78 + x, 5 * fact + w+walk); /// Dinosaur walk point up
glVertex2f(85 + x, 10 * fact + w+walk); /// Dinosaur walk point up
glVertex2f(85 + x, 5 * fact + w+walk); /// Dinosaur walk point up
glVertex2f(87 + x, 120 * fact + w);
glVertex2f(87 + x, 115 * fact + w);
glVertex2f(87 + x, 110 * fact + w);
glVertex2f(87 + x, 30 * fact + w);
glVertex2f(95 + x, 120 * fact + w);
glVertex2f(95 + x, 35 * fact + w);
glVertex2f(103 + x, 120 * fact + w);
glVertex2f(103 + x, 75 * fact + w);
glVertex2f(103 + x, 65 * fact + w);
glVertex2f(103 + x, 60 * fact + w);
glVertex2f(110 + x, 65 * fact + w);
glVertex2f(110 + x, 60 * fact + w);
glVertex2f(118 + x, 65 * fact + w);
glVertex2f(118 + x, 55 * fact + w);
glVertex2f(112 + x, 120 * fact + w);
glVertex2f(112 + x, 85 * fact + w);
glVertex2f(112 + x, 80 * fact + w);
glVertex2f(112 + x, 75 * fact + w);
glVertex2f(120 + x, 120 * fact + w);
glVertex2f(120 + x, 85 * fact + w);
glVertex2f(120 + x, 80 * fact + w);
glVertex2f(120 + x, 75 * fact + w);
glVertex2f(126 + x, 120 * fact + w);
glVertex2f(126 + x, 85 * fact + w);
glVertex2f(126 + x, 80 * fact + w);
glVertex2f(126 + x, 75 * fact + w);
glVertex2f(135 + x, 120 * fact + w);
glVertex2f(135 + x, 85 * fact + w);
glVertex2f(135 + x, 80 * fact + w);
glVertex2f(135 + x, 75 * fact + w);
glVertex2f(142 + x, 120 * fact + w);
glVertex2f(142 + x, 85 * fact + w);
glVertex2f(150 + x, 120 * fact + w);
glVertex2f(150 + x, 85 * fact + w);
```

```
        glVertex2f(157 + x, 115 * fact + w);
        glVertex2f(157 + x, 85 * fact + w);

    glEnd();
/// check collision between dinosaur and tree
    if(collision(1.0)){
            score=0;
            /// screen become reset
        reset();

    }
    else{

        score=score+1;

    }
     /// dinosaur leg up down
    if( w <=200){
        if(walk==-20 )
            walk = 20;
        else{
            walk = -20;
        }
    }
       /// dinosaur not walk
    else{
        walk = 0;
    }
/// dinosaur body move
    if(flag==1){
        if(w<=1000 ){
            w = w + 8;
        }
        /// dinosaur body not move
        else {
            flag = 0;
        }
    }
    else if(w >= 200 )
        w = w - 8;
    glFlush();
}


void setup(void){
/// basic window background color (light sky blue)
    glClearColor(0.5, 1.0, 1.0, 0.0);
    glMatrixMode(GL_PROJECTION); /// change to projection status
    gluOrtho2D(0.0, 2000, 0.0, 2000); /// Setup drawing x and Y coordinate
}
/// basic openGL structure start
int main( int argc , char** argv ){
    srand(time(NULL));
    glutInit( &argc, argv );
    glutInitDisplayMode( GLUT_SINGLE | GLUT_RGBA );
/// basic openGL structure end
/// screen window size
    glutInitWindowSize( 1230, 650 );
/// screen initial position
    glutInitWindowPosition( 50 , 0 );
/// screen window name
    glutCreateWindow("Dinosaur Game!!");
/// initialize the window screen (coordinate, color)
    setup();
/// calling the main drawing function
    glutDisplayFunc(render);
/// calling the keyboard function (arrow key)
    glutKeyboardFunc(keyInput);
/// calling the special keyboard function
```

```
    glutSpecialFunc(specialKeyInput);
/// render the drawing screen on loop
    glutMainLoop();
}
```