**Lab Taks-6**

|  |
| --- |
| **Question-**  Develop an animation that will change the background color of the window after 20ms. Use at least two different colors. |
| **Graph-** |
| **Code-**  #include <GL/glut.h> #include <stdlib.h> #include <time.h>    float bgColor[3] = {0.0f, 0.0f, 0.0f}; int i = 0; void changeBackgroundColor(int value){     if(i == 0){         bgColor[0] = 1.0f;         bgColor[1] = 0.0f;         bgColor[2] = 0.0f;     }     else if (i == 1){         bgColor[0] = 0.0f;         bgColor[1] = 1.0f;         bgColor[2] = 0.0f;     }     else if (i == 2){         bgColor[0] = 0.0f;         bgColor[1] = 0.0f;         bgColor[2] = 1.0f;     }        i = (i + 1) % 3;     glutTimerFunc(200, changeBackgroundColor, 0);     glutPostRedisplay(); }    void display() {     glClearColor(bgColor[0], bgColor[1], bgColor[2], 1.0f);     glClear(GL\_COLOR\_BUFFER\_BIT);        glutSwapBuffers(); }    void initOpenGL() {     glClearColor(0.0f, 0.0f, 0.0f, 1.0f);     glEnable(GL\_DEPTH\_TEST); }    int main(int argc, char\*\* argv) {     glutInit(&argc, argv);     glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH);     glutInitWindowSize(800, 600);     glutCreateWindow("GLUT Color Change Animation");        initOpenGL();        glutDisplayFunc(display);     glutTimerFunc(200, changeBackgroundColor, 0);        glutMainLoop();        return 0; } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Develop an animation that will call four objects separately, each after 20 ms. |
| **Graph-** |
| **Code-**  **#include <GL/glut.h> #include <iostream> #include <math.h>**    **int objectState = 0;**    **void square() {     glColor3f(1.0f, 0.0f, 0.0f);     glBegin(GL\_QUADS);     glVertex2f(-0.5f, -0.5f);     glVertex2f(0.5f, -0.5f);     glVertex2f(0.5f, 0.5f);     glVertex2f(-0.5f, 0.5f);     glEnd(); }**    **void circle(){     glColor3f(0.0f, 1.0f, 0.0f);     glBegin(GL\_TRIANGLE\_FAN);     glVertex2f(0.0f, 0.0f);     for (int i = 0; i <= 100; i++) {         float angle = 2.0f \* 3.14159f \* i / 100;         glVertex2f(cos(angle) \* 0.5f, sin(angle) \* 0.5f);     }     glEnd(); }**    **void triangle() {     glColor3f(0.0f, 0.0f, 1.0f);     glBegin(GL\_TRIANGLES);     glVertex2f(-0.5f, -0.5f);     glVertex2f(0.5f, -0.5f);     glVertex2f(0.0f, 0.5f);     glEnd(); }**    **void line() {     glColor3f(1.0f, 1.0f, 0.0f);     glBegin(GL\_LINES);     glVertex2f(-0.5f, 0.0f);     glVertex2f(0.5f, 0.0f);     glEnd(); }**    **void display() {     glClear(GL\_COLOR\_BUFFER\_BIT);**    **switch (objectState) {         case 0: square(); break;         case 1: circle(); break;         case 2: triangle(); break;         case 3: line(); break;     }**    **glutSwapBuffers(); }**    **void timer(int value) {     objectState++;     if (objectState > 3) {         objectState = 0;     }     glutPostRedisplay();     glutTimerFunc(200, timer, 0); }**    **void initGL() {     glClearColor(0.0f, 0.0f, 0.0f, 1.0f); }**    **int main(int argc, char\*\* argv) {     glutInit(&argc, argv);     glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);     glutInitWindowSize(500, 500);     glutCreateWindow("Object Animation with GLUT");**    **initGL();**    **glutDisplayFunc(display);     glutTimerFunc(200, timer, 0);     glutMainLoop();**    **return 0; }** |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Develop a code that will have four different objects (keep it simple). The objects will move to the left, right, up and down in a loop. |
| **Graph-** |
| **Code-**  #include <GL/glut.h> #include <iostream> #include <math.h>    float sq\_x = -0.5f; float tr\_x = 0.5f; float cr\_y = 0.5f; float ln\_y = -0.5f;    float speed = 0.01f;    void square() {     glColor3f(1.0f, 0.0f, 0.0f);     glBegin(GL\_QUADS);     glVertex2f(sq\_x - 0.1f, -0.1f);     glVertex2f(sq\_x + 0.1f, -0.1f);     glVertex2f(sq\_x + 0.1f, 0.1f);     glVertex2f(sq\_x - 0.1f, 0.1f);     glEnd(); }    void triangle() {     glColor3f(0.0f, 1.0f, 0.0f);     glBegin(GL\_TRIANGLES);     glVertex2f(tr\_x, -0.2f);     glVertex2f(tr\_x - 0.1f, -0.4f);     glVertex2f(tr\_x + 0.1f, -0.4f);     glEnd(); }    void circle() {     glColor3f(0.0f, 0.0f, 1.0f);     glBegin(GL\_TRIANGLE\_FAN);     glVertex2f(0.0f, cr\_y);     for (int i = 0; i <= 100; i++) {         float angle = 2.0f \* 3.14159f \* i / 100;         glVertex2f(cos(angle) \* 0.1f, cr\_y + sin(angle) \* 0.1f);     }     glEnd(); }    void line() {     glColor3f(1.0f, 1.0f, 0.0f);     glBegin(GL\_LINES);     glVertex2f(-0.2f, ln\_y);     glVertex2f(0.2f, ln\_y);     glEnd(); }    void display() {     glClear(GL\_COLOR\_BUFFER\_BIT);        square();     triangle();     circle();     line();        glutSwapBuffers(); }    void update(int value) {     sq\_x += speed;     if (sq\_x > 1.0f) sq\_x = -1.0f;        tr\_x -= speed;     if (tr\_x < -1.0f) tr\_x = 1.0f;        cr\_y -= speed;     if (cr\_y < -1.0f) cr\_y = 1.0f;        ln\_y += speed;     if (ln\_y > 1.0f) ln\_y = -1.0f;        glutPostRedisplay();     glutTimerFunc(10, update, 0); }    void initGL() {     glClearColor(0.0f, 0.0f, 0.0f, 1.0f); }    int main(int argc, char\*\* argv) {     glutInit(&argc, argv);     glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);     glutInitWindowSize(500, 500);     glutCreateWindow("Moving Objects Animation");        initGL();        glutDisplayFunc(display);     glutTimerFunc(10, update, 0);     glutMainLoop();        return 0; } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Develop a code that will have four different objects (keep it simple). Four different keys will be dedicated each objects. The objects will move to the left, right, up and down in a loop as the keys are pressed individually. |
| **Graph-** |
| **Code-**  **#include <GL/glut.h>**  **#include <iostream>**  **#include <math.h>**  **float sq\_x = -0.5f;**  **float tr\_x = 0.5f;**  **float cr\_y = 0.5f;**  **float ln\_y = -0.5f;**  **bool sq\_mv = false;**  **bool tr\_mv = false;**  **bool cr\_mv = false;**  **bool ln\_mv = false;**  **float speed = 0.01f;**  **void square() {**  **glColor3f(1.0f, 0.0f, 0.0f);**  **glBegin(GL\_QUADS);**  **glVertex2f(sq\_x - 0.1f, -0.1f);**  **glVertex2f(sq\_x + 0.1f, -0.1f);**  **glVertex2f(sq\_x + 0.1f, 0.1f);**  **glVertex2f(sq\_x - 0.1f, 0.1f);**  **glEnd();**  **}**  **void triangle() {**  **glColor3f(0.0f, 1.0f, 0.0f);**  **glBegin(GL\_TRIANGLES);**  **glVertex2f(tr\_x, -0.2f);**  **glVertex2f(tr\_x - 0.1f, -0.4f);**  **glVertex2f(tr\_x + 0.1f, -0.4f);**  **glEnd();**  **}**  **void circle() {**  **glColor3f(0.0f, 0.0f, 1.0f);**  **glBegin(GL\_TRIANGLE\_FAN);**  **glVertex2f(0.0f, cr\_y);**  **for (int i = 0; i <= 100; i++) {**  **float angle = 2.0f \* 3.14159f \* i / 100;**  **glVertex2f(cos(angle) \* 0.1f, cr\_y + sin(angle) \* 0.1f);**  **}**  **glEnd();**  **}**  **void line() {**  **glColor3f(1.0f, 1.0f, 0.0f);**  **glBegin(GL\_LINES);**  **glVertex2f(-0.2f, ln\_y);**  **glVertex2f(0.2f, ln\_y);**  **glEnd();**  **}**  **void display() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **square();**  **triangle();**  **circle();**  **line();**  **glutSwapBuffers();**  **}**  **void update(int value) {**  **if(sq\_mv){**  **sq\_x += speed;**  **if (sq\_x > 1.0f) sq\_x = -1.0f;**  **}**  **if(tr\_mv){**  **tr\_x -= speed;**  **if (tr\_x < -1.0f) tr\_x = 1.0f;**  **}**  **if(cr\_mv){**  **cr\_y -= speed;**  **if (cr\_y < -1.0f) cr\_y = 1.0f;**  **}**  **if(ln\_mv){**  **ln\_y += speed;**  **if (ln\_y > 1.0f) ln\_y = -1.0f;**  **}**  **glutPostRedisplay();**  **glutTimerFunc(10, update, 0);**  **}**  **void handleKey(unsigned char key, int x, int y){**  **if(key == 's' || key == 'S')**  **sq\_mv = !sq\_mv;**  **if(key == 't' || key == 'T')**  **tr\_mv = !tr\_mv;**  **if(key == 'c' || key == 'C')**  **cr\_mv = !cr\_mv;**  **if(key == 'l' || key == 'L')**  **ln\_mv = !ln\_mv;**  **}**  **void initGL() {**  **glClearColor(0.0f, 0.0f, 0.0f, 1.0f);**  **}**  **int main(int argc, char\*\* argv) {**  **glutInit(&argc, argv);**  **glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);**  **glutInitWindowSize(500, 500);**  **glutCreateWindow("Moving Objects Animation");**  **initGL();**  **glutDisplayFunc(display);**  **glutTimerFunc(10, update, 0);**  **glutKeyboardFunc(handleKey);**  **glutMainLoop();**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question-**  Develop a scenario where it will rain and gradually create flood |
| **Graph-** |
| **Code-**  **#include <GL/glut.h>**  **#include <vector>**  **#include <cstdlib>**  **#include <ctime>**  **#include <math.h>**  **struct Raindrop {**  **float x, y;**  **};**  **std::vector<Raindrop> raindrops;**  **float waterLevel = -1.0f;**  **float rainSpeed = 0.02f;**  **float waterRiseSpeed = 0.001f;**  **void drawRaindrop(float x, float y) {**  **glColor3f(0.5f, 0.5f, 1.0f);**  **glBegin(GL\_LINES);**  **glVertex2f(x, y);**  **glVertex2f(x, y - 0.05f);**  **glEnd();**  **}**  **void drawWater() {**  **glColor3f(0.0f, 0.0f, 1.0f);**  **glBegin(GL\_QUADS);**  **glVertex2f(-1.0f, -1.0f);**  **glVertex2f(1.0f, -1.0f);**  **glVertex2f(1.0f, waterLevel);**  **glVertex2f(-1.0f, waterLevel);**  **glEnd();**  **}**  **void drawHouse() {**  **glColor3f(0.8f, 0.5f, 0.2f);**  **glBegin(GL\_QUADS);**  **glVertex2f(-0.5f, -0.5f);**  **glVertex2f(-0.2f, -0.5f);**  **glVertex2f(-0.2f, -0.2f);**  **glVertex2f(-0.5f, -0.2f);**  **glEnd();**  **glColor3f(0.9f, 0.0f, 0.0f);**  **glBegin(GL\_TRIANGLES);**  **glVertex2f(-0.55f, -0.2f);**  **glVertex2f(-0.35f, 0.0f);**  **glVertex2f(-0.15f, -0.2f);**  **glEnd();**  **}**  **void drawCircle(float x, float y, float radius, float r, float g, float b) {**  **glColor3f(r, g, b);**  **glBegin(GL\_TRIANGLE\_FAN);**  **glVertex2f(x, y);**  **for (int i = 0; i <= 100; i++) {**  **float angle = 2.0f \* 3.14159f \* i / 100;**  **float dx = cos(angle) \* radius;**  **float dy = sin(angle) \* radius;**  **glVertex2f(x + dx, y + dy);**  **}**  **glEnd();**  **}**  **void drawTree() {**  **glColor3f(0.5f, 0.25f, 0.1f);**  **glBegin(GL\_QUADS);**  **glVertex2f(0.4f, -0.5f);**  **glVertex2f(0.45f, -0.5f);**  **glVertex2f(0.45f, -0.3f);**  **glVertex2f(0.4f, -0.3f);**  **glEnd();**  **drawCircle(0.425f, -0.25f, 0.07f, 0.0f, 0.8f, 0.0f);**  **drawCircle(0.375f, -0.2f, 0.07f, 0.0f, 0.8f, 0.0f);**  **drawCircle(0.475f, -0.2f, 0.07f, 0.0f, 0.8f, 0.0f);**  **drawCircle(0.425f, -0.15f, 0.07f, 0.0f, 0.8f, 0.0f);**  **drawCircle(0.375f, -0.1f, 0.07f, 0.0f, 0.8f, 0.0f);**  **drawCircle(0.475f, -0.1f, 0.07f, 0.0f, 0.8f, 0.0f);**  **drawCircle(0.425f, -0.05f, 0.07f, 0.0f, 0.8f, 0.0f);**  **}**  **void display() {**  **glClear(GL\_COLOR\_BUFFER\_BIT);**  **drawWater();**  **drawHouse();**  **drawTree();**  **for (const auto& raindrop : raindrops) {**  **drawRaindrop(raindrop.x, raindrop.y);**  **}**  **glutSwapBuffers();**  **}**  **void update(int value) {**  **for (auto& raindrop : raindrops) {**  **raindrop.y -= rainSpeed;**  **if (raindrop.y < -1.0f) {**  **raindrop.y = 1.0f;**  **raindrop.x = (rand() % 200 - 100) / 100.0f;**  **}**  **}**  **if (waterLevel < -0.2f) {**  **waterLevel += waterRiseSpeed;**  **}**  **glutPostRedisplay();**  **glutTimerFunc(16, update, 0);**  **}**  **void initRain() {**  **for (int i = 0; i < 100; ++i) {**  **Raindrop drop;**  **drop.x = (rand() % 200 - 100) / 100.0f;**  **drop.y = (rand() % 200) / 100.0f;**  **raindrops.push\_back(drop);**  **}**  **}**  **void initGL() {**  **glClearColor(0.2f, 0.2f, 0.2f, 1.0f);**  **}**  **int main(int argc, char\*\* argv) {**  **srand(time(0));**  **glutInit(&argc, argv);**  **glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);**  **glutInitWindowSize(500, 500);**  **glutCreateWindow("Rain and Flood Simulation");**  **initGL();**  **initRain();**  **glutDisplayFunc(display);**  **glutTimerFunc(16, update, 0);**  **glutMainLoop();**  **return 0;**  **}** |
| **Output Screenshot (Full Screen)-** |