**Lab Taks-4**

|  |
| --- |
| **Question- 1**  Draw the scenario of a traffic signal using function to represent each object |
| **Graph Plot (Picture)-** |
| **Code-**  #include <windows.h> // for MS Windows  #include <GL/glut.h> // GLUT, include glu.h and gl.h  #include <math.h>  /\* Handler for window-repaint event. Call back when the window first appears and  whenever the window needs to be re-painted. \*/  // 22-46013-1  // MD. SHOHANUR RAHMAN SHOHAN  void circle(float radius, float xc, float yc, float r, float g, float b)  {  glBegin(GL\_POLYGON);// Draw a Red 1x1 Square centered at origin  for(int i=0;i<200;i++)  {  glColor3f(r,g,b);  float pi=3.1416;  float A=(i\*2\*pi)/200;  float r=radius;  float x = r \* cos(A);  float y = r \* sin(A);  glVertex2f(x+xc,y+yc);  }  glEnd();  }  void display() {  glClearColor(0.0f, 0.0f, 0.0f, 0.0f); // Set background color to black and opaque  glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)  glLineWidth(1);  // ################################  // ## ##  // ## S K Y ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.5607,0.7412,0.8902);  glVertex2f(-60.0f, -5.0f);  glVertex2f(-60.0, 60.0f);  glVertex2f(95.0f, 60.0f);  glVertex2f(95.0f, -5.0f);  glEnd();  // ################################  // ## ##  // ## R O A D ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.35f,0.35f,0.35f);  glVertex2f(-60.0f, -30.0f);  glVertex2f(-60.0, -5.0f);  glVertex2f(95.0f, -5.0f);  glVertex2f(95.0f, -30.0f);  glEnd();  // ################################  // ## ##  // ## MOUNTAIN ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.01f,0.2,0.13f);  glVertex2f(-60.0f, -5.0f);  glVertex2f(-60.0, 20.0f);  glVertex2f(-50.0f, 30.0f);  glVertex2f(-20.0f, -5.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.01f,0.2,0.13f);  glVertex2f(-50.0f, -5.0f);  glVertex2f(-20.0, 40.0f);  glVertex2f(80.0f, -5.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.01f,0.2,0.13f);  glVertex2f(-20.0f, -5.0f);  glVertex2f(30.0, 40.0f);  glVertex2f(95.0f, -5.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.01f,0.2,0.13f);  glVertex2f(35.0f, -5.0f);  glVertex2f(95.0, 40.0f);  glVertex2f(95.0f, -5.0f);  glEnd();  // ################################  // ## ##  // ## TRAFFIC LIGHT ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.0f,0.0,0.0f);  glVertex2f(-25.0f, -5.0f);  glVertex2f(-27.0, -5.0f);  glVertex2f(-27.0f, 15.0f);  glVertex2f(-25.0f, 15.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.3f,0.3f,0.3f);  glVertex2f(-23.0f, 15.0f);  glVertex2f(-29.0, 15.0f);  glVertex2f(-29.0f, 27.0f);  glVertex2f(-23.0f, 27.0f);  glEnd();  circle(1.22947, -26.0, 24.0, 1.0,0.0,0.0); //RED  circle(1.22947, -26.0, 20.8, 1.0,1.0,0.0); //YELLOW  circle(1.22947, -26.0, 17.7, 0.0,1.0,0.0); //YELLOW  // ################################  // ## ##  // ## ZEBRA CROSSING ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(1.0f,1.0,1.0f);  glVertex2f(-30.0f, -6.0f);  glVertex2f(-44.0, -6.0f);  glVertex2f(-44.0f, -5.0f);  glVertex2f(-30.0f, -5.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(1.0f,1.0,1.0f);  glVertex2f(-30.0f, -10.0f);  glVertex2f(-44.0, -10.0f);  glVertex2f(-44.0f, -8.0f);  glVertex2f(-30.0f, -8.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(1.0f,1.0,1.0f);  glVertex2f(-30.0f, -14.0f);  glVertex2f(-44.0, -14.0f);  glVertex2f(-44.0f, -12.0f);  glVertex2f(-30.0f, -12.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(1.0f,1.0,1.0f);  glVertex2f(-30.0f, -18.0f);  glVertex2f(-44.0, -18.0f);  glVertex2f(-44.0f, -16.0f);  glVertex2f(-30.0f, -16.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(1.0f,1.0,1.0f);  glVertex2f(-30.0f, -22.0f);  glVertex2f(-44.0, -22.0f);  glVertex2f(-44.0f, -20.0f);  glVertex2f(-30.0f, -20.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(1.0f,1.0,1.0f);  glVertex2f(-30.0f, -26.0f);  glVertex2f(-44.0, -26.0f);  glVertex2f(-44.0f, -24.0f);  glVertex2f(-30.0f, -24.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(1.0f,1.0,1.0f);  glVertex2f(-30.0f, -30.0f);  glVertex2f(-44.0, -30.0f);  glVertex2f(-44.0f, -28.0f);  glVertex2f(-30.0f, -28.0f);  glEnd();  // ################################  // ## ##  // ## C A R ##  // ## ##  // ################################  //BODY  glBegin(GL\_POLYGON);  glColor3f(0.8f,0.8,0.0f);  glVertex2f(50.0f, -10.0f);  glVertex2f(0.0, -10.0f);  glVertex2f(-8.0f, -8.0f);  glVertex2f(-8.0f, -2.0f);  glVertex2f(6.0f, 0.0f);  glVertex2f(12.0f, 8.0f);  glVertex2f(34.0f, 8.0f);  glVertex2f(38.0f, 0.0f);  glVertex2f(50.0f, 0.0f);  glVertex2f(50.0f, -10.0f);  glEnd();  // Window  glBegin(GL\_POLYGON);  glColor3f(0.5f,1.0,1.0f);  glVertex2f(21.3f, 0.0f);  glVertex2f(7.4f, 0.0f);  glVertex2f(12.8f, 7.0f);  glVertex2f(21.3f, 7.0f);  glEnd();  // Window  glBegin(GL\_POLYGON);  glColor3f(0.5f,1.0,1.0f);  glVertex2f(36.5f, 0.0f);  glVertex2f(22.3f, 0.0f);  glVertex2f(22.3f, 7.0f);  glVertex2f(32.2f, 7.0f);  glEnd();  circle(4.24264, 8.0, -10.0, 0.35,0.35,0.35); // WHEEL  circle(3.62243, 8.0, -10.0, 0.0,0.0,0.0); // WHEEL  circle(4.24264, 38.0, -10.0, 0.35,0.35,0.35); // WHEEL  circle(3.62243, 38.0, -10.0, 0.0,0.0,0.0); // WHEEL  glFlush(); // Render now  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  glutInitWindowSize(920, 520);// Set the window's initial width & height  glutCreateWindow("OpenGL Setup Test");  //gluOrtho2D(-0.1,0.7,-0.1,0.3); // Create a window with the given title  //glutInitWindowSize(320, 320);// Set the window's initial width & height  glutDisplayFunc(display);// Register display callback handler for window re-paint  gluOrtho2D(-60,95,-30,60);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-** |

|  |
| --- |
| **Question- 2**  Draw two village scenarios for day and night using function to represent each object |
| **Graph Plot (Picture)-**  **DAY**    **NIGHT** |
| **Code-**  **DAY**  #include <windows.h> // for MS Windows  #include <GL/glut.h> // GLUT, include glu.h and gl.h  #include <math.h>  /\* Handler for window-repaint event. Call back when the window first appears and  whenever the window needs to be re-painted. \*/  // 22-46013-1  // MD. SHOHANUR RAHMAN SHOHAN  void circle(float radius, float xc, float yc, float r, float g, float b)  {  glBegin(GL\_POLYGON);// Draw a Red 1x1 Square centered at origin  for(int i=0;i<200;i++)  {  glColor3f(r,g,b);  float pi=3.1416;  float A=(i\*2\*pi)/200;  float r=radius;  float x = r \* cos(A);  float y = r \* sin(A);  glVertex2f(x+xc,y+yc);  }  glEnd();  }  void display() {  glClearColor(0.0f, 0.0f, 0.0f, 0.0f); // Set background color to black and opaque  glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)  glLineWidth(1);  glBegin(GL\_POLYGON); //sky  glColor3f(0.5607,0.7412,0.8902);  glVertex2f(-25.0f, 15.0f);  glVertex2f(10.0, 15.0f);  glVertex2f(10.0f, 0.0f);  glVertex2f(-25.0f, 0.0f);  glEnd();  circle(1.7592, 4.0, 12.0, 1.0,0.7,0.2);//sun  // ################################  // ## ##  // ## J U N G L E ##  // ## ##  // ################################  circle(1.0, -24.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -22.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -20.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -18.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -16.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -14.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -12.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -10.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 0.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 2.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 4.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 6.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 8.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 10.0, 0.0, 0.01,0.25,0.13);  //Big  circle(1.51329, 7.0, 1.0, 0.01,0.25,0.13);  circle(1.51329, 3.0, 1.0, 0.01,0.25,0.13);  circle(1.51329, -1.5, 1.0, 0.01,0.25,0.13);  circle(1.51329, -20.0, 1.0, 0.01,0.25,0.13);  circle(1.51329, -23.5, 1.0, 0.01,0.25,0.13);  // ################################  // ## ##  // ## G R A S S ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.2f, 0.4, 0.24f);  glVertex2f(-25.0f, -8.0f);  glVertex2f(10.0, -8.0f);  glVertex2f(10.0f, 0.0f);  glVertex2f(-25.0f, 0.0f);  glEnd();  // ################################  // ## ##  // ## T R E E ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.4f, 0.18f, 0.0);  glVertex2f(-14.0f, -3.0f);  glVertex2f(-12.0, -3.0f);  glVertex2f(-12.0f, 3.0f);  glVertex2f(-14.0f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.4f, 0.18f, 0.0);  glVertex2f(-14.0f, -3.0f);  glVertex2f(-15.0, -4.0f);  glVertex2f(-13.6f, -3.3f);  glVertex2f(-13.0f, -4.0f);  glVertex2f(-12.4f, -3.3f);  glVertex2f(-11.0f, -4.0f);  glVertex2f(-12.0f, -3.0f);  glEnd();  circle(3.3801, -16.0, 6.0, 0.0,0.5,0.0);  circle(4.1227, -13.0, 7.5, 0.0,0.5,0.0);  circle(3.3844, -10.0, 6.0, 0.0,0.5,0.0);  circle(3.5737, -12.0, 4.0, 0.0,0.5,0.0);  circle(3.5737, -14.0, 4.0, 0.0,0.5,0.0);  circle(1.4889, -13.0, 1.0, 0.0,0.5,0.0);  // ################################  // ## ##  // ## S T R A W ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.7f, 0.7, 0.0f);  glVertex2f(3.0f, -3.0f);  glVertex2f(0.0, -3.0f);  glVertex2f(0.0f, -1.0f);  glVertex2f(0.4f, 0.4f);  glVertex2f(1.4f, 1.0f);  glVertex2f(1.6f, 1.0f);  glVertex2f(2.6f, 0.4f);  glVertex2f(3.0f, -1.0f);  glEnd();  // ################################  // ## ##  // ## H O U S E ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.44f, 0.4, 0.24f);  glVertex2f(-6.0f, -3.0f);  glVertex2f(-1.0f, -3.0f);  glVertex2f(-1.0f, 1.0f);  glVertex2f(-6.0f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.44f, 0.4, 0.24f);  glVertex2f(-6.0f, -3.0f);  glVertex2f(-8.0f, -2.5f);  glVertex2f(-8.0f, 1.0f);  glVertex2f(-6.0f, 1.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.44f, 0.4, 0.24f);  glVertex2f(-6.0f, 1.0f);  glVertex2f(-8.2f, 1.0f);  glVertex2f(-7.2f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.25f, 0.25f, 0.25f);  glVertex2f(-8.0f, 1.0f);  glVertex2f(-8.2f, 1.0f);  glVertex2f(-7.2f, 3.0f);  glVertex2f(-7.0f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.35f, 0.35f, 0.35f);  glVertex2f(-6.0f, 1.0f);  glVertex2f(-1.0f, 1.0f);  glVertex2f(-2.0f, 3.0f);  glVertex2f(-7.2f, 3.0f);  glEnd();  //DOOR  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-3.0f, -3.0f);  glVertex2f(-4.0f, -3.0f);  glVertex2f(-4.0f, -1.0f);  glVertex2f(-3.0f, -1.0f);  glEnd();  //Window  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-2.5f, -1.0f);  glVertex2f(-1.5f, -1.0f);  glVertex2f(-1.5f, 0.0f);  glVertex2f(-2.5f, 0.0f);  glEnd();  //Window  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-5.5f, -1.0f);  glVertex2f(-5.5f, 0.0f);  glVertex2f(-4.5f, 0.0f);  glVertex2f(-4.5f, -1.0f);  glEnd();  //DOOR  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-6.8f, -2.8f);  glVertex2f(-6.8f, -1.1f);  glVertex2f(-7.2f, -1.0f);  glVertex2f(-7.2f, -2.7f);  glEnd();  glFlush(); // Render now  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  glutInitWindowSize(920, 520);// Set the window's initial width & height  glutCreateWindow("OpenGL Setup Test");  //gluOrtho2D(-0.1,0.7,-0.1,0.3); // Create a window with the given title  //glutInitWindowSize(320, 320);// Set the window's initial width & height  glutDisplayFunc(display);// Register display callback handler for window re-paint  gluOrtho2D(-25,10,-8,15);  glutMainLoop(); // Enter the event-processing loop  return 0;  }  **NIGHT**  #include <windows.h> // for MS Windows  #include <GL/glut.h> // GLUT, include glu.h and gl.h  #include <math.h>  /\* Handler for window-repaint event. Call back when the window first appears and  whenever the window needs to be re-painted. \*/  // 22-46013-1  // MD. SHOHANUR RAHMAN SHOHAN  void circle(float radius, float xc, float yc, float r, float g, float b)  {  glBegin(GL\_POLYGON);// Draw a Red 1x1 Square centered at origin  for(int i=0;i<200;i++)  {  glColor3f(r,g,b);  float pi=3.1416;  float A=(i\*2\*pi)/200;  float r=radius;  float x = r \* cos(A);  float y = r \* sin(A);  glVertex2f(x+xc,y+yc);  }  glEnd();  }  void display() {  glClearColor(0.0f, 0.0f, 0.0f, 0.0f); // Set background color to black and opaque  glClear(GL\_COLOR\_BUFFER\_BIT); // Clear the color buffer (background)  glLineWidth(1);  glBegin(GL\_POLYGON); //sky  glColor3f(0.0f,0.0,0.17f);  glVertex2f(-25.0f, 15.0f);  glVertex2f(10.0, 15.0f);  glVertex2f(10.0f, 0.0f);  glVertex2f(-25.0f, 0.0f);  glEnd();  circle(1.7592, 4.0, 12.0, 1.0,1.0,1.0);//sun  // STAR  circle(0.0981, 1.9, 8.9, 1.0,1.0,1.0);  circle(0.0981, -1.9, 8.3, 1.0,1.0,1.0);  circle(0.0981, -6.1, 8.5, 1.0,1.0,1.0);  circle(0.0981, -9.1, 10.5, 1.0,1.0,1.0);  circle(0.0981, -9.3, 12.1, 1.0,1.0,1.0);  circle(0.0981, -10.9, 13.1, 1.0,1.0,1.0);  circle(0.0981, -3.88, 13.51, 1.0,1.0,1.0);  circle(0.0981, -1.4, 14.3, 1.0,1.0,1.0);  // ################################  // ## ##  // ## J U N G L E ##  // ## ##  // ################################  circle(1.0, -24.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -22.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -20.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -18.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -16.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -14.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -12.0, 0.0, 0.01,0.25,0.13);  circle(1.0, -10.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 0.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 2.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 4.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 6.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 8.0, 0.0, 0.01,0.25,0.13);  circle(1.0, 10.0, 0.0, 0.01,0.25,0.13);  //Big  circle(1.51329, 7.0, 1.0, 0.01,0.25,0.13);  circle(1.51329, 3.0, 1.0, 0.01,0.25,0.13);  circle(1.51329, -1.5, 1.0, 0.01,0.25,0.13);  circle(1.51329, -20.0, 1.0, 0.01,0.25,0.13);  circle(1.51329, -23.5, 1.0, 0.01,0.25,0.13);  // ################################  // ## ##  // ## G R A S S ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.2f, 0.4, 0.24f);  glVertex2f(-25.0f, -8.0f);  glVertex2f(10.0, -8.0f);  glVertex2f(10.0f, 0.0f);  glVertex2f(-25.0f, 0.0f);  glEnd();  // ################################  // ## ##  // ## T R E E ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.4f, 0.18f, 0.0);  glVertex2f(-14.0f, -3.0f);  glVertex2f(-12.0, -3.0f);  glVertex2f(-12.0f, 3.0f);  glVertex2f(-14.0f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.4f, 0.18f, 0.0);  glVertex2f(-14.0f, -3.0f);  glVertex2f(-15.0, -4.0f);  glVertex2f(-13.6f, -3.3f);  glVertex2f(-13.0f, -4.0f);  glVertex2f(-12.4f, -3.3f);  glVertex2f(-11.0f, -4.0f);  glVertex2f(-12.0f, -3.0f);  glEnd();  circle(3.3801, -16.0, 6.0, 0.0,0.5,0.0);  circle(4.1227, -13.0, 7.5, 0.0,0.5,0.0);  circle(3.3844, -10.0, 6.0, 0.0,0.5,0.0);  circle(3.5737, -12.0, 4.0, 0.0,0.5,0.0);  circle(3.5737, -14.0, 4.0, 0.0,0.5,0.0);  circle(1.4889, -13.0, 1.0, 0.0,0.5,0.0);  // ################################  // ## ##  // ## S T R A W ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.7f, 0.7, 0.0f);  glVertex2f(3.0f, -3.0f);  glVertex2f(0.0, -3.0f);  glVertex2f(0.0f, -1.0f);  glVertex2f(0.4f, 0.4f);  glVertex2f(1.4f, 1.0f);  glVertex2f(1.6f, 1.0f);  glVertex2f(2.6f, 0.4f);  glVertex2f(3.0f, -1.0f);  glEnd();  // ################################  // ## ##  // ## H O U S E ##  // ## ##  // ################################  glBegin(GL\_POLYGON);  glColor3f(0.44f, 0.4, 0.24f);  glVertex2f(-6.0f, -3.0f);  glVertex2f(-1.0f, -3.0f);  glVertex2f(-1.0f, 1.0f);  glVertex2f(-6.0f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.44f, 0.4, 0.24f);  glVertex2f(-6.0f, -3.0f);  glVertex2f(-8.0f, -2.5f);  glVertex2f(-8.0f, 1.0f);  glVertex2f(-6.0f, 1.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.44f, 0.4, 0.24f);  glVertex2f(-6.0f, 1.0f);  glVertex2f(-8.2f, 1.0f);  glVertex2f(-7.2f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.25f, 0.25f, 0.25f);  glVertex2f(-8.0f, 1.0f);  glVertex2f(-8.2f, 1.0f);  glVertex2f(-7.2f, 3.0f);  glVertex2f(-7.0f, 3.0f);  glEnd();  glBegin(GL\_POLYGON);  glColor3f(0.35f, 0.35f, 0.35f);  glVertex2f(-6.0f, 1.0f);  glVertex2f(-1.0f, 1.0f);  glVertex2f(-2.0f, 3.0f);  glVertex2f(-7.2f, 3.0f);  glEnd();  //DOOR  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-3.0f, -3.0f);  glVertex2f(-4.0f, -3.0f);  glVertex2f(-4.0f, -1.0f);  glVertex2f(-3.0f, -1.0f);  glEnd();  //Window  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-2.5f, -1.0f);  glVertex2f(-1.5f, -1.0f);  glVertex2f(-1.5f, 0.0f);  glVertex2f(-2.5f, 0.0f);  glEnd();  //Window  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-5.5f, -1.0f);  glVertex2f(-5.5f, 0.0f);  glVertex2f(-4.5f, 0.0f);  glVertex2f(-4.5f, -1.0f);  glEnd();  //DOOR  glBegin(GL\_POLYGON);  glColor3f(0.5f, 0.35, 0.05f);  glVertex2f(-6.8f, -2.8f);  glVertex2f(-6.8f, -1.1f);  glVertex2f(-7.2f, -1.0f);  glVertex2f(-7.2f, -2.7f);  glEnd();  glFlush(); // Render now  }  /\* Main function: GLUT runs as a console application starting at main() \*/  int main(int argc, char\*\* argv) {  glutInit(&argc, argv); // Initialize GLUT  glutInitWindowSize(920, 520);// Set the window's initial width & height  glutCreateWindow("OpenGL Setup Test");  //gluOrtho2D(-0.1,0.7,-0.1,0.3); // Create a window with the given title  //glutInitWindowSize(320, 320);// Set the window's initial width & height  glutDisplayFunc(display);// Register display callback handler for window re-paint  gluOrtho2D(-25,10,-8,15);  glutMainLoop(); // Enter the event-processing loop  return 0;  } |
| **Output Screenshot (Full Screen)-**  **DAY**    **NIGHT** |