CS 352
Computer Graphics & Visualization
Assignment - 01

Name – Anjani Kumar
Roll No - 210001004

Question – 1

CODE:

```
#include <GL/glut.h> //Include the GLUT library for OpenGL functions
#include <bits/stdc++.h> // Include standard C++ library headers
using namespace std;

// Display callback function
void displayCB(void) {
glClear(GL_COLOR_BUFFER_BIT); // Clear the color buffer

// Drawing the bottom most triangle (green)
glColor3f(0, 1, 0); // doing it for green color
glBegin(GL_POLYGON); // adding all the details of the polygon vertex
// coordinates in counterclockwise direction
glVertex2i(220, 260);
glVertex2i(380, 260);
glVertex2i(300, 390);
glEnd();

// Drawing the middle triangle (yellow)
glColor3f(1, 1, 0); // doing it for yellow color
glBegin(GL_POLYGON); // adding all the details of the polygon vertex
// coordinates in counterclockwise direction
glVertex2i(220 + 10, 260 + 50);
glVertex2i(380 - 10, 260 + 50);
glVertex2i(300, 390 + 50);
glEnd();

// Drawing the top most triangle (cyan)
glColor3f(0, 1, 1); // doing it for cyan color
glBegin(GL_POLYGON); // adding all the details of the polygon vertex
// coordinates in counterclockwise direction
glVertex2i(220 + 20, 260 + 100);
glVertex2i(380 - 20, 260 + 100);
glVertex2i(300, 390 + 80);
glEnd();

// Drawing the wood base (rectangle) in brown color
glColor3f(0.52, 0.35, 0.052); // doing it for brown color
glBegin(GL_POLYGON); // adding all the details of the polygon vertex
// coordinates in counterclockwise direction
glVertex2i(280, 60 + 40);
glVertex2i(280, 260);
glVertex2i(320, 260);
glVertex2i(320, 60 + 40);
glEnd();

glFlush(); // Flush OpenGL buffers to display
}

// Main function
int main(int argc, char *argv[]) {
// Initialize GLUT
glutInit(&argc, argv);
glutInitDisplayMode(GLUT_RGB);
glutInitWindowPosition(0, 0);
```

```
glutInitWindowSize(600, 600);
glutCreateWindow("Question_01 - Tree");

// Set up the OpenGL environment
glClearColor(1, 1, 1, 0.0); // Set the background color to white
gluOrtho2D(0, 600, 0, 600); // Set up an orthographic projection

// Register display callback function
glutDisplayFunc(displayCB);

// Enter the GLUT event processing loop
glutMainLoop();

return 0;
}
```

SCREENSHOT OF THE OUTPUT:

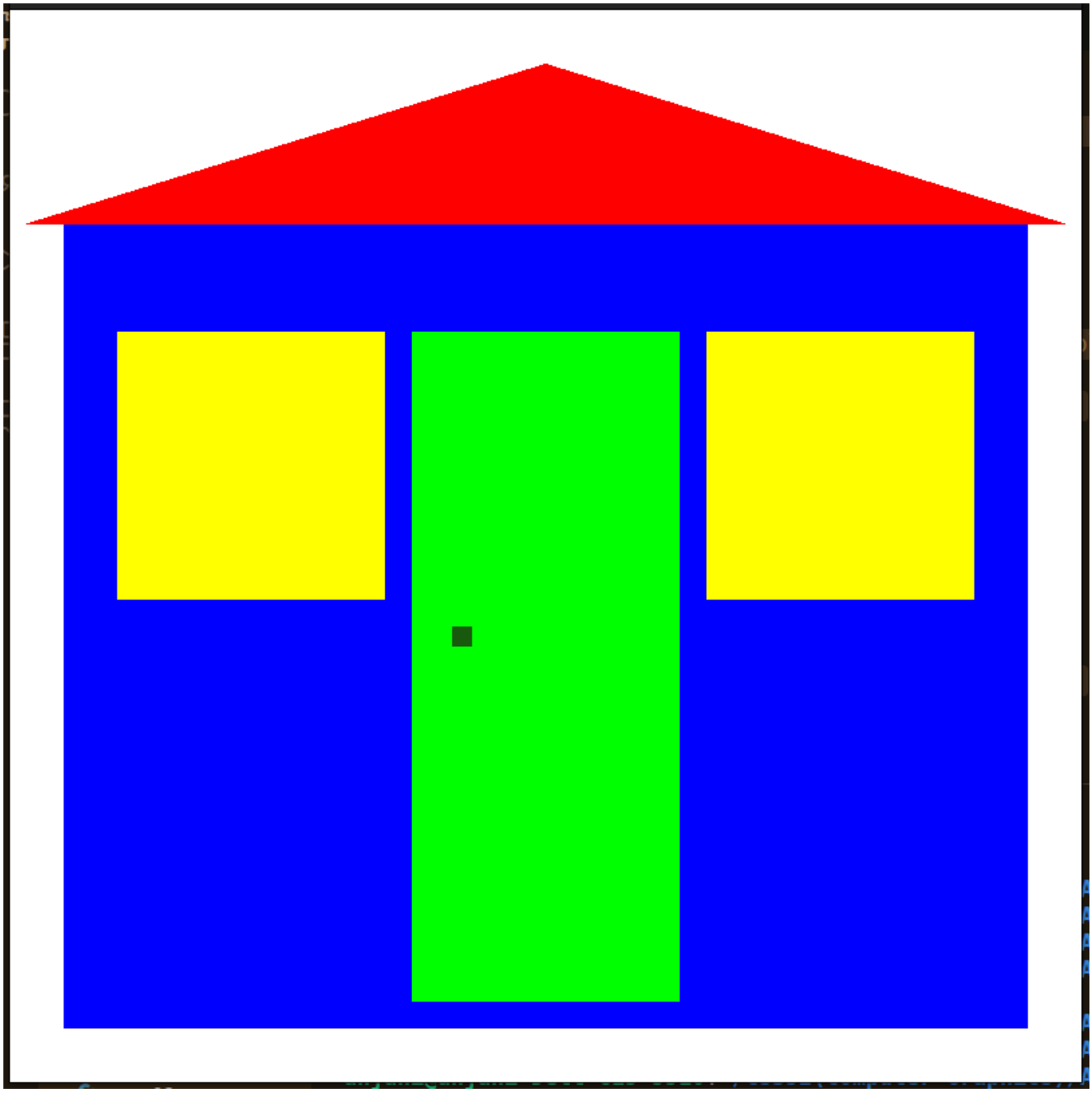
Question 2:

CODE:

```
#include <GL/gl.h>
#include <bits/stdc++.h> // Standard C++ library headers
#include <GL/glut.h>
#include <GL/glu.h>
using namespace std;

// Display callback function
void displayCB(void) {
glClear(GL_COLOR_BUFFER_BIT); // Clear the color buffer

// Drawing the lower rectangular part (blue)
glColor3f(0, 0, 1);
glBegin(GL_POLYGON);
glVertex2i(40, 640);
glVertex2i(40, 40);
glVertex2i(760, 40);
glVertex2i(760, 640);
glEnd();

// Drawing the rectangular door part (green)
glColor3f(0, 1, 0);
glBegin(GL_POLYGON);
glVertex2i(300, 560);
glVertex2i(300, 60);
glVertex2i(500, 60);
glVertex2i(500, 560);
glEnd();

// Drawing the left square window (yellow)
glColor3f(1, 1, 0);
glBegin(GL_POLYGON);
glVertex2i(80, 560);
glVertex2i(80, 360);
glVertex2i(280, 360);
glVertex2i(280, 560);
glEnd();

// Drawing the right square window (yellow)
glColor3f(1, 1, 0);
glBegin(GL_POLYGON);
glVertex2i(520, 560);
glVertex2i(520, 360);
glVertex2i(720, 360);
glVertex2i(720, 560);
glEnd();

// Drawing the rectangular knob of the gate (brown)
glColor3f(0.1, 0.35, 0.05);
glBegin(GL_POLYGON);
glVertex2i(330, 340);
glVertex2i(330, 325);
glVertex2i(345, 325);
```

```
glVertex2i(345, 340);
glEnd();

// Drawing the upper triangular part (red)
glColor3f(1, 0, 0);
glBegin(GL_POLYGON);
glVertex2i(400, 760);
glVertex2i(10, 640);
glVertex2i(790, 640);
glEnd();

glFlush(); // Flush OpenGL buffers to display
}

// Main function
int main(int argc, char *argv[]) {
glutInit(&argc, argv); // Initialize GLUT
glutInitDisplayMode(GLUT_RGB); // Set display mode to RGB
glutInitWindowSize(800, 800); // Set initial window size
glutCreateWindow("Question_02 - House"); // Create window with a title

glClearColor(1, 1, 1, 1); // Set the clear color to white
gluOrtho2D(0, 800, 0, 800); // Set up an orthographic projection

glutDisplayFunc(displayCB); // Register display callback function
glutMainLoop(); // Enter the GLUT event processing loop

return 0;
}
```

SCREENSHOT OF THE OUTPUT:



Question 3:

CODE:

```
#include<GL/glut.h>
#include<math.h>
#include<bits/stdc++.h>
using namespace std;

// Function to draw a circle using center and radius
void Circle(float a, float b, float radius) {
GLfloat xi, yi, angle = 0;

// Draw points around the circle using trigonometry
for (int i = 1 ; i <= 6500 ; i++) {
angle = i * 0.001;
xi = a + radius * cos(angle);
yi = b + radius * sin(angle);
glBegin(GL_POINTS);
glColor3f(0, 0, 0); // Set color to black
glVertex2f(xi, yi); // Draw the point
glEnd();
}
glFlush(); // Flush OpenGL buffers to display
}

// Display callback function
void displayCB(){
glClear(GL_COLOR_BUFFER_BIT); // Clear the color buffer

glColor3f(1, 0, 1); // Set color to magenta
glPointSize(2.0); // Set point size to 2.0
glLineWidth(2.0); // Set line width to 2.0
// Drawing the body of the cycle using lines
glBegin(GL_LINES);
glVertex2f(130, 150);
glVertex2f(300, 150);
glVertex2f(300, 150);
glVertex2f(300, 400);
glVertex2f(300, 400);
glVertex2f(470, 400);
glVertex2f(470, 400);
glVertex2f(470, 150);
glVertex2f(130, 150);
glVertex2f(300, 400);
glVertex2f(300, 150);
glVertex2f(470, 400);
glEnd();

// Draw back wheel
Circle(130, 150, 100);

// Draw front wheel
Circle(470, 150, 100);
```

```
// Drawing the seat
glColor3f(0, 0, 0); // Set color to black
glBegin(GL_POLYGON);
glVertex2f(270, 400);
glVertex2f(330, 400);
glVertex2f(330, 445);
glVertex2f(270, 430);
glEnd();

// Drawing the handle
glColor3f(0, 0, 1); // Set color to blue
glBegin(GL_LINES);
glVertex2f(430, 440);
glVertex2f(510, 360);

glVertex2f(430, 440);
glVertex2f(450, 460);

glVertex2f(510, 360);
glVertex2f(530, 380);
glEnd();

glFlush(); // Flush OpenGL buffers to display
}

// Main function
int main(int argc, char *argv[]) {
glutInit(&argc, argv); // Initialize GLUT
glutInitDisplayMode(GLUT_RGB); // Set display mode to RGB
glutInitWindowPosition(0, 0); // Set initial window position
glutInitWindowSize(600, 600); // Set initial window size
glutCreateWindow("Question_03 - Cycle"); // Create window with a title

glClearColor(1, 1, 1, 0.0); // Set clear color to white
gluOrtho2D(0, 600, 0, 600); // Set up an orthographic projection

glutDisplayFunc(displayCB); // Register display callback function
glutMainLoop(); // Enter the GLUT event processing loop

return 0; // Return 0 to exit main function
}
```

SCREENSHOT OF THE OUTPUT: