#include<GL/glut.h>

#include<fstream>

#include<iostream>

double floor(double thickness)

{

double size = 1.4;

double sizeScaled = 1 / 1.4;

glPushMatrix();

glScaled(1, thickness, 1);

glutSolidCube(size);

glPopMatrix();

glBegin(GL\_QUADS);

glColor3f(255, 0, 255);

glVertex3f(sizeScaled, thickness, -sizeScaled);

glVertex3f(0, thickness, -sizeScaled);

glVertex3f(0, thickness, 0);

glVertex3f(sizeScaled, thickness, 0);

glEnd();

glBegin(GL\_QUADS);

glColor3f(255, 0, 255);

glVertex3f(0, thickness, 0);

glVertex3f(-sizeScaled, thickness, 0);

glVertex3f(-sizeScaled, thickness, sizeScaled);

glVertex3f(0, thickness, sizeScaled);

glEnd();

return 0;

}

// Draws one table leg

void tableLeg(double thick, double len)

{

glPushMatrix();

glScaled(thick, len, thick);

glutSolidCube(1.0);

glPopMatrix();

}

void tableTop(double topWidth, double topThick, double topLength)

{

glPushMatrix();

glScaled(topWidth, topThick, topLength); //define size

glutSolidCube(1.0); //apply above to cube

glPopMatrix();

}

// Creates the table top and table legs

void table(double topWidth, double topThick, double legThick, double legLen)

{

double upperPos = -.25; //x-pos of upper table top

//create upper table top

glPushMatrix();

glColor3ub(0, 255, 0); //green

glTranslated(-.25, 0, 0); //define position upperPos, legLen, 0

tableTop(topWidth / 1.5, topThick, topWidth \* 1.5); //width, thickness, length

glPopMatrix();

double lowerPos = .25; //x-pos of lower table top

//create lower table top

glPushMatrix();

glColor3ub(255, 0, 0); //red

glTranslated(topWidth - .25, 0, 0); // lowerPos, legLen, 0

tableTop(topWidth \* 1.5, topThick, topWidth / 1.5); //width, thickness, length

glPopMatrix();

//create lower table legs

glPushMatrix();

//front-most lower table leg on lower table top

glColor3ub(0, 0, 255); //blue

glTranslated(topWidth\*1.8 - legThick / 2 - .25, topThick - legLen, legThick);

tableLeg(legThick, legLen / 2);

glPopMatrix();

glPushMatrix();

//right-most lower table leg on lower table top

glColor3ub(0, 128, 128); //teal

glTranslated(topWidth\*1.8 - legThick / 2 - .25, topThick - legLen, -legThick);

tableLeg(legThick, legLen / 2);

glPopMatrix();

glPushMatrix();

//left-most lower table leg on upper table top

glColor3ub(160, 80, 40); //brown

glTranslated(-topWidth / 2 + legThick\*1.1 - .25, topThick - legLen, topWidth / 1.5);

tableLeg(legThick, legLen / 2);

glPopMatrix();

glPushMatrix();

//back-most lower table leg on upper table top

glColor3ub(128, 128, 128); //grey

glTranslated(-topWidth / 2 + legThick\*1.1 - .25, topThick - legLen, -topWidth / 1.5);

tableLeg(legThick, legLen / 2);

glPopMatrix();

glPushMatrix();

//create upper table legs

//front-most upper table leg on lower table top

glColor3ub(255, 255, 0); //yellow

glTranslated(topWidth\*1.8-legThick/2 - .25, -topThick\*1.75, legThick);

tableLeg(legThick / 2, legLen / 2);

glPopMatrix();

glPushMatrix();

//right-most upper table leg on lower table top

glColor3ub(255, 0, 255); //pink

glTranslated(topWidth\*1.8 - legThick / 2 - .25, -topThick\*1.75, -legThick);

tableLeg(legThick / 2, legLen / 2);

glPopMatrix();

glPushMatrix();

//left-most upper table leg on upper table top

glColor3ub(148, 0, 211); //purple

glTranslated(-topWidth/2+legThick\*1.1 - .25, -topThick\*1.75, topWidth / 1.5);

tableLeg(legThick / 2, legLen / 2);

glPopMatrix();

glPushMatrix();

//back-most upper table leg on upper table top

glColor3ub(255, 140, 0); //orange

glTranslated(-topWidth / 2 + legThick\*1.1 - .25, -topThick\*1.75, -topWidth / 1.5);

tableLeg(legThick / 2, legLen / 2);

glPopMatrix();

glPushMatrix();

glColor3ub(255, 255, 0); //yellow

glTranslated(0, .14, 0);

glutSolidTeapot(.1);

glPopMatrix();

}

void readFile(double &topWidth, double &topThick, double &legThick, double &legLen)

{

std::ifstream inFile("table.txt");

if (inFile.is\_open())

{

inFile >> topWidth >> topThick >> legThick >> legLen;

}

else

{

std::cout << "Could not open file." << std::endl;

}

}

// Displays the object created

void displaySolid(void)

{

double topWidth, topThick, legThick, legLen;

readFile(topWidth, topThick, legThick, legLen);

// Sets the view

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glOrtho(-1, 1, -1, 1, 0.1, 100.0);

glMatrixMode(GL\_MODELVIEW);

glLoadIdentity();

//gluLookAt(1, 1, 1, 0.0, 0.25, 0.0, 0.0, 1.0, 0.0);

gluLookAt(25.0, 4.0, 2.0, 0.0, 0.25, 0.0, 0.0, 1.0, 0.0);

glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

// Create table

glPushMatrix();

glTranslated(0, .26, 0); //whole table position

table(topWidth, topThick, legThick, legLen); //table dimensions: top width, top thickness, leg thickness, leg length

glPopMatrix();

glPushMatrix();

// Creates the floor

glColor3ub(255, 255, 255); //white

glTranslated(0, -.3, 0); //move floor down

floor(0.05);

glPopMatrix();

glFlush();

}

// Enables specific GL capabilities

void init()

{

glEnable(GL\_LIGHTING); //apply lighting techniques to see shape

glEnable(GL\_LIGHT0); //include light 0 in lighting equation

glEnable(GL\_NORMALIZE); //normalize vectors

glEnable(GL\_DEPTH\_TEST); //enable and update depth buffer

glEnable(GL\_COLOR\_MATERIAL); //enable coloring material

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGBA | GLUT\_DEPTH); //set initial display mode

glutInitWindowSize(640, 480); //specify window size

glutInitWindowPosition(350, 100); //specify window location

glutCreateWindow("Table and Checkerboard Floor"); //give window title

glutDisplayFunc(displaySolid); //set display callback for current window

init();

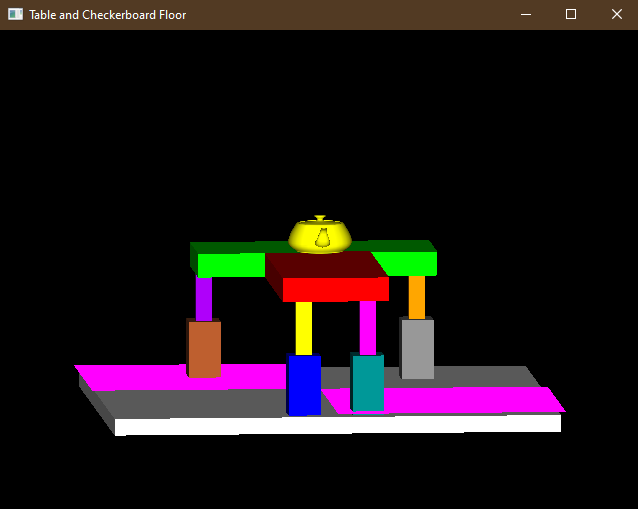
glClearColor(0.f, 0.f, 0.f, 1.f); //set clear color to black

glViewport(0, 0, 640, 480); //defines where to render on screen

glutMainLoop();

return 0;

}



A picture containing LEGO, toy

Description automatically generated