**b) Complete Integration Program**

**#include<stdlib.h>**

**#include<GL/glut.h>**

**#include<stdio.h>**

**#include<math.h>**

**#include"1.c"**

**#include"2.c"**

**#include"3.c"**

**#include"4.c"**

**#include"5.c"**

**#include"6.c"**

**#include"teapot.c"**

**#include"8.c"**

**#include"9.c"**

**#include"10.c"**

**#include"stroke.c"**

**void Write1(char \*string) //to write the string passed as parameter**

**{**

**while(\*string)**

**glutBitmapCharacter(GLUT\_BITMAP\_TIMES\_ROMAN\_24, \*string++);**

**}**

**void Write2(char \*string) //to write the string passed as parameter**

**{**

**while(\*string)**

**glutBitmapCharacter(GLUT\_BITMAP\_HELVETICA\_18, \*string++);**

**}**

**void OnMouseClick(int button, int state, int x, int y)**

**{**

**if (button == GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN) //if left button of the mouse is pressed**

**{**

**if(x>25&&x<275&&y>15&&y<42) //if position of mouse lies between the first button coordinates**

**{**

**glBegin(GL\_POLYGON); //to draw the polygon using the following coordinates**

**glColor3f (0.9,0.9, 0.9); //set the color**

**glVertex2f(-0.45,0.85);**

**glVertex2f(-0.95,0.85);**

**glVertex2f(-0.95,0.95);**

**glVertex2f(-0.45,0.95);**

**glEnd(); //to end drawing the polygon**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB|GLUT\_DEPTH); //define the display mode**

**glutInitWindowSize(700,600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("3D Gasket"); //create the window**

**glutDisplayFunc(display2); //for display callback**

**init();**

**glEnable(GL\_DEPTH\_TEST); //enables an openGL feature**

**glClearColor(1.0,1.0,1.0,1.0); //set the present RGBA clear color**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>69&&y<96) //if position of mouse lies between the second button coordinates**

**{**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB|GLUT\_DEPTH);**

**glutInitWindowSize(700,600);**

**glutInitWindowPosition(400,100);**

**glutCreateWindow("teapot");**

**glutDisplayFunc(displaysolid);**

**glEnable(GL\_LIGHTING);**

**glEnable(GL\_LIGHT0);**

**glShadeModel(GL\_SMOOTH);**

**glEnable(GL\_DEPTH\_TEST);**

**glEnable(GL\_NORMALIZE);**

**glClearColor(0.1,0.1,0.1,0.0);**

**glViewport(0,0,640,480);**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>123&&y<150) //if position of mouse lies between the third button coordinates**

**{ glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB | GLUT\_DEPTH); //define the display mode**

**glutInitWindowSize(700, 600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("Color Cube"); //create the window**

**glutReshapeFunc(myReshape); //for myReshape callback**

**glutDisplayFunc(display4); //for display callback**

**glutIdleFunc(spinCube); //for spinCube callback**

**glutMouseFunc(mouse); //for mouse callback**

**glEnable(GL\_DEPTH\_TEST); //Enable hidden-surface removal**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>177&&y<204) //if position of mouse lies between the fourth button coordinates**

**{**

**theta1=theta1\*(3.14/180); //specifies the angle of rotation**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //define the display mode**

**glutInitWindowSize(700,600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("House"); //create the window**

**glutDisplayFunc(display1); //for display callback**

**myinit();**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>231&&y<258) //if position of mouse lies between the fourth button coordinates**

**{**

**xz=50;yz=50;**

**xo=110;yo=110;**

**xmin=50;ymin=50;**

**xmax=100;ymax=100;**

**xvmin=200,yvmin=200;**

**xvmax=300;yvmax=300;**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //define the display mode**

**glutInitWindowSize(700,600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("Cohen SutherLand"); //create the window**

**glutDisplayFunc(display5); //for display callback**

**myinit1();**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>285&&y<312) //if position of mouse lies between the fifth button coordinates**

**{**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //define the display mode**

**glutInitWindowSize(700,600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("Filling a Polygon using Scan-line Algorithm"); //create the window**

**glutDisplayFunc(display6); //for display callback**

**myinit2();**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>339&&y<366) //if position of mouse lies between the sixth button coordinates**

**{**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //define the display mode**

**glutInitWindowSize(700,600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("Rectangular Mesh"); //create the window**

**glutDisplayFunc(display7); //for display callback**

**init2();**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>393&&y<420) //if position of mouse lies between the sixth button coordinates**

**{**

**glutInitDisplayMode(GLUT\_DOUBLE|GLUT\_RGB|GLUT\_DEPTH); //define the display mode**

**glutInitWindowSize(700,600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("Colorcube Viewer"); //create the window**

**glutReshapeFunc(myReshape8); //for myReshape callback**

**glutDisplayFunc(disp8); //for display callback**

**glutMouseFunc(mouse8); //for mouse callback**

**glutKeyboardFunc(keys8); //for keyboard callback**

**glEnable(GL\_DEPTH\_TEST); //enables an openGL feature**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>447&&y<474) //if position of mouse lies between the seventh button coordinates**

**{**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //define the display mode**

**glutInitWindowPosition(400,100); //establish the window position**

**glutInitWindowSize(700,600); //choose the window size**

**glutCreateWindow("Cylinder & ParralelPiped"); //create the window**

**init3();**

**glutDisplayFunc(display9); //for display callback**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>501&&y<528) //if position of mouse lies between the eighth button coordinates**

**{**

**glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB); //define the display mode**

**glutInitWindowSize(700,600); //choose the window size**

**glutInitWindowPosition(400,100); //establish the window position**

**glutCreateWindow("Lian-Barsky"); //create the window**

**glutDisplayFunc(disp2); //for display callback**

**init5();**

**glutMainLoop();**

**}**

**if(x>25&&x<275&&y>550&&y<580)**

**{**

**int i, submenu;**

**/\* for (i = 1; i < argc; i++) {**

**if (!strcmp(argv[i], "-mono")) {**

**font = GLUT\_STROKE\_MONO\_ROMAN;**

**}**

**}\*/**

**glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);**

**glutInitWindowSize(600, 600);**

**glutCreateWindow("anti-aliased stroke font");**

**glMatrixMode(GL\_PROJECTION);**

**glLoadIdentity();**

**gluOrtho2D(0, 2000, 0, 2000);**

**glMatrixMode(GL\_MODELVIEW);**

**glEnable(GL\_LINE\_SMOOTH);**

**glEnable(GL\_BLEND);**

**glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA);**

**glLineWidth(3.0);**

**glTranslatef(1000, 1000, 0);**

**glClearColor(0.0, 0.0, 0.0, 1.0);**

**glColor3f(1.0, 1.0, 1.0);**

**glutDisplayFunc(display11);**

**glutIdleFunc(tick);**

**submenu = glutCreateMenu(selectMessage);**

**glutAddMenuEntry("abc", 1);**

**glutAddMenuEntry("ABC", 2);**

**glutCreateMenu(selectFont);**

**glutAddMenuEntry("Roman", 0);**

**glutAddMenuEntry("Mono Roman", 1);**

**glutAddSubMenu("Messages", submenu);**

**glutAttachMenu(GLUT\_RIGHT\_BUTTON);**

**glutMainLoop();**

**}**

**}**

**}**

**void display()**

**{**

**glClear(GL\_COLOR\_BUFFER\_BIT); //clears the color buffer**

**glColor3f (0.45,0.0,0.45); //set the color**

**glBegin(GL\_POLYGON); //to draw the polygon using the following coordinates**

**glVertex2f(-0.4,-1.0);**

**glVertex2f(-1.0,-1.0);**

**glVertex2f(-1.0,1.0);**

**glVertex2f(-0.4,1.0);**

**glEnd(); //to end drawing the polygon**

**glBegin(GL\_POLYGON); //to draw the polygon using the following coordinates**

**glColor3f (0.7,0.65,0.8); //set the color**

**glVertex2f(-0.4,1.0);**

**glColor3f(0.95,0.89,1); //set the color**

**glVertex2f(1.0,1.0);**

**glColor3f (0.7,0.65,0.8); //set the color**

**glVertex2f(1.0,-1.0);**

**glColor3f(0.53,0.46,0.67); //set the color**

**glVertex2f(-0.4,-1.0);**

**glEnd();**

**glColor3f (0.45,0.0,0.45);**

**glBegin(GL\_LINE\_LOOP); //to draw the border**

**glVertex2f(-0.3,0.9);**

**glVertex2f(0.9,0.9);**

**glVertex2f(0.9,-0.9);**

**glVertex2f(-0.3,-0.9);**

**glEnd();**

**glBegin(GL\_LINE\_LOOP); //to draw the border**

**glVertex2f(-0.28,0.87);**

**glVertex2f(0.88,0.87);**

**glVertex2f(0.88,-0.87);**

**glVertex2f(-0.28,-0.87);**

**glEnd();**

**glColor3f(0.3,0.1,0.4); //set the color**

**glRasterPos2f(-0.255,0.67); //to provide position for rasterization**

**Write1("COMPUTER GRAPHICS PROJECT WITH OPENGL");**

**glBegin(GL\_LINES); //to underline**

**glVertex2f(-0.255,0.663);**

**glVertex2f(0.855,0.663);**

**glEnd();**

**glColor3f(0,0.2,0.4); //set the color**

**glRasterPos2f(0.02,0.47); //to provide position for rasterization**

**Write1("A MENU BASED PACKAGE");**

**glBegin(GL\_LINES); //to underline**

**glVertex2f(0.02,0.463);**

**glVertex2f(0.64,0.463);**

**glEnd();**

**glColor3f(0,0.2,0.7); //set the color**

**glRasterPos2f(0.08,0.17); //to provide position for rasterization**

**Write2(" BY:");**

**glColor3f(0.3,0.2,0.4); //set the color**

**glRasterPos2f(0.08,-0.07); //to provide position for rasterization**

**Write1("NAME :AARSI");**

**glRasterPos2f(0.08,-0.37);**

**Write1("REG NO:14GAEI6001");**

**glRasterPos2f(0.08,-0.47);**

**Write1("CLASS:V SEM ISE");**

**glColor3f(0.8,0.1,0.2); //set the color**

**glRasterPos2f(0.08,-0.67); //to provide position for rasterization**

**Write2(" GUIDE:");**

**glRasterPos2f(0.08,-0.77); //to provide position for rasterization**

**Write1("Mrs.VIMALA H S");**

**glRasterPos2f(-0.23,-0.87); //to provide position for rasterization**

**GLUquadricObj \* quadricObj; //Creates a quadratic object**

**quadricObj = gluNewQuadric(); //Sets a pointer to a new quadratic object**

**gluQuadricDrawStyle(quadricObj, GLU\_FILL);**

**glPushMatrix();**

**glColor3f (0.4,0.3,0.8); //Sets ellipse color**

**glTranslatef(-0.7,0.91,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.89,0.88); //to provide position for rasterization**

**Write1("TETRAHEDRON");**

**glColor3f (0.4,0.4,0.7); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,0.74,0.0); //Translates the ellipse**

**glScalef(0.35,0.089,0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.81,0.69); //to provide position for rasterization**

**Write1("TEAPOT");**

**glColor3f (0.5,0.4,0.6); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,0.56,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.85,0.54); //to provide position for rasterization**

**Write1("COLOR CUBE");**

**glColor3f (0.6,0.4,0.5); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,0.37,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.79,0.35); //to provide position for rasterization**

**Write1("HOUSE");**

**glColor3f (0.7,0.3,0.4); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,0.19,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.95,0.17); //to provide position for rasterization**

**Write1("COHEN SUTHERLAND");**

**glColor3f (0.8,0.45,0.0); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,-0.71,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.88,-0.73); //to provide position for rasterization**

**Write1("LIANG-BARSKY");**

**glColor3f (0.9,0.4,0.1); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,-0.53,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Translates the ellipse**

**glRasterPos2f(-0.97,-0.55); //to provide position for rasterization**

**Write1("CYLINDER AND IIPIPED");**

**glColor3f (0.85,0.3,0.1); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,-0.35,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.97,-0.37); //to provide position for rasterization**

**Write1("PERSPECTIVE VIEWING");**

**glColor3f (0.85,0.2,0.1); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,-0.17,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.77,-0.18); //to provide position for rasterization**

**Write1("MESH");**

**glColor3f (0.8,0.1,0.1); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,0.01,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.90,0.0); //to provide position for rasterization**

**Write1("SCANLINE AREA");**

**glColor3f (0.8,0.45,0.0); //Sets ellipse color**

**glPushMatrix();**

**glTranslatef(-0.7,-0.89,0.0); //Translates the ellipse**

**glScalef(0.35, 0.089, 0.0); //Scales the ellipse**

**gluDisk(quadricObj, 0.0,0.75, 100, 100);**

**glPopMatrix();**

**glColor3f(1.0,1.0,1.0); //Sets ellipse color**

**glRasterPos2f(-0.86,-0.90); //to provide position for rasterization**

**Write1("STROKE");**

**glutSwapBuffers();**

**glFlush();**

**}**

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

**{**

**glutInit(&argc,argv); //initializes GLUT**

**glutInitDisplayMode(GLUT\_DOUBLE|GLUT\_RGB); //defines the display mode**

**glutInitWindowSize(1000,600); //choose the window size**

**glutInitWindowPosition(100,100); //establish the window position**

**glutCreateWindow("PROJECT MENU CHART"); //create the window**

**glutMouseFunc(OnMouseClick); //for mouse callback**

**glutDisplayFunc(display); //for display callback**

**glutMainLoop();**

**}**