#include <GL/glut.h> #include <math.h> #include <stdlib.h>

static float angle=0.0,ratio; static float x=0.0f,y=1.75f,z=5.0f;

static float lx=0.10f,ly=0.10f,lz=-1.0f;

static GLint carr\_display\_list,house\_display\_list;

float theta=0.01,fxincr=0.1,fzincr=0,temp,theta1,fx=-10,fz=80; int xxxx=0,yyyy=0,kk=0,housevisible=0,movecarvar=0;

int a[36]={55,97,44,152,55,171,108,86,168,99,147,207,238,55,233,167,105,80,134,29

,253,130,32,240,110,199,224,121,93,199,180,61,110,251,77,237};

int b[36]={102,194,110,152,153,184,137,113,55,138,104,43,240,255,203,8,100,53,88, 64,127,64,87,5,2,144,211,128,10,89,27,11,175,185,157,241};

int c[36]={159,243,133,253,233,228,141,18,46,195,75,52,253,204,169,30,78,94,68,11 7,4,2,33,12,2,25,195,76,26,54,98,103,205,173,65,242};

void changeSize(int w, int h)

{

if(h == 0) // Prevent a divide by zero, when window is too short

// (you cant make a window of zero width).

h = 1;

ratio = 1.0f \* w / h; // Reset the coordinate system before modifying

glMatrixMode(GL\_PROJECTION); glLoadIdentity();

glViewport(0, 0, w, h); // Set the viewport to be the entire window gluPerspective(45,ratio,1,1000);

glMatrixMode(GL\_MODELVIEW); glLoadIdentity();

gluLookAt(x, y, z,x + lx,y + ly,z + lz,0.0f,1.0f,0.0f);

}

void drawcarr()

{

glTranslatef(.0,0.8,0.0);

glEnable(GL\_BLEND); //TRANCPARENCY1

glBlendFunc(GL\_ONE, GL\_ZERO);//TRANCPARENCY2 glBegin(GL\_LINE\_LOOP);

glVertex3f(-1.12,-.48,0.7);//a

glVertex3f(-0.86,-.48,0.7);//b

glVertex3f(-.74,-0.2,0.7);//c

glVertex3f(-.42,-.2,0.7);//d

glVertex3f(-0.3,-.48,0.7);//e

glVertex3f(.81,-0.48,0.7);//f

glVertex3f(.94,-0.2,0.7);//g

glVertex3f(1.24,-.2,0.7);//h

glVertex3f(1.38,-.48,0.7);//i

glVertex3f(1.52,-.44,0.7);//j glVertex3f(1.52,.14,0.7);//k glVertex3f(1.14,0.22,0.7);//l glVertex3f(0.76,.22,0.7);//m glVertex3f(.52,0.56,0.7);//n glVertex3f(-0.1,0.6,0.7);//0

glVertex3f(-1.02,0.6,0.7);//p

glVertex3f(-1.2,0.22,0.7);//q

glVertex3f(-1.2,-.28,0.7);//r glEnd();

glBegin(GL\_LINE\_LOOP);

glVertex3f(-1.12,-.48,-0.7);//a'

glVertex3f(-0.86,-.48,-0.7);//b'

glVertex3f(-.74,-0.2,-0.7);//c'

glVertex3f(-.42,-.2,-0.7);//d'

glVertex3f(-0.3,-.48,-0.7);//e'

glVertex3f(.81,-0.48,-0.7);//f'

glVertex3f(.94,-0.2,-0.7);//g'

glVertex3f(1.24,-.2,-0.7);//h'

glVertex3f(1.38,-.48,-0.7);//i'

glVertex3f(1.52,-.44,-0.7);//j'

glVertex3f(1.52,.14,-0.7);//k'

glVertex3f(1.14,0.22,-0.7);//l'

glVertex3f(0.76,.22,-0.7);//m'

glVertex3f(.52,0.56,-0.7);//n'

glVertex3f(-0.1,0.6,-0.7);//o'

glVertex3f(-1.02,0.6,-0.7);//p'

glVertex3f(-1.2,0.22,-0.7);//q'

glVertex3f(-1.2,-.28,-0.7);//r' glEnd();

glBegin(GL\_LINES);

glVertex3f(-1.12,-.48,0.7);//a

glVertex3f(-1.12,-.48,-0.7);//a'

glVertex3f(-0.86,-.48,0.7);//b

glVertex3f(-0.86,-.48,-0.7);//b'

glVertex3f(-.74,-0.2,0.7);//c

glVertex3f(-.74,-0.2,-0.7);//c'

glVertex3f(-.42,-.2,0.7);//d

glVertex3f(-.42,-.2,-0.7);//d'

glVertex3f(-0.3,-.48,0.7);//e

glVertex3f(-0.3,-.48,-0.7);//e'

glVertex3f(.81,-0.48,0.7);//f

glVertex3f(.81,-0.48,-0.7);//f'

glVertex3f(.94,-0.2,0.7);//g

glVertex3f(.94,-0.2,-0.7);//g'

glVertex3f(1.24,-.2,0.7);//h

glVertex3f(1.24,-.2,-0.7);//h'

glVertex3f(1.38,-.48,0.7);//i

glVertex3f(1.38,-.48,-0.7);//i'

glVertex3f(1.52,-.44,0.7);//j

glVertex3f(1.52,-.44,-0.7);//j' glVertex3f(1.52,.14,0.7);//k glVertex3f(1.52,.14,-0.7);//k' glVertex3f(1.14,0.22,0.7);//l glVertex3f(1.14,0.22,-0.7);//l' glVertex3f(0.76,.22,0.7);//m glVertex3f(0.76,.22,-0.7);//m' glVertex3f(.52,0.56,0.7);//n glVertex3f(.52,0.56,-0.7);//n'

glVertex3f(-0.1,0.6,0.7);//0

glVertex3f(-0.1,0.6,-0.7);//o'

glVertex3f(-1.02,0.6,0.7);//p

glVertex3f(-1.02,0.6,-0.7);//p'

glVertex3f(-1.2,0.22,0.7);//q

glVertex3f(-1.2,0.22,-0.7);//q'

glVertex3f(-1.2,-.28,0.7);//r

glVertex3f(-1.2,-.28,-0.7);//r' glEnd();

glBegin(GL\_POLYGON); // top filling glVertex3f(-0.1,0.6,0.7);//o

glVertex3f(-0.1,0.6,-0.7);//o'

glVertex3f(-1.02,0.6,-0.7);//p'

glVertex3f(-1.02,0.6,0.7);//p glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-0.1,0.6,0.7);//o

glVertex3f(-0.1,0.6,-0.7);//o'

glVertex3f(.52,0.56,-0.7);//n' glVertex3f(.52,0.56,0.7);//n

glEnd();

glBegin(GL\_POLYGON); //back filling glVertex3f(-1.2,0.22,0.7);//q

glVertex3f(-1.2,0.22,-0.7);//q'

glVertex3f(-1.2,-.28,-0.7);//r'

glVertex3f(-1.2,-.28,0.7);//r glEnd();

glBegin(GL\_POLYGON); glVertex3f(1.52,.14,0.7);//k glVertex3f(1.14,0.22,0.7);//l glVertex3f(1.14,0.22,-0.7);//l'

glVertex3f(1.52,.14,-0.7);//k' glEnd();

glBegin(GL\_POLYGON); glVertex3f(0.76,.22,0.7);//m glVertex3f(0.76,.22,-0.7);//m'

glVertex3f(1.14,0.22,-0.7);//l' glVertex3f(1.14,0.22,0.7);//l

glEnd(); glBegin(GL\_POLYGON);

glVertex3f(-1.12,-.48,0.7);//a

glVertex3f(-0.86,-.48,0.7);//b

glVertex3f(-.74,-0.2,0.7);//c glVertex3f(-0.64,0.22,0.7);//cc glVertex3f(-1.08,0.22,0.7);//dd glVertex3f(-1.2,0.22,0.7);//q

glVertex3f(-1.2,-.28,0.7);//r glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-.74,-0.2,0.7);//c glVertex3f(-0.64,0.22,0.7);//cc glVertex3f(-0.5,0.22,0.7);//hh glVertex3f(-0.5,-0.2,0.7);//pp

glEnd(); glBegin(GL\_POLYGON);

glVertex3f(0.0,0.22,0.7);//gg glVertex3f(1.14,0.22,0.7);//l glVertex3f(1.24,-.2,0.7);//h

glVertex3f(0.0,-0.2,0.7);//oo glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-1.12,-.48,-0.7);//a'

glVertex3f(-0.86,-.48,-0.7);//b'

glVertex3f(-.74,-0.2,-0.7);//c'

glVertex3f(-0.64,0.22,-0.7);//cc'

glVertex3f(-1.08,0.22,-0.7);//dd'

glVertex3f(-1.2,0.22,-0.7);//q'

glVertex3f(-1.2,-.28,-0.7);//r' glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-.74,-0.2,-0.7);//c'

glVertex3f(-0.64,0.22,-0.7);//cc'

glVertex3f(-0.5,0.22,-0.7);//hh'

glVertex3f(-0.5,-0.2,-0.7);//pp' glEnd();

glBegin(GL\_POLYGON);

glVertex3f(0.0,0.22,-0.7);//gg'

glVertex3f(1.14,0.22,-0.7);//l'

glVertex3f(1.24,-.2,-0.7);//h'

glVertex3f(0.0,-0.2,-0.7);//oo' glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-1.2,0.22,0.7);//q glVertex3f(-1.08,0.22,0.7);//dd glVertex3f(-0.98,0.5,0.7);//aa glVertex3f(-1.02,0.6,0.7);//p

glEnd(); glBegin(GL\_POLYGON);

glVertex3f(-1.02,0.6,0.7);//p glVertex3f(-0.98,0.5,0.7);//aa glVertex3f(0.44,0.5,0.7);//jj glVertex3f(.52,0.56,0.7);//n glVertex3f(-0.1,0.6,0.7);//0

glEnd(); glBegin(GL\_POLYGON);

glVertex3f(-0.64,0.5,0.7);//bb glVertex3f(-0.64,0.22,0.7);//cc glVertex3f(-0.5,0.22,0.7);//hh glVertex3f(-0.5,0.5,0.7);//ee

glEnd(); glBegin(GL\_POLYGON);

glVertex3f(0.0,0.5,0.7);//ff glVertex3f(0.0,0.22,0.7);//gg glVertex3f(0.12,0.22,0.7);//ll glVertex3f(0.12,0.5,0.7);//ii

glEnd(); glBegin(GL\_POLYGON);

glVertex3f(.52,0.56,0.7);//n glVertex3f(0.44,0.5,0.7);//jj glVertex3f(0.62,0.22,0.7);//kk glVertex3f(0.76,.22,0.7);//m

glEnd(); glBegin(GL\_POLYGON);

glVertex3f(-.42,-.2,0.7);//d

glVertex3f(.94,-0.2,0.7);//g

glVertex3f(.81,-0.48,0.7);//f

glVertex3f(-0.3,-.48,0.7);//e glEnd();

glBegin(GL\_POLYGON); glVertex3f(1.14,0.22,0.7);//l glVertex3f(1.52,.14,0.7);//k glVertex3f(1.52,-.44,0.7);//j

glVertex3f(1.38,-.48,0.7);//i

glVertex3f(1.24,-.2,0.7);//h glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-1.2,0.22,-0.7);//q'

glVertex3f(-1.08,0.22,-0.7);//dd'

glVertex3f(-0.98,0.5,-0.7);//aa'

glVertex3f(-1.02,0.6,-0.7);//p' glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-1.02,0.6,-0.7);//p'

glVertex3f(-0.98,0.5,-0.7);//aa'

glVertex3f(0.44,0.5,-0.7);//jj'

glVertex3f(.52,0.56,-0.7);//n'

glVertex3f(-0.1,0.6,-0.7);//0' glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-0.64,0.5,-0.7);//bb'

glVertex3f(-0.64,0.22,-0.7);//cc'

glVertex3f(-0.5,0.22,-0.7);//hh'

glVertex3f(-0.5,0.5,-0.7);//ee' glEnd();

glBegin(GL\_POLYGON); glVertex3f(0.0,0.5,-0.7);//ff'

glVertex3f(0.0,0.22,-0.7);//gg'

glVertex3f(0.12,0.22,-0.7);//ll'

glVertex3f(0.12,0.5,-0.7);//ii' glEnd();

glBegin(GL\_POLYGON);

glVertex3f(.52,0.56,-0.7);//n'

glVertex3f(0.44,0.5,-0.7);//jj'

glVertex3f(0.62,0.22,-0.7);//kk'

glVertex3f(0.76,.22,-0.7);//m' glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-.42,-.2,-0.7);//d'

glVertex3f(.94,-0.2,-0.7);//g'

glVertex3f(.81,-0.48,-0.7);//f'

glVertex3f(-0.3,-.48,-0.7);//e' glEnd();

glBegin(GL\_POLYGON); glVertex3f(1.14,0.22,-0.7);//l'

glVertex3f(1.52,.14,-0.7);//k'

glVertex3f(1.52,-.44,-0.7);//j'

glVertex3f(1.38,-.48,-0.7);//i'

glVertex3f(1.24,-.2,-0.7);//h' glEnd();

glBegin(GL\_POLYGON); // door1 body- rear, near glVertex3f(-0.5,0.22,0.7);//hh glVertex3f(0.0,0.22,0.7);//gg glVertex3f(0.0,-0.2,0.7);//oo

glVertex3f(-0.5,-0.2,0.7);//pp glEnd();

glBegin(GL\_POLYGON); // door body- rear, far glVertex3f(-0.5,0.22,-0.7);//hh'

glVertex3f(0.0,0.22,-0.7);//gg'

glVertex3f(0.0,-0.2,-0.7);//oo'

glVertex3f(-0.5,-0.2,-0.7);//pp' glEnd();

glBegin(GL\_POLYGON); // door2 body- near, driver glVertex3f(0.12,0.22,0.7);//ll glVertex3f(0.62,0.22,0.7);//kk glVertex3f(0.62,-0.2,0.7);//mm glVertex3f(0.12,-0.2,0.7);//nn

glEnd();

glBegin(GL\_POLYGON); // door2 body- far, driver glVertex3f(0.12,0.22,-0.7);//ll'

glVertex3f(0.62,0.22,-0.7);//kk'

glVertex3f(0.62,-0.2,-0.7);//mm'

glVertex3f(0.12,-0.2,-0.7);//nn' glEnd();

glBegin(GL\_POLYGON);//front\*\* glVertex3f(1.52,.14,0.7);//k glVertex3f(1.52,.14,-0.7);//k'

glVertex3f(1.52,-.44,-0.7);//j'

glVertex3f(1.52,-.44,0.7);//j glEnd();

glTranslatef(-.58,-.52,0.7); //translate to 1st tyre glColor3f(0.09,0.09,0.09); // tyre color\*\*\*\*\*\*\*\* glutSolidTorus(0.12f, .14f, 10, 25); glTranslatef(1.68,0.0,0.0); //translate to 2nd tyre glutSolidTorus(0.12f, .14f, 10, 25); glTranslatef(0.0,0.0,-1.4); //translate to 3rd tyre glutSolidTorus(0.12f, .14f, 10, 25);

glTranslatef(-1.68,0.0,0.0); //translate to 4th tyre which is behind 1st tyre rearback

glutSolidTorus(0.12f, .14f, 10, 25); glTranslatef(.58,.52,0.7); //translate to origin glRotatef(90.0,0.0,1.0,0.0);

glTranslatef(0.0,0.0,-1.40); glutSolidTorus(0.2f, .2f, 10, 25); glTranslatef(0.0,0.0,1.40); glRotatef(270.0,0.0,1.0,0.0);

glBegin(GL\_POLYGON); //bottom filling

glColor3f(0.25,0.25,0.25);

glVertex3f(-0.3,-.48,0.7);//e

glVertex3f(-0.3,-.48,-0.7);//e'

glVertex3f(.81,-0.48,-0.7);//f'

glVertex3f(.81,-0.48,0.7);//f glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-.42,-.2,0.7);//d

glVertex3f(-.42,-.2,-0.7);//d'

glVertex3f(-0.3,-.48,-0.7);//e'

glVertex3f(-0.3,-.48,0.7);//e glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-1.2,-.28,0.7);//r

glVertex3f(-1.2,-.28,-0.7);//r'

glVertex3f(-1.12,-.48,-0.7);//a'

glVertex3f(-1.12,-.48,0.7);//a glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-1.12,-.48,0.7);//a

glVertex3f(-1.12,-.48,-0.7);//a'

glVertex3f(-0.86,-.48,-0.7);//b'

glVertex3f(-0.86,-.48,0.7);//b glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-0.86,-.48,0.7);//b

glVertex3f(-0.86,-.48,-0.7);//b'

glVertex3f(-.74,-0.2,-0.7);//c'

glVertex3f(-.74,-0.2,0.7);//c glEnd();

glBegin(GL\_POLYGON);

glVertex3f(-.74,-0.2,0.7);//c

glVertex3f(-.74,-0.2,-0.7);//c'

glVertex3f(-.42,-.2,-0.7);//d'

glVertex3f(-.42,-.2,0.7);//d glEnd();

glBegin(GL\_POLYGON); glVertex3f(.81,-0.48,0.7);//f

glVertex3f(.81,-0.48,-0.7);//f'

glVertex3f(.94,-0.2,-0.7);//g'

glVertex3f(.94,-0.2,0.7);//g glEnd();

glBegin(GL\_POLYGON); glVertex3f(.94,-0.2,0.7);//g

glVertex3f(.94,-0.2,-0.7);//g'

glVertex3f(1.24,-.2,-0.7);//h'

glVertex3f(1.24,-.2,0.7);//h

glEnd();

glBegin(GL\_POLYGON); glVertex3f(1.24,-.2,0.7);//h

glVertex3f(1.24,-.2,-0.7);//h'

glVertex3f(1.38,-.48,-0.7);//i'

glVertex3f(1.38,-.48,0.7);//i glEnd();

glBegin(GL\_POLYGON); glVertex3f(1.38,-.48,0.7);//i

glVertex3f(1.38,-.48,-0.7);//i'

glVertex3f(1.52,-.44,-0.7);//j'

glVertex3f(1.52,-.44,0.7);//j glEnd();

glBegin(GL\_LINE\_LOOP); // door outline- rear, front glColor3f(1.0,1.0,1.0);

glVertex3f(-0.5,0.22,0.7);//hh glVertex3f(0.0,0.22,0.7);//gg glVertex3f(0.0,-0.2,0.7);//oo

glVertex3f(-0.5,-0.2,0.7);//pp glEnd();

glBegin(GL\_LINE\_LOOP); // door2 outline- near, driver glVertex3f(0.12,0.22,0.7);//ll glVertex3f(0.62,0.22,0.7);//kk

glVertex3f(0.62,-0.2,0.7);//mm glVertex3f(0.12,-0.2,0.7);//nn

glEnd(); glColor3f(0.0,0.0,0.0);

glBegin(GL\_LINE\_LOOP); // door2 outline- far, driver glVertex3f(0.12,0.22,-0.7);//ll'

glVertex3f(0.62,0.22,-0.7);//kk'

glVertex3f(0.62,-0.2,-0.7);//mm'

glVertex3f(0.12,-0.2,-0.7);//nn' glEnd();

glBegin(GL\_LINE\_LOOP); // door outline- rear, far glVertex3f(-0.5,0.22,-0.7);//hh'

glVertex3f(0.0,0.22,-0.7);//gg'

glVertex3f(0.0,-0.2,-0.7);//oo'

glVertex3f(-0.5,-0.2,-0.7);//pp' glEnd();

glBegin(GL\_POLYGON); //front\*\* glVertex3f(1.52,.14,0.7);//k glVertex3f(1.52,.14,-0.7);//k'

glVertex3f(1.52,-.44,-0.7);//j'

glVertex3f(1.52,-.44,0.7);//j glEnd();

glColor3f(0.0,0.0,1.0);

// transparent objects are placed next .. glBlendFunc(GL\_SRC\_ALPHA, GL\_ONE\_MINUS\_SRC\_ALPHA); //TRANCPARENCY3

//windscreen glBegin(GL\_POLYGON);

glColor4f(0.0,0.0,0.0,0.7); //COLOR =WHITE TRANSPARENT

glVertex3f(0.562,.5,.6);//AAA glVertex3f(.562,.5,-.6);//AAA'

glVertex3f(.76,.22,-.6);//MMM' glVertex3f(.76,.22,.6);//MMM

glEnd();

glBegin(GL\_POLYGON); //rear window

//COLOR =WHITE TRANSPARENT

glVertex3f(-1.068,0.5,0.6);//pp glVertex3f(-1.068,0.5,-0.6);//pp'

glVertex3f(-1.2,0.22,-0.6);//qq' glVertex3f(-1.2,0.22,0.6);//qq

glEnd();

glBegin(GL\_POLYGON); //leftmost window front glVertex3f(-0.98,0.5,0.7);//aa glVertex3f(-0.64,0.5,0.7);//bb glVertex3f(-0.64,0.22,0.7);//cc glVertex3f(-1.08,0.22,0.7);//dd

glEnd();

glBegin(GL\_POLYGON); //leftmost window back glVertex3f(-0.98,0.5,-0.7);//aa

glVertex3f(-0.64,0.5,-0.7);//bb

glVertex3f(-0.64,0.22,-0.7);//cc

glVertex3f(-1.08,0.22,-0.7);//dd glEnd();

glBegin(GL\_POLYGON); //middle window front glVertex3f(-0.5,0.5,0.7);

glVertex3f(0.0,0.5,0.7);

glVertex3f(0.0,0.22,0.7);

glVertex3f(-0.5,0.22,0.7); glEnd();

glBegin(GL\_POLYGON); //middle window back glVertex3f(-0.5,0.5,-0.7);

glVertex3f(0.0,0.5,-0.7);

glVertex3f(0.0,0.22,-0.7);

glVertex3f(-0.5,0.22,-0.7);

glEnd();

glBegin(GL\_POLYGON); //rightmost window front glVertex3f(0.12,0.5,0.7);//ii glVertex3f(0.44,0.5,0.7);//jj glVertex3f(0.62,0.22,0.7);//kk glVertex3f(0.12,0.22,0.7);//ll

glEnd();

glBegin(GL\_POLYGON); //rightmost window back glVertex3f(0.12,0.5,-0.7);//ii'

glVertex3f(0.44,0.5,-0.7);//jj'

glVertex3f(0.62,0.22,-0.7);//kk'

glVertex3f(0.12,0.22,-0.7);//ll' glEnd();

glColor3f(0.0,0.0,1.0);

}

void drawhouse()

{

glBegin(GL\_LINE\_LOOP);

glVertex3f(-2.6,-.84,2.5);//m

glVertex3f(-2.6,0.84,2.5);//n glVertex3f(-3.04,0.84,2.8);//o glVertex3f(0,1.95,2.8);//p glVertex3f(3.04,0.84,2.8);//w glVertex3f(2.6,0.84,2.5);//q glVertex3f(2.6,-0.84,2.5);//r glVertex3f(1.59,-0.84,2.5);//s glVertex3f(1.59,0.16,2.5);//t glVertex3f(-1.59,0.16,2.5);//u glVertex3f(-1.59,-0.84,2.5);//v

glEnd(); glBegin(GL\_LINES);

glVertex3f(1.59,-0.84,2.5);//s glVertex3f(-1.59,-0.84,2.5);//v

glEnd(); glBegin(GL\_LINE\_LOOP);

glVertex3f(-2.6,-.84,-2.5);//m'

glVertex3f(-2.6,0.84,-2.5);//n'

glVertex3f(-3.04,0.84,-2.8);//o'

glVertex3f(0,1.95,-2.8);//p'

glVertex3f(3.04,0.84,-2.8);//w'

glVertex3f(2.6,0.84,-2.5);//q'

glVertex3f(2.6,-0.84,-2.5);//r'

glVertex3f(1.59,-0.84,-2.5);//s'

glVertex3f(1.59,0.16,-2.5);//t'

glVertex3f(-1.59,0.16,-2.5);//u'

glVertex3f(-1.59,-0.84,-2.5);//v' glEnd();

glBegin(GL\_LINES);

glVertex3f(-2.6,-.84,2.5);//m

glVertex3f(-2.6,-.84,-2.5);//m'

glVertex3f(-2.6,0.84,2.5);//n

glVertex3f(-2.6,0.84,-2.5);//n'

glVertex3f(-3.04,0.84,2.8);//o glVertex3f(-3.04,0.84,-2.8);//o' glVertex3f(0,1.95,2.8);//p glVertex3f(0,1.95,-2.8);//p' glVertex3f(3.04,0.84,2.8);//w glVertex3f(3.04,0.84,-2.8);//w' glVertex3f(2.6,0.84,2.5);//q glVertex3f(2.6,0.84,-2.5);//q'

glVertex3f(2.6,-0.84,2.5);//r

glVertex3f(2.6,-0.84,-2.5);//r' glVertex3f(1.59,-0.84,2.5);//s glVertex3f(1.59,-0.84,-2.5);//s'

glVertex3f(-1.59,-0.84,2.5);//v

glVertex3f(-1.59,-0.84,-2.5);//v' glEnd();

glColor3ub(255,185,1); //\*\*\*\*\*\*\*\*\*\*\*\*\* glBegin(GL\_QUADS);

glVertex3f(-2.6,-.84,2.5);//m glVertex3f(-2.6,0.16,2.5);//uu glVertex3f(-1.59,0.16,2.5);//u

glVertex3f(-1.59,-0.84,2.5);//v

glVertex3f(-2.6,0.16,2.5);//uu glVertex3f(-2.6,0.84,2.5);//n glVertex3f(2.6,0.84,2.5);//q glVertex3f(2.6,0.16,2.5);//tt glVertex3f(1.59,-0.84,2.5);//s glVertex3f(1.59,0.16,2.5);//t glVertex3f(2.6,0.16,2.5);//tt

glVertex3f(2.6,-0.84,2.5);//r

glVertex3f(-2.6,-.84,-2.5);//m'

glVertex3f(-2.6,0.16,-2.5);//uu'

glVertex3f(-1.59,0.16,-2.5);//u'

glVertex3f(-1.59,-0.84,-2.5);//v'

glVertex3f(-2.6,0.16,-2.5);//uu'

glVertex3f(-2.6,0.84,-2.5);//n'

glVertex3f(2.6,0.84,-2.5);//q'

glVertex3f(2.6,0.16,-2.5);//tt'

glVertex3f(1.59,-0.84,-2.5);//s'

glVertex3f(1.59,0.16,-2.5);//t'

glVertex3f(2.6,0.16,-2.5);//tt'

glVertex3f(2.6,-0.84,-2.5);//r'

glVertex3f(-2.6,-.84,2.5);//m

glVertex3f(-2.6,-.84,-2.5);//m'

glVertex3f(-2.6,0.84,-2.5);//n'

glVertex3f(-2.6,0.84,2.5);//n glVertex3f(2.6,0.84,2.5);//q

glVertex3f(2.6,0.84,-2.5);//q'

glVertex3f(2.6,-0.84,-2.5);//r'

glVertex3f(2.6,-0.84,2.5);//r glEnd();

glBegin(GL\_TRIANGLES); glVertex3f(0,1.95,2.5);//p glVertex3f(3.04,0.84,2.5);//w glVertex3f(-3.04,0.84,2.5);//o glVertex3f(0,1.95,-2.5);//p'

glVertex3f(3.04,0.84,-2.5);//w'

glVertex3f(-3.04,0.84,-2.5);//o' glEnd();

glColor3ub(255,102,0); //\*\*\*\*\*\*\*\*\*\*\*top color glBegin(GL\_QUADS);

glVertex3f(0,1.95,2.8);//p glVertex3f(0,1.95,-2.8);//p'

glVertex3f(3.04,0.84,-2.8);//w' glVertex3f(3.04,0.84,2.8);//w glVertex3f(-3.04,0.84,2.8);//o glVertex3f(-3.04,0.84,-2.8);//o'

glVertex3f(0,1.95,-2.8);//p' glVertex3f(0,1.95,2.8);//p

glEnd();

glColor3ub(116,18,0); //\*\*\*\*\*\*\*base color glBegin(GL\_QUADS);

glVertex3f(-2.6,-.84,2.5);//m

glVertex3f(2.6,-0.84,2.5);//r

glVertex3f(2.6,-0.84,-2.5);//r'

glVertex3f(-2.6,-.84,-2.5);//m' glEnd();

}

GLuint createDL()

{

GLuint carrDL;

carrDL = glGenLists(1); // Create the id for the list glNewList(carrDL,GL\_COMPILE); // start list

drawcarr(); // call the function that contains the rendering commands glEndList(); // endList

return(carrDL);

}

GLuint createDL2() //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

{

GLuint houseDL;

houseDL = glGenLists(1); // Create the id for the list glNewList(houseDL,GL\_COMPILE); // start list

drawhouse(); // call the function that contains the rendering commands glEndList(); // endList

return(houseDL);

} //\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void initScene()

{

glEnable(GL\_DEPTH\_TEST); carr\_display\_list = createDL();

house\_display\_list= createDL2(); //\*\*\*\*\*\*\*\*\*\*\*

}

void renderScene(void)

{

int i,j;

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

glClearColor(.7,0.85,1.0,1.0); glColor3f(0.25f, 0.25f, 0.25f); // Draw ground

glBegin(GL\_QUADS);

glVertex3f(-100.0f, 0.0f, -100.0f); glVertex3f(-100.0f, 0.0f, 100.0f); glVertex3f( 100.0f, 0.0f, 100.0f); glVertex3f( 100.0f, 0.0f, -100.0f);

glEnd();

for( i = -3; i < 3; i++) // Draw 36 car for( j=-3; j < 3; j++)

{

glPushMatrix(); glTranslatef((i)\*10.0,0,(j) \* 10.0);

glColor3ub(a[i],b[j],c[i]); glCallList(carr\_display\_list); glPopMatrix();

}

if(housevisible)

{

glPushMatrix(); glScalef(2.0,2.0,2.0);

glTranslatef(0.0,.85,-20.0); glCallList(house\_display\_list); glTranslatef(10.0,0.0,0.0); glCallList(house\_display\_list); glTranslatef(-20.0,0.0,0.0); glCallList(house\_display\_list); glRotatef(90,0.0,1.0,0.0);

glTranslatef(-10.0,0.0,-10.0); glCallList(house\_display\_list); glTranslatef(-10.0,0.0,0.0); glCallList(house\_display\_list); glTranslatef(-10.0,0.0,0.0); glCallList(house\_display\_list); glPopMatrix();

glPushMatrix(); glTranslatef(10.0,3.4,-80.0);

glScalef(4.0,4.0,4.0); glCallList(house\_display\_list); glTranslatef(-10.0,0.0,0.0); glCallList(house\_display\_list); glPopMatrix();

glPushMatrix(); glRotatef(90,0.0,1.0,0.0);

glScalef(2.0,2.0,2.0); glTranslatef(0.0,0.85,15.0); glCallList(house\_display\_list); glTranslatef(10.0,0.,0.0); glCallList(house\_display\_list); glTranslatef(-20.0,0.,0.0); glCallList(house\_display\_list); glPopMatrix();

}

if(fxincr!=0) theta1=(atan(fzincr/fxincr)\*180)/3.141; else if(fzincr>0)

theta1=-90.0; else theta1=90.0; if(fxincr>0&&fzincr<0)

{

theta1=-theta1;

}

else if(fxincr<0&&fzincr<0)

{

theta1=180-theta1;

}

else if(fxincr<0&&fzincr>0)

{

theta1=-180-theta1;

}else if(fxincr>0&&fzincr>0)

{

theta1=-theta1;

}

glPushMatrix(); glTranslatef(fx,0,fz); glRotatef(theta1,0,1,0); glColor3f(0.8,0.8,0); glCallList(carr\_display\_list); glPopMatrix(); glutSwapBuffers();

}

void orientMe(float ang)

{

lx = sin(ang); lz = -cos(ang);

glLoadIdentity();

gluLookAt(x, y, z, x + lx,y + ly,z + lz,0.0f,1.0f,0.0f);

}

void moveMeFlat(int i)

{

if(xxxx==1)

y=y+i\*(lz)\*0.1; //\*\*\*\*\*\*\*\*\* if(yyyy==1)

{

}

else

{

x=x+i\*(lz)\*.1;

z = z + i\*(lz)\*0.5; x = x + i\*(lx)\*0.5;}

glLoadIdentity();

gluLookAt(x, y, z,x + lx,y + ly,z + lz,0.0f,1.0f,0.0f);

}

void processNormalKeys(unsigned char key, int x, int y)

{

glLoadIdentity(); if (key == 'q')

exit(0);

if(key=='t')

gluLookAt(1,190,50,0,0 ,-10,0.0,1.0,.0);

if(key=='a')

if(key=='s')

if(key=='w')

if(key=='d')

}

moveMeFlat(4);xxxx=1,yyyy=0; moveMeFlat(-4);xxxx=1,yyyy=0; moveMeFlat(4);yyyy=1;xxxx=0; moveMeFlat(-4);yyyy=1;xxxx=0;

void inputKey(int key, int x, int y)

{

switch (key)

{

case GLUT\_KEY\_LEFT : angle -= 0.05f;orientMe(angle);break; case GLUT\_KEY\_RIGHT : angle +=0.05f;orientMe(angle);break; case GLUT\_KEY\_UP : moveMeFlat(2);xxxx=0,yyyy=0;break;

case GLUT\_KEY\_DOWN : moveMeFlat(-2);xxxx=0,yyyy=0;break;

}

}

void movecar(int key, int x, int y)

{

switch (key)

{

theta);

case GLUT\_KEY\_LEFT :temp=fxincr;

fxincr=fxincr\*cos(theta)+fzincr\*sin(theta); fzincr=-temp\*sin(theta)+fzincr\*cos(theta); fx+=fxincr;

fz+=fzincr; break;

case GLUT\_KEY\_RIGHT :temp=fxincr;

fxincr=fxincr\*cos(-theta)+fzincr\*sin(-

fzincr=-temp\*sin(-theta)+fzincr\*cos(-theta); fx+=fxincr;

fz+=fzincr; break;

case GLUT\_KEY\_UP :fx+=fxincr;

fz+=fzincr;break; case GLUT\_KEY\_DOWN :fx-=fxincr;

fz-=fzincr; break;

}

glutPostRedisplay();

}

void ProcessMenu(int value) // Reset flags as appropriate in response to menu selections

{

glutPostRedisplay();

}

void ProcessMenu1(int value)

{

switch(value)

{

case 1:if(housevisible==0)

housevisible=1;

else

housevisible=0;

glutPostRedisplay(); break;

case 2:if(movecarvar==0)

{

glutSpecialFunc(movecar);

movecarvar=1;

}

else

{

glutSpecialFunc(inputKey); movecarvar=0;

}

break;

}

}

void menu()

{

int control; int control1;

control= glutCreateMenu(ProcessMenu); glutAddMenuEntry("\*\*CONTROLS\*\*",1);

glutAddMenuEntry("1) UP KEY:to move in Forward Direction.",1); glutAddMenuEntry("2) DOWN KEY:to move in Backward Direction.",1);

glutAddMenuEntry("3) LEFT KEY:to Turn Left .",1); glutAddMenuEntry("4) RIGHT KEY:to Turn Right .",1); glutAddMenuEntry("5) d:moves Towards Right. ",1);

glutAddMenuEntry("6) a:moves Towards Left.",1); glutAddMenuEntry("7) s:moves Away.",1);

glutAddMenuEntry("8) w:moves Near.",1); glutAddMenuEntry("9) t:Top view.",1); glutAddMenuEntry("10) q:Quit.",1); glutAttachMenu(GLUT\_RIGHT\_BUTTON); control1=glutCreateMenu(ProcessMenu1); glutAddMenuEntry("HOUSE",1);

glutAddMenuEntry("MOVE CAR",2); glutAttachMenu(GLUT\_LEFT\_BUTTON);

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DEPTH | GLUT\_DOUBLE | GLUT\_RGBA); glutInitWindowPosition(0,0); glutInitWindowSize(1010,710);

glutCreateWindow("car lot"); initScene(); glutKeyboardFunc(processNormalKeys); glutSpecialFunc(inputKey);

menu(); glutDisplayFunc(renderScene); glutIdleFunc(renderScene); glutReshapeFunc(changeSize); glutMainLoop();

return(0);

}