Nom i Cognoms: _____Grup: ____DNI: _____

1 (3 punts) Indica els paràmetres d'una càmera ortogonal (posició + orientació amb transformacions geomètriques, i òptica) i codi OpenGL que permeti veure només el primer dels dos avions en una vista que permeti veure tant el seu cilindre central com les seves "ales". Dibuixa la imatge resultant i justifica l'elecció de tots els paràmetres. El *viewport* és quadrat.

Nom i Cognoms: _____Grup: ____DNI: _____

2 (3 punts) El model simplificat d'un avió és crea en base a 3 cilindres:

- un cilindre de radi 30, llargada 100, eix central orientat segons eix X+ i base centrada en origen de coordenades.
- un cilindre de radi 3, llargada 50, eix central orientat segons eix Z+ i base centrada en el punt (75, 0, 10).
- un cilindre de radi 3, llargada 50, eix central orientat segons eix Z- i base centrada en el punt (75, 0, -10).

Volem crear una escena en què hi ha dos avions que es pintaran ubicant el model de l'avió en dos llocs diferents de l'espai: un avió tindrà el centre de la seva capsa mínima contenidora en el (0,0,0) i estarà orientat cap l'eix X+; l'altre avió tindrà en centre de la seva capsa mínima contenidora en (0,0,120) i orientat cap l'eix Z^+.

- Indica el codi OpenGL requerit per a pintar el model de l'avió utilitzant el mètode: `pinta_cilindre(R,H)` que envia a pintar un cilindre de radi R, alçada H, amb la seva base al (0,0,0) i orientat segons l'eix Z+. Indica també l'expressió matemàtica de les transformacions que fas a cada cilindre per a configurar l'avió.

- Indica el codi OpenGL requerit per a pintar l'escena utilitzant el mètode proposat en l'apartat anterior.

Nom i Cognoms: _____Grup: ____DNI: _____

3 (1 punt) Disposem d'una càmera axonomètrica amb els següents paràmetres: OBS=(0.,0.,0.), VRP=(-1.,0.,0.), up=(0.,1.,0.), *window* de (-5,-5) a (5,5), zn=5, zf=10.
Indiqueu quin altre conjunt de paràmetres de càmera defineix exactament el mateix volum de visió (és a dir, garanteix generar exactament la mateixa imatge de l'escena):

a. OBS= (1,0,0), VRP= (0,0,0), up=(0,2,0), zn= 6, zf=11
b. OBS= (0,1,0), VRP=(0,0,0), up= (0,1,0), zn=5, zf=10
c. OBS= (0,0,0), VRP=(-2,0,0), up=(0,1,0), zn=6, zf=11
d. OBS= (-1,0,0), VRP=(0,0,0), up=(0,1,0), zn=-1, zf=9

4 (1 Punt) Tenim el següent codi que pinta una escena:

```
glViewport (0,0,600,600);
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
gluPerspective(60., 1.0, 1.0, 100.0);
glMatrixMode(GL_MODELVIEW);
glLoadIdentity();
gluLookAt(0., 0.,10,0,0,0,0,1,0);
glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
glEnable (GL_DEPTH_TEST);
glColor3f(1,0,0);
glScaled(1.0, 5.0, 1.0);
glRotatef(-90, 1.0, 0.0, 0.0);
glRotatef(-45, 0.0, 1.0, 0.0);
glScaled(1.0, 5.0, 1.0);
glutSolidCube(1.0);
glRotatef(30., 0, 0, 1.);
```

Digues què es veu:

a. Un triangle vermell amb la seva base horitzontal al centre de la pantalla i la punta superior cap amunt.
b. Cap de les altres
c. Un rombe més ample que alt amb el seu centre situat al centre de la pantalla.
d. Un rombe més alt que ample amb el seu centre situat al centre de la pantalla.

5 (1 Punt) Quan es realitza la crida a glVertex3f(x,y,z), OpenGL realitza una sèrie de transformacions per a obtenir el píxel en què cal pintar-lo. Quina d'aquestes seqüències es correspon amb les transformacions que es fan?

a. Transformació de *modelview*, transformació de projecció, transformació *window-viewport*.
b. Transformació de projecció, transformació *window-viewport*, transformació de *modelview*.
c. Transformació de projecció, transformació de *modelview*, transformació de *window-viewport*.
d. Transformació de *window-viewport*, transformació de projecció, transformació de *modelview*.

6 (1 punt) En les inicialitzacions prèvies al pintat d'una escena, tenim la següent seqüència d'instruccions OpenGL que defineix una càmera amb un *window* quadrat i un *viewport* també quadrat:

```
gluPerspective(myFovy, 1.0, myNear, myFar);
glViewport (0, 0, 400, 400);
```

quina diferència s'observaria en la visualització de l'escena si les canviem per:

```
gluPerspective (myFovy, 2.0, myNear, myFar);
glViewport (0, 0, 400, 400);
```

a. Cap perquè la relació d'aspecte de la càmera és superior a la del *viewport*, per això no cal modificar res més.
b. L'escena es veurà deformada amb el doble de llargada que amplada.
c. L'escena es veurà deformada amb el doble d'amplada que llargada.

d. Com no hem modificat FOV, es veurà retallada l'escena respecte l'inicial.