

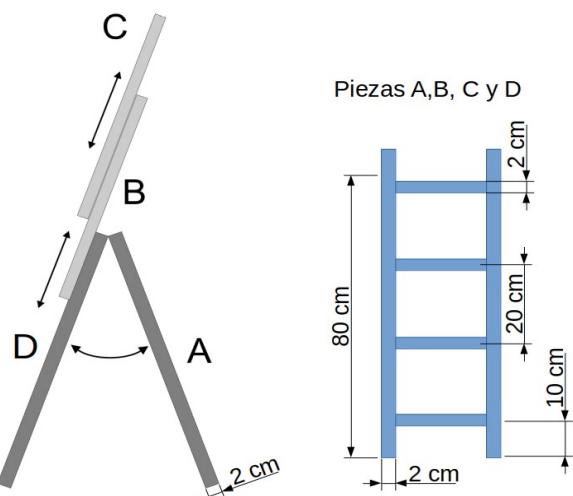
Apellidos: _____ Nombre: _____

- 1. (1 pto)** La imagen siguiente corresponde a la visualización de un poliedro regular de radio 1, centrado en el origen, que aproxima una esfera. Indica:

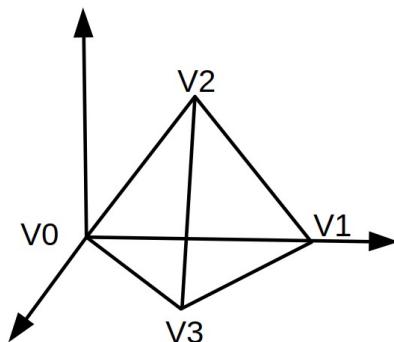
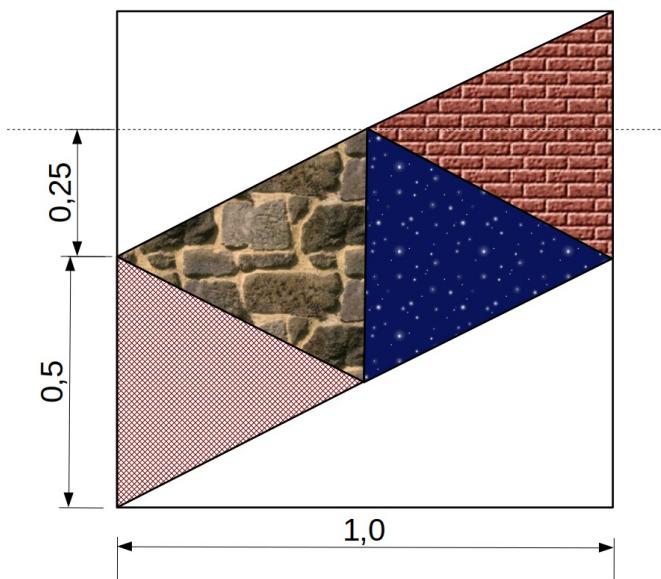
- a) Que fuentes de luz y que propiedades de material se han usado para generarla.
- b) Que cambios se deben hacer en el código para que la visualización simule una esfera, indicando como se calcularían las normales.



- 2. (3 ptos)** Diseñar el grafo de escena para la escalera de la figura. Se creará a partir de un cubo de lado unidad alineado con los ejes de coordenadas con el centro de la cara inferior en el origen de coordenadas. Las cuatro piezas (A,B, C y D) son iguales, su estructura se muestra en la figura de la derecha. En el modelo, las piezas D y A giran respecto a la bisagra de unión. La pieza B se despalza sobre D y C sobre B.



- 3. (2 puntos)** Escribe el código en C++ con OpenGL que dibuje un tetraedro (no necesariamente regular) aplicándole la textura que se muestra en la figura de la izquierda. Anotar las coordenadas usadas para los vértices del tetraedro en la figura de la derecha.



- 4. (2 puntos)** Se dispone de la siguiente estructura de datos para representar una malla de triángulos. Escribe un método para seleccionar un triángulo de la malla realizando un click con el botón izquierdo del ratón.

```
typedef struct{
    GLfloat x;
    GLfloat y;
    GLfloat z;
} punto;

typedef struct{
    int v0;
    int v1;
    int v2;
} cara;

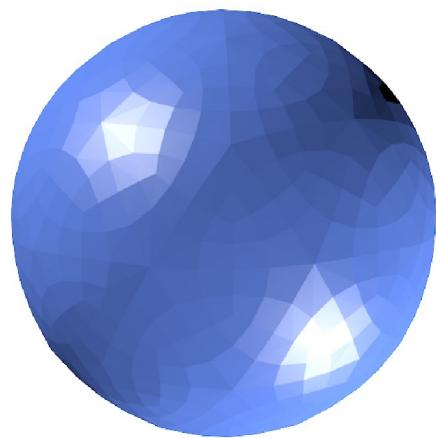
class Malla : public Objeto3D
{
public:
    std::vector<punto> vertices;
    std::vector<cara> caras;
    std::vector<punto> nv;
    std::vector<punto> nc;
    ....
    void draw(){
        float color[4] = { 0.5, 1.0, 0.5, 1 };
        glShadeModel(GL_FLAT);
        glMaterialfv (GL_FRONT, GL_AMBIENT_AND_DIFFUSE, color);
        glBegin(GL_TRIANGLES);
        for(int i =0; i<caras.size();i++){
            glNormal3f(nc[i].x,nc[i].y,nc[i].z);
            glVertex3f(vertices[caras[i].v0].x,vertices[caras[i].v0].y,vertices[caras[i].v0].z);
            glVertex3f(vertices[caras[i].v1].x,vertices[caras[i].v1].y,vertices[caras[i].v1].z);
            glVertex3f(vertices[caras[i].v2].x,vertices[caras[i].v2].y,vertices[caras[i].v2].z);
        }
        glEnd();
    }
};
```

- 5. (2 ptos)** Sobre la representación anterior describe cómo se puede determinar si la orientación de las normales de las caras es consistente (están orientadas hacia el exterior del objeto).

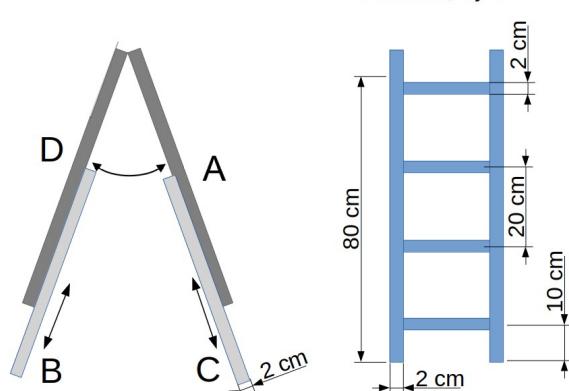
Apellidos: _____ Nombre: _____

- 1. (1 pto)** La imagen siguiente corresponde a la visualización de un poliedro regular de radio 1, centrado en el origen, que aproxima una esfera. Indica:

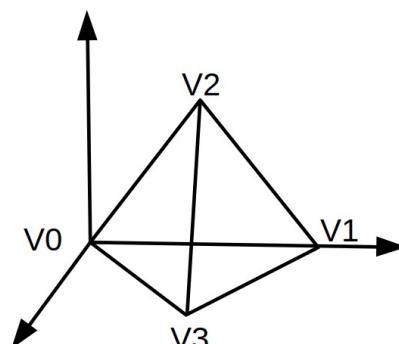
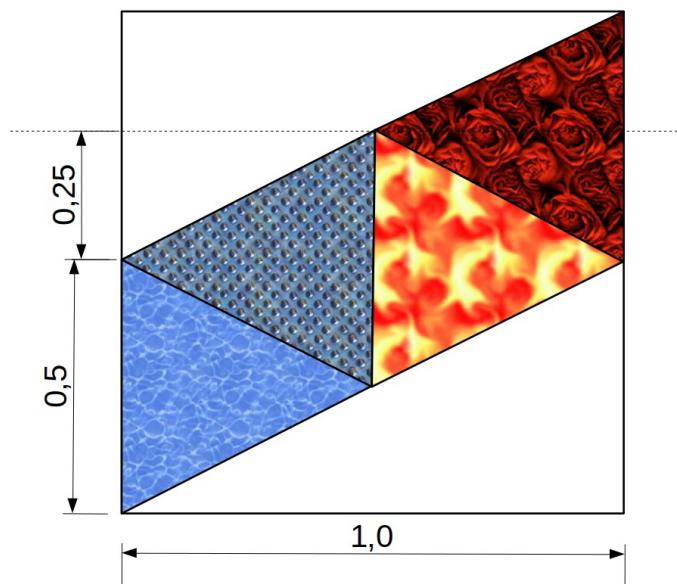
- a) Que fuentes de luz y que propiedades de material se han usado para generarla.
- b) Que cambios se deben hacer en el código para que la visualización simule una esfera, indicando como se calcularían las normales.



- 2. (3 ptos)** Diseñar el grafo de escena para la escalera de la figura. Se creará a partir de un cubo de lado unidad alineado con los ejes de coordenadas con el centro de la cara inferior en el origen de coordenadas. Las cuatro piezas (A,B, C y D) son iguales, su estructura se muestra en la figura de la derecha. En el modelo, las piezas B y C se desplazan sobre D y A. Estas últimas giran respecto a la bisagra de unión.



- 3. (2 puntos)** Escribe el código en C++ con OpenGL que dibuje un tetraedro (no necesariamente regular) aplicándole la textura que se muestra en la figura de la izquierda. Anotar las coordenadas usadas para los vértices del tetraedro en la figura de la derecha.



- 4. (2 puntos)** Se dispone de la siguiente estructura de datos para representar una malla de triángulos. Escribe un método para seleccionar un triángulo de la malla realizando un click con el botón central del ratón.

```
typedef struct{
    GLfloat x;
    GLfloat y;
    GLfloat z;
} punto;

typedef struct{
    int v0;
    int v1;
    int v2;
} cara;

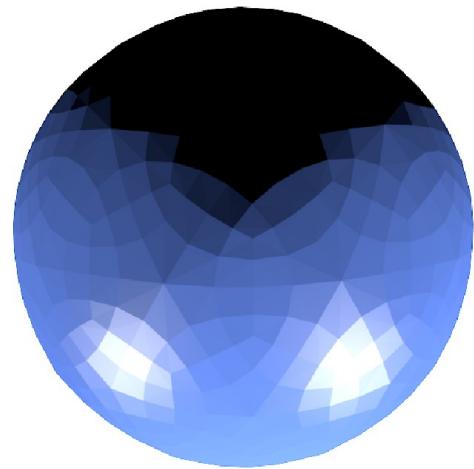
class Malla : public Objeto3D
{
public:
    std::vector<punto> vertices;
    std::vector<cara> caras;
    std::vector<punto> nv;
    std::vector<punto> nc;
    ....
    void draw(){
        float color[4] = { 0.5, 1.0, 0.5, 1 };
        glShadeModel(GL_FLAT);
        glMaterialfv (GL_FRONT, GL_AMBIENT_AND_DIFFUSE, color);
        glBegin(GL_TRIANGLES);
        for(int i =0; i<caras.size();i++){
            glNormal3f(nc[i].x,nc[i].y,nc[i].z);
            glVertex3f(vertices[caras[i].v0].x,vertices[caras[i].v0].y,vertices[caras[i].v0].z);
            glVertex3f(vertices[caras[i].v1].x,vertices[caras[i].v1].y,vertices[caras[i].v1].z);
            glVertex3f(vertices[caras[i].v2].x,vertices[caras[i].v2].y,vertices[caras[i].v2].z);
        }
        glEnd();
    }
};
```

- 5. (2 ptos)** Sobre la representación anterior describe cómo se puede determinar si la orientación de las normales de las caras es consistente (están orientadas hacia el exterior del objeto).

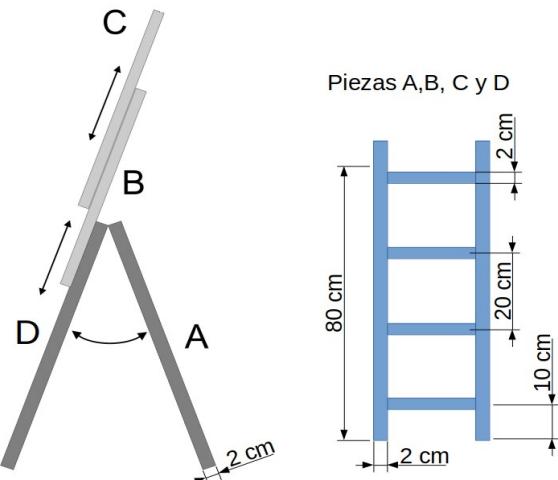
Apellidos: _____ Nombre: _____

- 1. (1 pto)** La imagen siguiente corresponde a la visualización de un poliedro regular de radio 1, centrado en el origen, que aproxima una esfera. Indica:

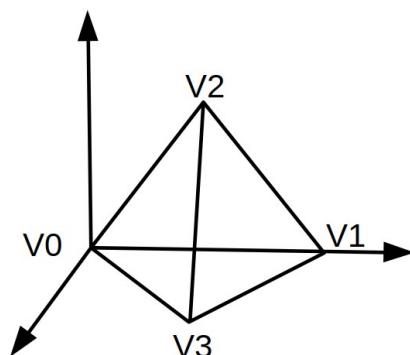
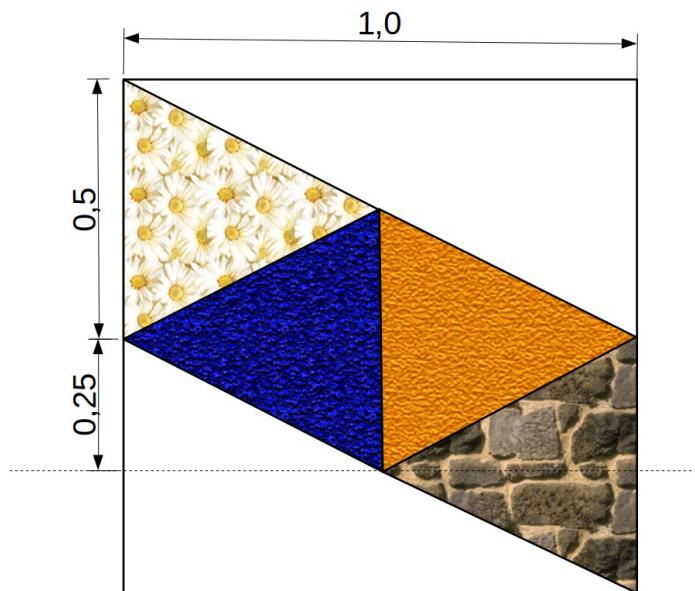
- a) Que fuentes de luz y que propiedades de material se han usado para generarla.
- b) Que cambios se deben hacer en el código para que la visualización simule una esfera, indicando como se calcularían las normales.



- 2. (3 ptos)** Diseñar el grafo de escena para la escalera de la figura. Se creará a partir de un cubo de lado unidad alineado con los ejes de coordenadas con el centro de la cara inferior en el origen de coordenadas. Las cuatro piezas (A,B, C y D) son iguales, su estructura se muestra en la figura de la derecha. En el modelo, las piezas D y A giran respecto a la bisagra de unión. La pieza B se despalza sobre D y C sobre B.



- 3. (2 puntos)** Escribe el código en C++ con OpenGL que dibuje un tetraedro (no necesariamente regular) aplicándole la textura que se muestra en la figura de la izquierda. Anotar las coordenadas usadas para los vértices del tetraedro en la figura de la derecha.



- 4. (2 puntos)** Se dispone de la siguiente estructura de datos para representar una malla de triángulos. Escribe un método para seleccionar un triángulo de la malla realizando un click con el botón derecho del ratón.

```

typedef struct{
    GLfloat x;
    GLfloat y;
    GLfloat z;
} punto;

typedef struct{
    int v0;
    int v1;
    int v2;
} cara;

class Malla : public Objeto3D
{
public:
    std::vector<punto> vertices;
    std::vector<cara> caras;
    std::vector<punto> nv;
    std::vector<punto> nc;
    ....
    void draw(){
        float color[4] = { 0.5, 1.0, 0.5, 1 };
        glShadeModel(GL_FLAT);
        glMaterialfv (GL_FRONT, GL_AMBIENT_AND_DIFFUSE, color);
        glBegin(GL_TRIANGLES);
        for(int i =0; i<caras.size();i++){
            glNormal3f(nc[i].x,nc[i].y,nc[i].z);
            glVertex3f(vertices[caras[i].v0].x,vertices[caras[i].v0].y,vertices[caras[i].v0].z);
            glVertex3f(vertices[caras[i].v1].x,vertices[caras[i].v1].y,vertices[caras[i].v1].z);
            glVertex3f(vertices[caras[i].v2].x,vertices[caras[i].v2].y,vertices[caras[i].v2].z);
        }
        glEnd();
    }
};

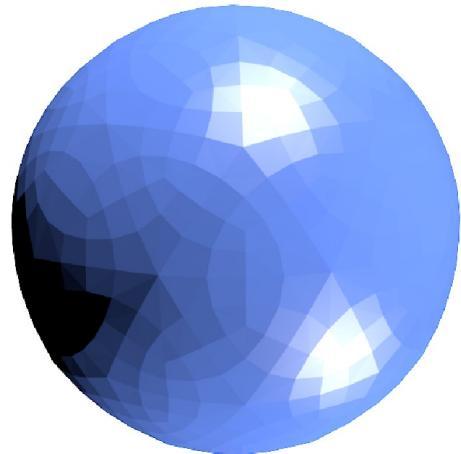
```

- 5. (2 ptos)** Sobre la representación anterior describe cómo se puede determinar si la orientación de las normales de las caras es consistente (están orientadas hacia el exterior del objeto).

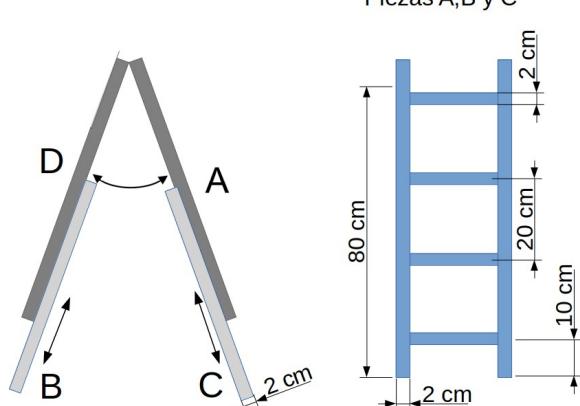
Apellidos: _____ Nombre: _____

- 1. (1 pto)** La imagen siguiente corresponde a la visualización de un poliedro regular de radio 1, centrado en el origen, que aproxima una esfera. Indica:

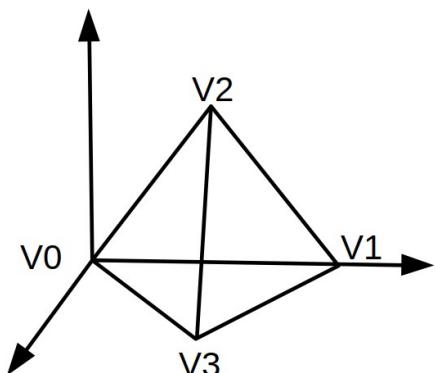
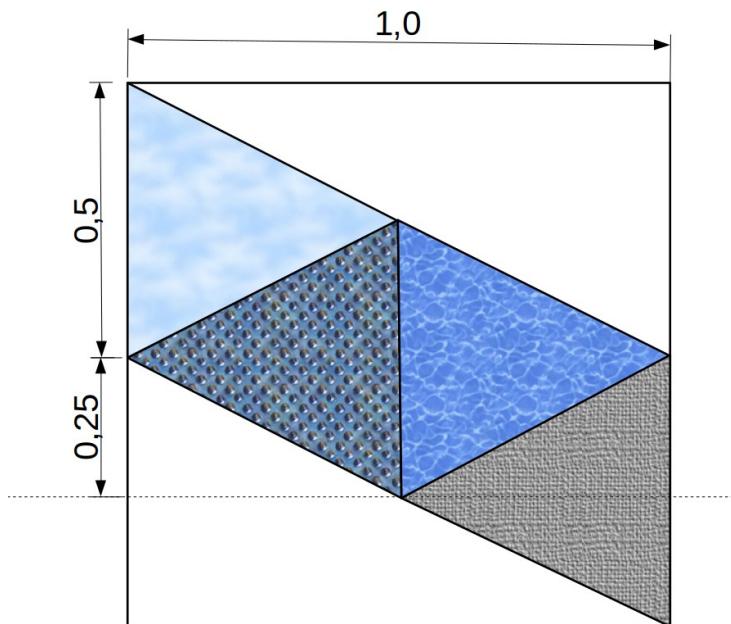
- a) Que fuentes de luz y que propiedades de material se han usado para generarla.
- b) Que cambios se deben hacer en el código para que la visualización simule una esfera, indicando como se calcularían las normales.



- 2. (3 ptos)** Diseñar el grafo de escena para la escalera de la figura. Se creará a partir de un cubo de lado unidad alineado con los ejes de coordenadas con el centro de la cara inferior en el origen de coordenadas. Las cuatro piezas (A,B, C y D) son iguales, su estructura se muestra en la figura de la derecha. En el modelo, las piezas B y C se desplazan sobre D y A. Estas últimas giran respecto a la bisagra de unión.



- 3. (2 puntos)** Escribe el código en C++ usando OpenGL que dibuje un tetraedro (no necesariamente regular) aplicándole la textura que se muestra en la figura de la izquierda. Anotar las coordenadas usadas para los vértices del tetraedro en la figura de la derecha.



- 4. (2 puntos)** Se dispone de la siguiente estructura de datos para representar una malla de triángulos.
Escribe un método para seleccionar un triángulo de la malla realizando un click con el botón izquierdo del ratón.

```
typedef struct{
    GLfloat x;
    GLfloat y;
    GLfloat z;
} punto;

typedef struct{
    int v0;
    int v1;
    int v2;
} cara;

class Malla : public Objeto3D
{
public:
    std::vector<punto> vertices;
    std::vector<cara> caras;
    std::vector<punto> nv;
    std::vector<punto> nc;
    ....
void draw(){
    float color[4] = { 0.5, 1.0, 0.5, 1 };
    glShadeModel(GL_FLAT);
    glMaterialfv (GL_FRONT, GL_AMBIENT_AND_DIFFUSE, color);
    glBegin(GL_TRIANGLES);
    for(int i =0; i<caras.size();i++){
        glNormal3f(nc[i].x,nc[i].y,nc[i].z);
        glVertex3f(vertices[caras[i].v0].x,vertices[caras[i].v0].y,vertices[caras[i].v0].z);
        glVertex3f(vertices[caras[i].v1].x,vertices[caras[i].v1].y,vertices[caras[i].v1].z);
        glVertex3f(vertices[caras[i].v2].x,vertices[caras[i].v2].y,vertices[caras[i].v2].z);
    }
    glEnd();
}
};
```

- 5. (2 ptos)** Sobre la representación anterior describe cómo se puede determinar si la orientación de las normales de las caras es consistente (están orientadas hacia el exterior del objeto).