1. Se pueden usar .position, .rotate y .localscale de la componente transform del objeto. También existen otras funciones como: .translate(), localposition, .rotation... Que permiten realizar también movimientos y que también son necesarias para los movimientos.
2. Mediante la localscale de la siguiente forma:

```
transform.localScale = new Vector3(transform.localScale.x, transform.localScale.y, transform.localScale.y) * 2;
```

3. La función .position nos lo permite con un Vector3:

```
transform.position = new Vector3(3, 5, 1);
```

4. Primeros aumentamos la posición en 3 unidades en todos los ejes y luego lo rotamos sobre el eje como se nos pide. ```cs Una forma de realizar esta rotación es mediante rotate transform.position = new Vector3(transform.position.x + 3, transform.position.y + 3, transform.position.z + 3); transform.rotation(0, 30, 0);

```
5. Una forma es mediante los cuaterniones
```cs
transform.rotation = Quaternion.Euler(1, 1, 1);
```

6. Si. La Rotación del objeto no influye luego en las traslaciones.

```
transform.rotation = Quaternion.Euler(tranform.rotation.x, transform.rotation.y + 30, transform.rotation.z);
transform.position = new Vector3(transform.position.x + 3, transform.position.y + 3, transform.position.z + 3);
```

7. Se puede acceder al atributo dentro de la componente camera del objeto o mediante el inspector en el campo near.

```
//declaramos la camara (publica para acceder desde el inspector)
public Camera cam;

//en el cuerpo de alguno de los métodos de la clase
float nearPlane = cam.NearClipPlane;
```

8. Es una propiedad de la componente camera y se puede acceder de la misma forma que los planos de la camara o una mejor manera es directamente del inspector de la cámara.

```
//declaramos la camara (publica para acceder desde el inspector)
public Camera cam;

//en el cuerpo de alguno de los métodos de la clase
float fieldOfView = cam.fieldOfView;
```

9. Disminuir el ángulo tiene el efecto de zoom reduciendo la cantidad de espacio visible.
10. Se puede o bien modificar los planos cercano y lejano de la cámara o bien reducir el campo field of view de la cámara y cualquiera de estos cambios resultaría en reducir el número de objetos visibles en la escena.
11. Con las matriz de proyección.
12. Se emplea .rotation de la componente transform y un cuaternión especificandole ángulos de Euler.

```
transform.rotation = Quaternion.Euler(tranform.rotation.x, transform.rotation.y + 30, transform.rotation.z);
```

13. Es una propiedad de la componente de la cámara a la cual se puede acceder de la siguente manera:

```
//declaramos la camara (publica para acceder desde el inspector)
public Camera cam;

//en el cuerpo de alguno de los métodos de la clase
Matrix4x4 previousViewProyectionMatrix = cam.previousViewProyectionMatrix;
```

14. Primero habría que modificar la una de las propiedades de la cámara para que funcione en perspectiva ortogonal y luego obtenemos la matriz:
```

```
 //declaramos la camara (publica para acceder desde el inspector)
public Camera cam;

//en el cuerpo de alguno de los métodos de la clase
cam.orthographic = true;
Matrix4x4 previousViewProyectionMatrix = cam.previousViewProyectionMatrix;
```

15. Se puede obtener mediante .worldToCamera de la componente transform (transform.worldToCamera).
16. Mediante uno de los atributos de la cámara .worldToCameraMatrix

```
 //declaramos la camara (publica para acceder desde el inspector)
public Camera cam;

//en el cuerpo de alguno de los métodos de la clase
Matrix4x4 worldToCameraMatrix = cam.worldToCameraMatrix;
```

17. Matriz de proyección de la práctica 1

```
-1.66162   0.00000   0.00000  -49.84860
 0.00000  -3.73205   0.00000    6.34449
 0.00000   0.00000   0.00005    0.01894
 0.00000   0.00000  -1.00000   21.32999
```

18. Matriz de modelo

    1.00000 0.00000 0.00000 33.00000 0.00000 1.00000 0.00000 0.10000 0.00000 0.00000 3.00000 8.00000 0.00000 0.00000 0.00000 1.00000

19. Matriz de modelo despues de aplicar una rotacion

    0.86603 0.00000 1.50000 33.00000 0.00000 1.00000 0.00000 0.10000 -0.50000 0.00000 2.59808 -8.00000 0.00000 0.00000 0.00000 1.00000

20. Mediante la propiedad de la componente transform localToWorldMatrix.

Aram Pérez Dios