|  |  |  |
| --- | --- | --- |
| Texto  Descripción generada automáticamente con confianza media | TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS  FACULTAD DE INGENIERÍA  Universidad Nacional de Jujuy |  |

Trabajo Práctico N.º 2

Caro Victor Facundo – TUV000741

*Profesores:*

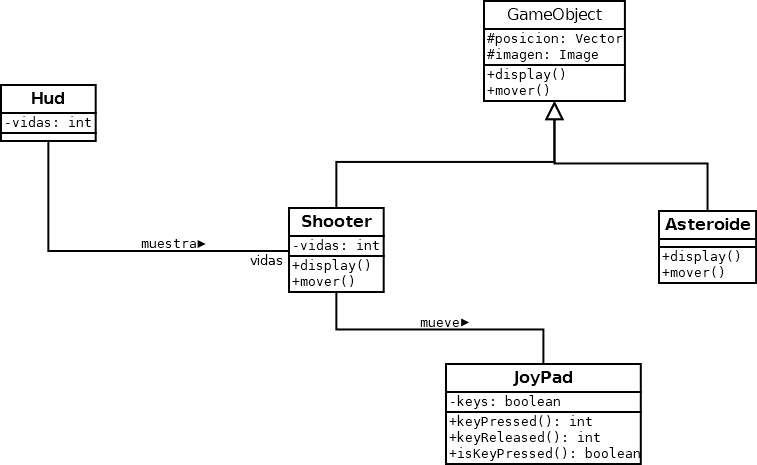
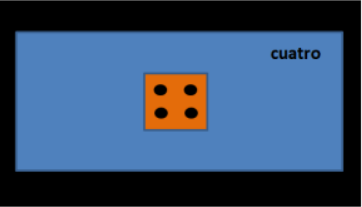
*Mg. Ing. Ariel Alejandro Vega*

**FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS**

**Punto 1**: Desarrolle una historia de usuario, en la cual defina la visualización y movimiento de una clase GameObject, de la que heredan Shooter y Asteroide. GameObjects es abstracta, y posee atributos protegidos: posición, imagen; además del método abstracto display() y mover(). Además debe poseer un HUD que visualice la cantidad de vidas del Shooter. Utilce un JoyPad para generar los movimientos.

***Titulo*: Asteroides**

***Descripción*:** Como piloto espacial, debo enfrentarme a asteroides y sobrevivir el mayor tiempo posible. También necesito tener una forma de visualizar mis vidas restantes durante la batalla. Los asteroides se mueven automáticamente por la pantalla en patrones de ataque predefinidos. El jugador puede controlar su nave espacial usando un JoyPad para esquivar disparos y contraatacar. Un HUD muestra la cantidad de vidas restantes del jugador en la esquina superior derecha de la pantalla.

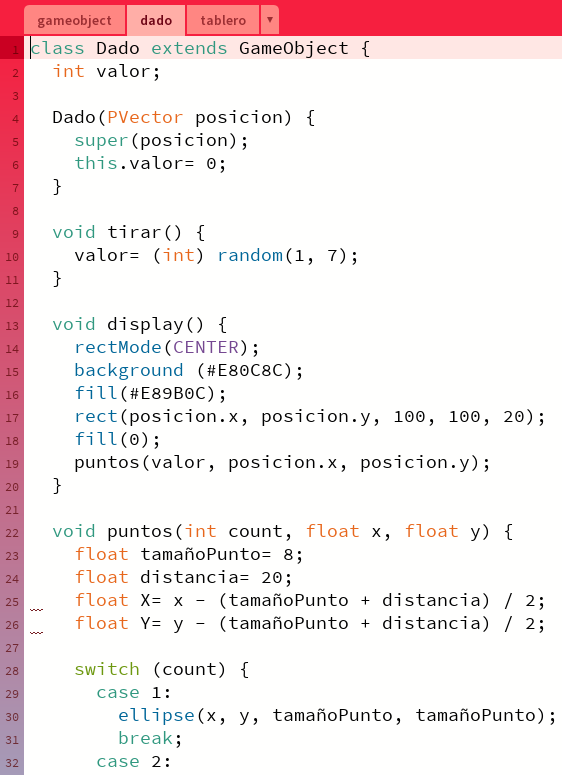
**Punto 2**: Desarrolle un videojuego que cumpla con las siguientes especificaciones: Realice un diagrama de clases

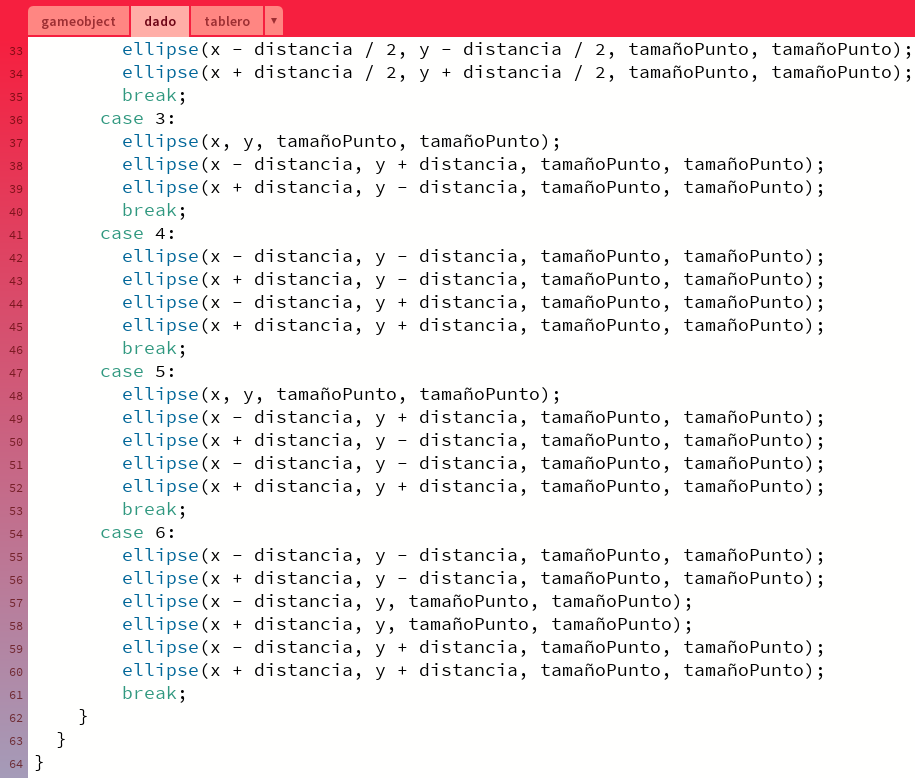
Como se observa se trata de un dado. El cual al presionar un botón debe generar un número aleatorio entre 1 y 6 y dibujarlo. Además, debe mostrar el número en la parte superior derecha. Repetir esto cuantas veces lo desee y al finalizar (con otro botón) debe dibujar por consola y agrupado en filas de 4 columnas los dados obtenidos.

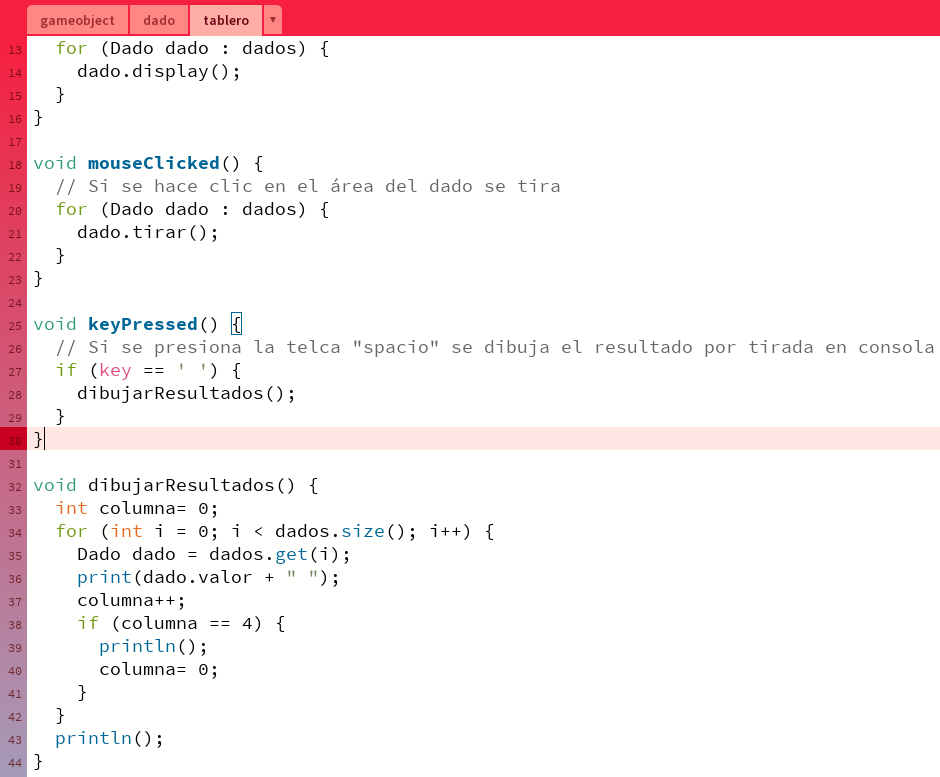
Al momento de programar utilice constructores sobrecargados. Considere que el dado se muestra en un tablero, este tablero contiene al dado, y al texto.

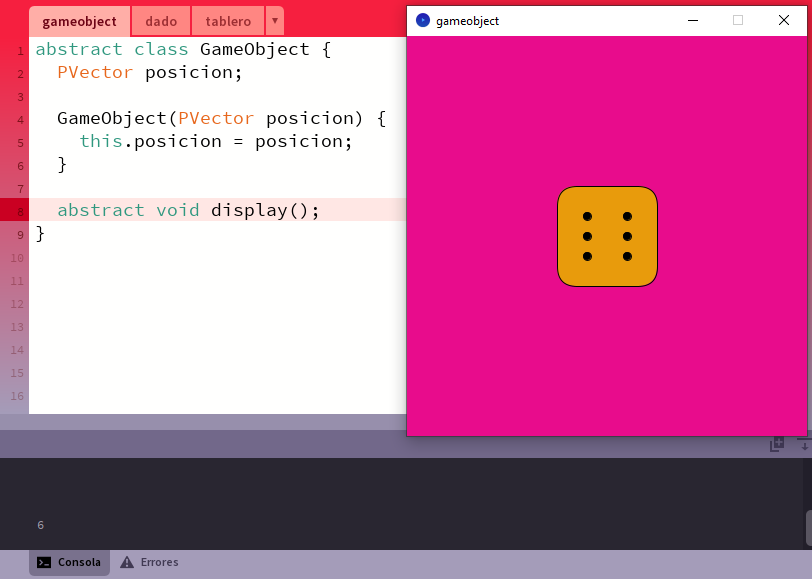
Además, almacene cada dado obtenido en un arreglo. Considere aplicar la herencia respecto de que existe una clase abstracta padre GameObject, de la que hereda la posición y el método abstracto display(). Luego recrear otra versión donde use imágenes en lugar de dibujar con las primitivas.

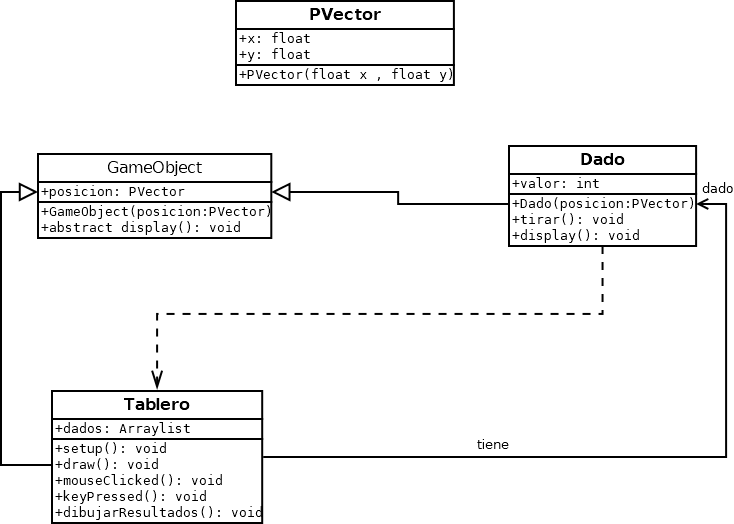
**Captura del Processing**:



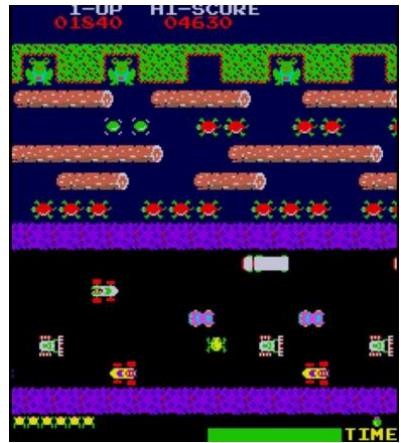


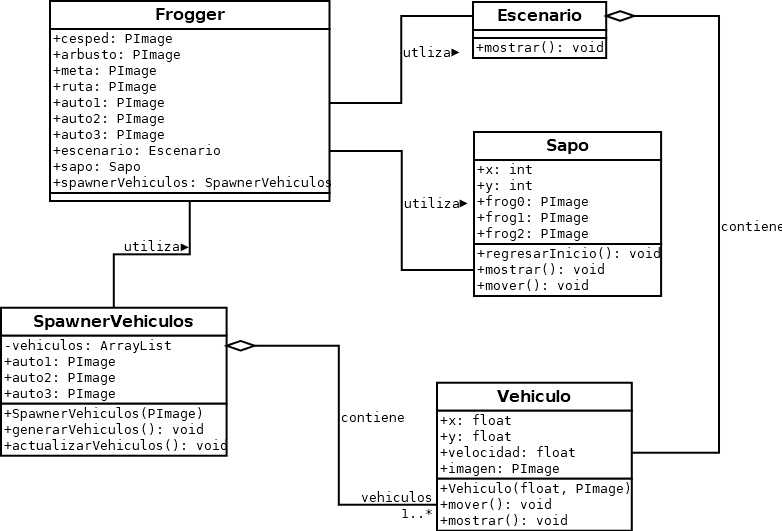




****

**Punto 3**: Realice el modelado de las clases que intervienen en el juego frogger a partir de la Fig. 1. Realice la construcción de las clases en processing. El juego debe llegar a poder mostrar en pantalla la visualización de los diferentes objetos modelados. Utilice herencia y encapsulamiento para los vehículos. Además, los vehículos deben guardarse en una lista de objetos que es atributo de la clase SpawnerVehiculos.

Figura 1. Modelo juego Fogger

**Punto 4**

Considere programar un juego de naves. Debe usar imágenes para las naves, los asteroides y los enemigos. Aplique herencia. Use una interface denominada IDisplayable que tenga el método display(). Defina dos interfaces más: IMoveable que tenga el método mover() y Otra IControler que tenga el método readCommand();

Usando el sentido común haga que las clases Nave, Asteroid y Enemy implementen las interfaces correspondientes. Finalmente use la dependencia para que la nave dispare balas que serán almacenadas en una lista de balas. Las balas se deben destruir cuando salen de pantalla.

