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.

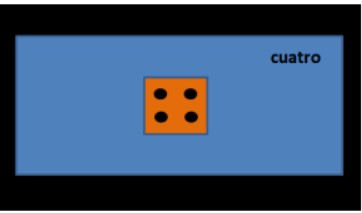
**ENLACE AL EJERCICIO 1 TPNº 2- GITHUB**

**https://github.com/users/Rodri1981/projects/1/views/1?pane=issue&itemId=61377483**

|  |
| --- |
| HISTORIA DE USUARIO |
| **Codigo:** HU001 **Usuario:** |
| **Nombre de Historia de Usuario:** creación de un escenario con movimiento de una clase GameObject la cual es abstracta y posee atributos protegidos, de la que heredan Shooter y Asteroide. |
| **Prioridad:** Media **Riesgo de desarrollo:** Media |
| **Estimacion:** tres horas **Iteracion asignada:**1 |
| **Responsable:** Rodrigo Acevedo |
| **Descripcion:**  Como jugador quiero poder observar en el escenario una nave con movimiento y que dispare a los asteroides y crear alguna estrategia para sobrevivir esquivando colisiones de los asteroides. |
| **Criterios de aceptación:**  El jugador (nave) podrá moverse en todas direcciones ya que podrá girar sobre su eje y podrá también desplazarse en todas direcciones por el escenario y esquivar los asteroides.  Los enemigos (asteroides) se moverán de forma aleatoria por todo el escenario en todas direcciones.  También tendremos a los enemigos (asteroides) de diferentes tamaños, y estos una vez que sean colisionados por un disparo se podrán partir a la mitad, generando mayor dificultad.  El jugador (nave) no tendrá limitaciones de desplazamientos sobre los limites del escenario ya que podrá traspasar las paredes del escenario y reaparecer nuevamente. El desplazamiento lo realizara con la función JOYPAD |

Punto 2: Desarrolle un videojuego

Realice un diagrama de clases y programe el siguiente juego



|  |  |  |
| --- | --- | --- |
|  | PROGRAMACIÓN PARA VIDEOJUEGOS  TECNICATURA EN DESARROLLO Y DISEÑO DE VIDEOJUEGOS  Universidad Provincial de Administración,  Tecnología y Oficios  Trabajo Práctico N° 2: Abstracción – encapsulamiento y Herencia |  |

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.

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.

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();

Mg. Ing. Ariel Alejandro Vega

|  |  |  |
| --- | --- | --- |
|  | PROGRAMACIÓN PARA VIDEOJUEGOS  TECNICATURA EN DESARROLLO Y DISEÑO DE VIDEOJUEGOS  Universidad Provincial de Administración,  Tecnología y Oficios  Trabajo Práctico N° 2: Abstracción – encapsulamiento y Herencia |  |

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.



Figura 1. Modelo juego Fogger

Mg. Ing. Ariel Alejandro Vega

Enlaces de canal del profe

[Fundamentos de Programación Orientada a Objetos - YouTube](https://www.youtube.com/playlist?list=PLMq7gzUJ4efXWns0UZJRf3M6m1XCZ2Xh5)