Programación Orientada a Objetos

Examen Final

La entrega deberá contener una solución hecha en *Visual Studio,* con un nombre indicativo del examen final y nombre del alumno. Por cada ejercicio, crear un proyecto nuevo, también indicando mediante la nomenclatura el número del ejercicio a resolverse en cada uno. El examen contará también con una sección teórica que se rendirá de forma presencial el día del examen.

Ejercicio 1

Crear un juego de consola en el que se controle a una **nave espacial**, pudiendo moverla horizontalmente a lo largo del borde inferior de la zona de juego. Además, la nave podrá disparar **balas** que saldrán lanzadas desde la punta de la nave, viajando verticalmente hasta la parte superior de la pantalla. Dichas balas podrán colisionar con **asteroides**, los cuales son obstáculos que se mueven desde la parte superior de la consola hacia la inferior. Si un **asteroide** colisiona con la nave, esta última perderá una vida. En cambio, si una **bala** colisiona con el asteroide, ambos se destruyen, y el jugador gana puntaje. Si se pierden todas las vidas, el juego se pierde; si se llega a cierto puntaje, el juego se gana.

- El **jugador**:
    - Se mueve de izquierda a derecha, limitado por las dimensiones de la zona de juego.
    - Pierde vidas al colisionar con asteroides.
    - Dispara balas.

- Las **balas**:
    - Viajan verticalmente hacia arriba al ser disparadas.
    - Destruyen asteroides al colisionar con ellos, y le suman puntos al jugador cuando ocurre.

- Los **asteroides**:
    - Viajan verticalmente hacia abajo.
    - Se destruyen al colisionar o al llegar al borde inferior, y se crean de nuevo en una posición horizontal aleatoria del borde superior.
    - Le restan una vida al jugador en caso de colisionar con éste.

Para realizar el juego se deberá crear una jerarquía de clases en la que todas las entidades (nave, balas y asteroides) partan de una misma clase base, la cual maneje la posición y el tamaño de la entidad en cuestión, así como la verificación de si colisiona con otra que reciba como parámetro. Se deberá utilizar el polimorfismo y el concepto de métodos virtuales puros para hacer que todas las entidades respondan al mensaje de "dibujarse" de un modo diferente. Se deberá hacer que todas las entidades se agreguen a un vector de entidades, para recorrerlo y llamar al método de *"draw"* de cada entidad sin distinguir por tipo específico.

El juego también deberá contar con una clase que conozca al jugador, para mostrar constantemente las vidas de la nave, el puntaje actual y la cantidad de asteroides activos. Para mostrar la cantidad de asteroides, usar un campo estático que se aumente o reste al aparecer o destruir asteroides, respectivamente. A su vez, la clase del *HUD* deberá manejar los mensajes de victoria y derrota.

Aspectos a considerar

- **Aspecto de las entidades:** dibujar a las entidades usando varios caracteres para la nave, y uno (o más) los asteroides y balas. Usar la tabla *ASCII* como referencia para dibujar los caracteres. La nave debería verse de la siguiente manera:

- **Gráfico de vidas en la interfaz:** buscar un glifo que consista en un corazón para representar la cantidad de vidas. Entonces, en lugar de mostrar *"Lives: 3"*, se debe mostrar *"Lives: ♥♥♥"*.

Criterios de evaluación

- La parte práctica se entregará el día del examen, y se revisará en el momento. Dependiendo del estado de la entrega, se determinarán tres **estados**:
  - **Satisfactorio**.
  - **Regular**.
  - **Insatisfactorio**.

- Los **estados** de la entrega principal serán consecuencia de:
  - El buen funcionamiento del programa.
  - El uso de herencia para los casos propuestos.
  - El uso de polimorfismo para los casos propuestos.
  - El uso de campos y métodos estáticos para los casos propuestos.
  - La prolijidad y elegancia del código.

- Se obtendrá un **satisfactorio** cuando se cumpla con:
  - El buen funcionamiento del programa.
  - Al menos 3 de los otros 4 puntos a considerar.

- Se obtendrá un **regular** cuando se cumpla con:
    - El buen funcionamiento del programa.
    - Al menos 2 de los otros 4 puntos a considerar.

- Se obtendrá un **insatisfactorio** cuando no se cumpla con ninguno de los casos previos.

- Si el resultado de la entrega principal es **satisfactorio**:
    - La nota base será 4.
    - Se realizarán 6 preguntas teóricas oralmente; por cada una respondida de forma correcta, se sumará 1 punto adicional.

- Si el resultado de la entrega es **regular**:
    - La nota base será 2.
    - Se realizarán 6 preguntas teóricas oralmente. Si se responden al menos 3 correctamente, la nota pasará a ser 4. Si se responden las 6 correctamente, la nota pasará a ser 6.

- Si el resultado de la entrega es **insatisfactorio**:
    - La nota será 2.

- Adicionalmente, por cada trabajo práctico de la cursada que haya quedado sin entregar durante la cursada, se restará 1 punto a la nota final adquirida.

Aclaraciones

- Utilizar los archivos de la *library* proporcionada para realizar algunas de las funciones requeridas (dibujar un marco, obtener las dimensiones de la consola, etc.).
- No es necesario que las clases, campos y métodos tengan los mismos nombres que los diagramas propuestos; es posible modificar los nombres según se crea necesario, siempre que la interfaz sea la misma.