



TECNICATURA UNIVERSITARIA EN DISEÑO
INTEGRAL DE VIDEOJUEGOS

FACULTAD DE INGENIERÍA
Universidad Nacional de Jujuy



FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS

Trabajo Practico

N°2

Facundo Leonardo Condori



TUV000667

Profesores:



Mg. Ing. Ariel Alejandro Vega

Ing. Carolina Cecilia Apaza

Año

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy PENSAMIENTO COMPUTACIONAL y PROGRAMACIÓN: Problema y Solución – PC y P – Algoritmos – Principio de la P</p>	
---	--	---

Indice

	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy PENSAMIENTO COMPUTACIONAL y PROGRAMACIÓN: Problema y Solución – PC y P – Algoritmos – Principio de la P</p>	
---	---	---

Punto 1:

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. Utilice un JoyPad para generar los movimientos.

Desarrollo del punto

Historia de Usuario	
Codigo: HU001	Usuario: Jugador
Nombre de Historia de Usuario: Construcción de escenario y ubicación de game object	
Prioridad: Alta	Riesgo de desarrollo: Alta
Estimación: 2 horas	Iteración asignada: 1
Responsable: Facundo Condori	
Descripción: Como jugador quiero poder observar en el escenario la ubicación y movimiento de mi personaje y todos los asteroides y enemigos, para poder determinar mi estrategia de juego	
Criterios de aceptación: El jugador se puede mover en 4 posiciones <div data-bbox="252 1317 526 1541" data-label="Image"> </div> Los asteroides solo se mueven de arriba hasta abajo <div data-bbox="343 1608 438 1832" data-label="Image"> </div> Debe existir una representación visual en pantalla de los GameObjects para permitir la interacción con ellos.	



Los GameObjects deben moverse según sus velocidades y direcciones respectivas para garantizar un desafío y diversión adecuados en el juego.

Se debe incluir un indicador visual en la pantalla que indique la cantidad de vidas restantes del Shooter, lo que permitirá al jugador monitorear su progreso en el juego.

El movimiento del Shooter debe ser controlado por el usuario a través de un JoyPad para proporcionar una experiencia de juego más inmersiva y cómoda.

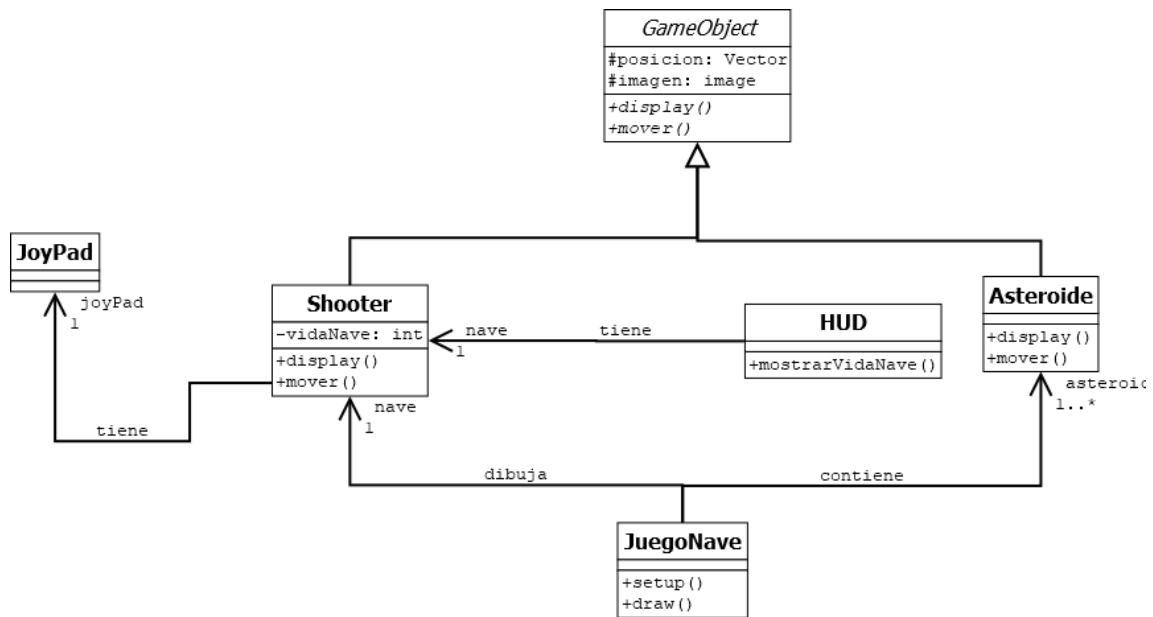
Es necesario garantizar que la clase GameObject sea abstracta y proporcione los métodos y atributos necesarios para su funcionamiento correcto.

Se deben implementar las clases Shooter y Asteroide, las cuales deben heredar de GameObject y sobrescribir los métodos abstractos según sea necesario.

Se requiere diseñar e implementar un HUD (Head-Up Display) que muestre claramente y de forma legible la cantidad de vidas restantes del Shooter.

Observaciones: En este modelo no se considera la rigidez de las paredes

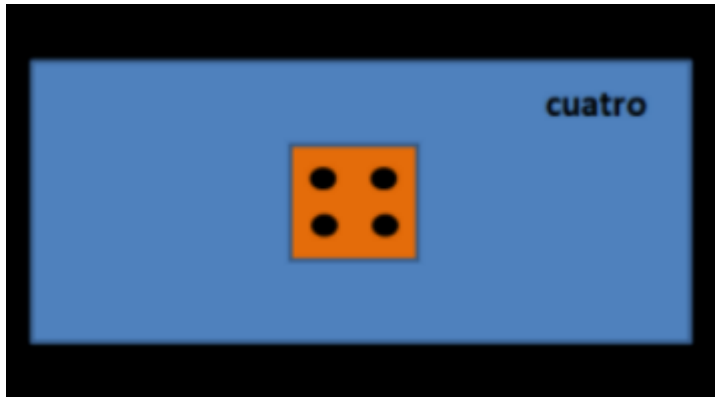
- **Visualización de GameObjects:** Debe haber una representación visual en pantalla de los GameObjects, incluyendo el Shooter y los Asteroides. Cada GameObject debe tener una posición en la pantalla y una imagen asociada que lo represente.
- **Movimiento de GameObjects:** Los GameObjects deben moverse de acuerdo con sus atributos de velocidad y dirección. El movimiento debe ser suave y realista para brindar una experiencia de juego satisfactoria.
- **Clase abstracta GameObject:** La clase GameObject debe ser abstracta y contener atributos protegidos como posición e imagen. Además, debe tener los métodos abstractos `display()` y `mover()` que serán implementados por las subclases.
- **Subclases Shooter y Asteroide:** Se deben implementar las subclases Shooter y Asteroide, que hereden de GameObject. Estas subclases deben sobrescribir los métodos abstractos según sea necesario para definir su comportamiento específico.
- **HUD para el Shooter:** Se debe diseñar e implementar un HUD que muestre claramente la cantidad de vidas restantes del Shooter en la pantalla. Este HUD debe ser visible en todo momento durante el juego para que el jugador pueda monitorear su progreso.
- **Control de movimiento con JoyPad:** El movimiento del Shooter debe ser controlado por el jugador a través de un JoyPad. Esto proporcionará una experiencia de juego más inmersiva y cómoda, permitiendo al jugador tener un control preciso sobre el movimiento del personaje.



Punto 2:

2: Desarrolle un videojuego que cumpla con las siguientes especificaciones:

Realice un diagrama de clases

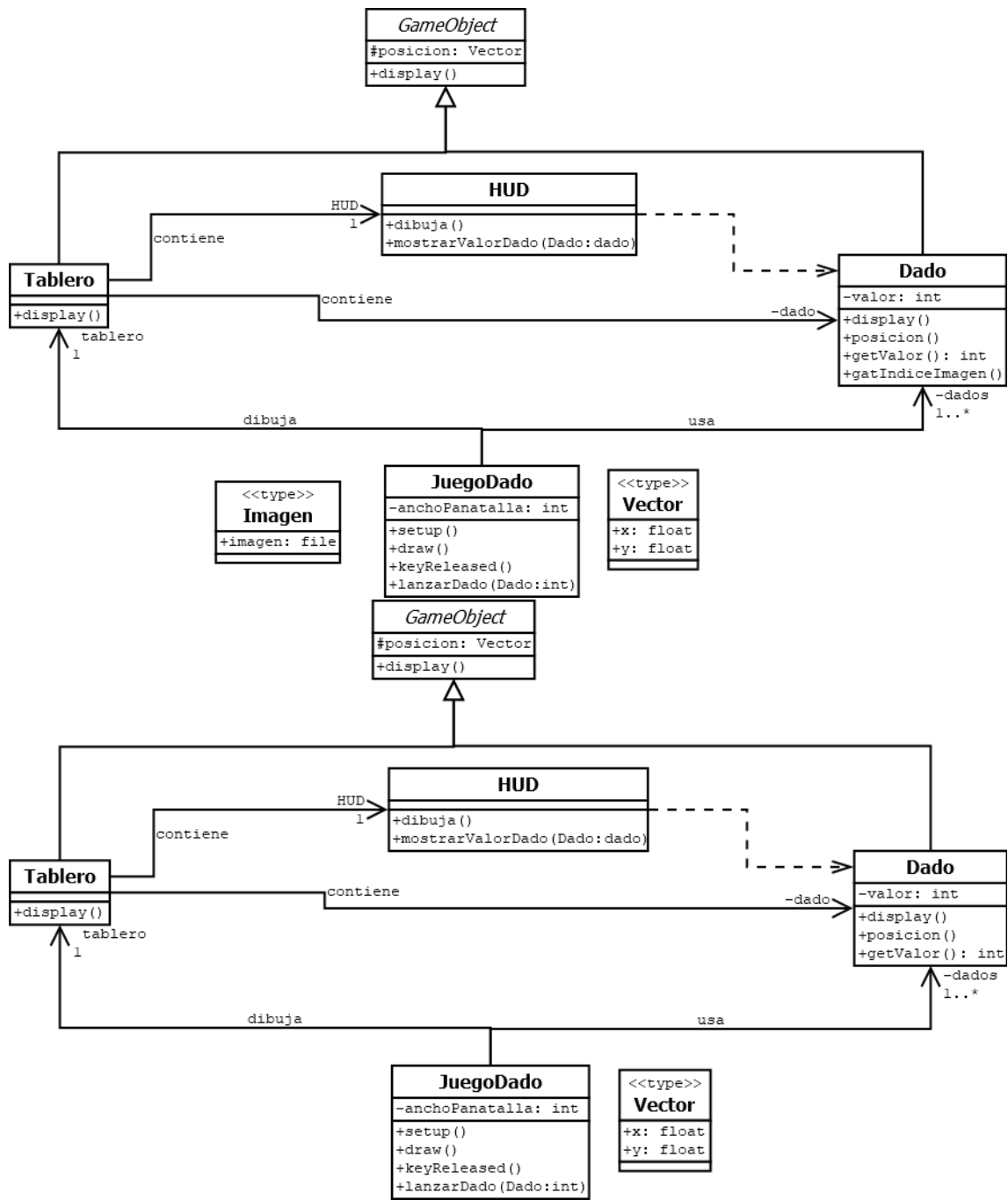


Como se observa se trata de un dado. El cual al presionar un botón debe generar un número aleatorio entre 1 y 6 y dibujarlo. Además, debe mostrar el número en la parte superior derecha. Repetir esto cuantas veces lo desee y al finalizar (con otro botón) debe dibujar por consola y agrupado en filas de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

Desarrollo del punto



JuegoDado TP02

Dado

GameObject

Hud

Tablero



```
1 class Tablero extends GameObject{
2     //atributos
3
4     //constructor por defecto
5     public Tablero() {
6     }
7
8     public void Tablero(PVector posicion){
9         this.posicion=posicion;
10    }
11
12
13    //metodos
14    public void display(){
15        rect(this.posicion.x, this.posicion.y, ((width *5)/6), ((height * 4)/6));
16        fill(0);
17    }
18
19    //metodos get
20
21    //metodos set
22
23 }
24
25
26
27
28
```




```
1 class GameObject{
2     //atributos
3     protected PVector posicion;
4
5     //constructor por defecto
6     public GameObject() {
7
8     }
9     //constructor parametrizado
10    //metodos
11    public void display(){
12    }
13    //metodos get
14    //metodos set
15 }
16
17
18
19
20
21
22
23
24
```

JuegoDado TP02

Dado

GameObject

Hud

Tablero



```
1 class Hud {
2     Dado dado;
3
4     public Hud(Dado dado) {
5         this.dado = dado;
6     }
7
8     public void display() {
9         int imagenDado = dado.getIndiceImagen();
10        textSize(50);
11        text((imagenDado+1), 500, 100);
12
13        // Centrar texto horizontalmente
14        textAlign(CENTER, CENTER);
15        textSize(30);
16        text("Presione espacio", width/2, height - 320);
17        fill(255);
18    }
19 }
```

JuegoDado TP02	Dado	GameObject	Hud	Tablero	▼
----------------	------	------------	-----	---------	---

```
1 private Tablero tablero;
2 private Dado dado;
3 private Hud hud;
4 PImage[] imagenes;
5 int imagen = 0;
6 int numeroDado;
7 int[] valoresDados = new int[5];
8
9
10 public void setup(){
11     size(600,400);
12     tablero = new Tablero();
13     tablero.Tablero(new PVector(50,50));
14     dado = new Dado();
15     imagenes = new PImage[6];
16     hud = new Hud(dado);
17
18
19
20
21     int img = 0;
22     do {
23         imagenes[img] = loadImage("dado" + img + ".png");
24         img++;
25     } while (img < imagenes.length);
26
27 }
28
29 public void draw(){
30     background(0);
31     tablero.display();
32     hud.display();
33     image(imagenes[dado.getIndiceImagen()], width/2, height/2,200,200);
34     imageMode(CENTER);
35 }
```

JuegoDado TP02	Dado	GameObject	Hud	Tablero	▼
----------------	------	------------	-----	---------	---

```
1 class Dado extends GameObject {
2     // Atributos
3     private int imagen;
4     private int[] valor;
5
6     public Dado() {
7
8         valor = new int[6];
9         posicion = new PVector(0, 0);
10    }
11    //
12    public void Posicion(int x, int y) {
13        posicion.set(x,y);
14    }
15
16    @Override
17    public void display(){
18        imagen = (int) random(imagenes.length);
19        int dado = 0;
20        while (dado < valor.length) {
21            valor[dado] = (int) random(1, 7);
22            dado++;
23        }
24    }
25
26    public int getIndiceImagen(){
27        return imagen;
28    }
29
30
31    public void setImagen(int imagen) {
32        this.imagen = imagen;
33    }
34
35    public int[] getValor(){
```



Presione espacio

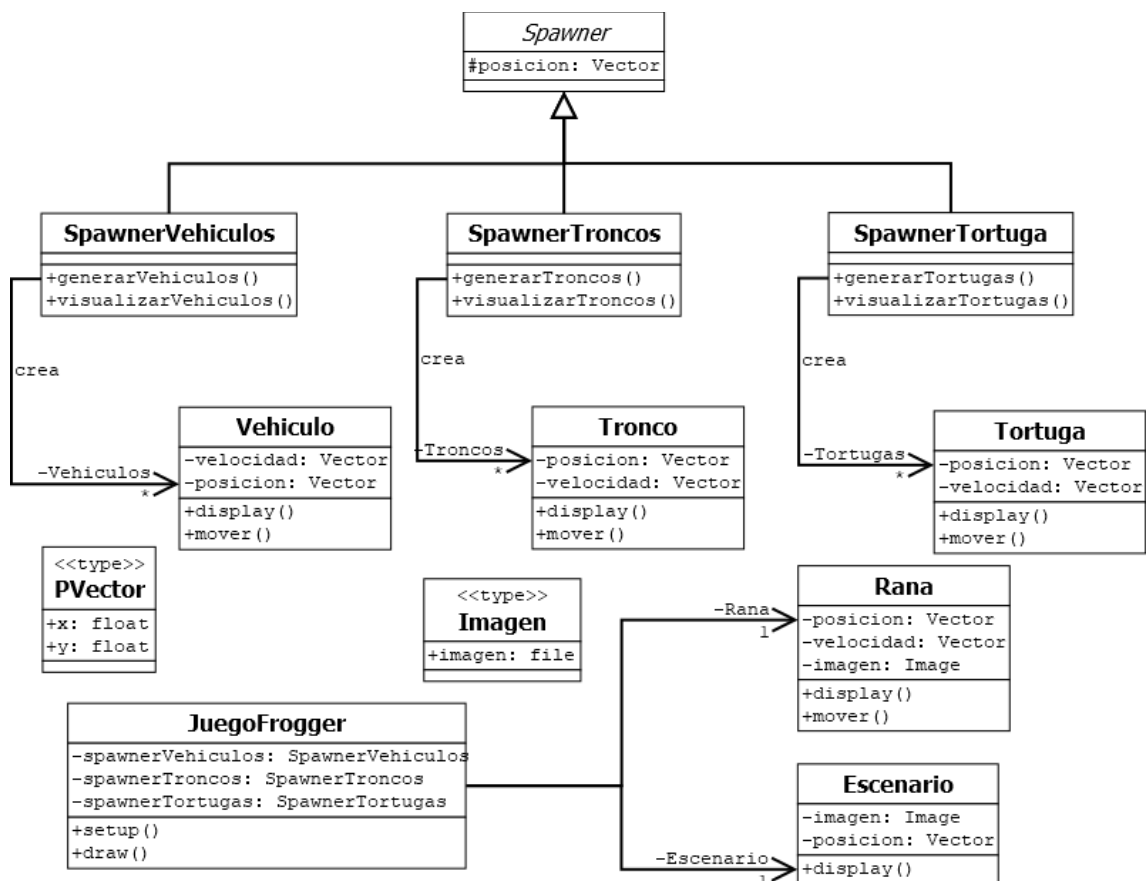


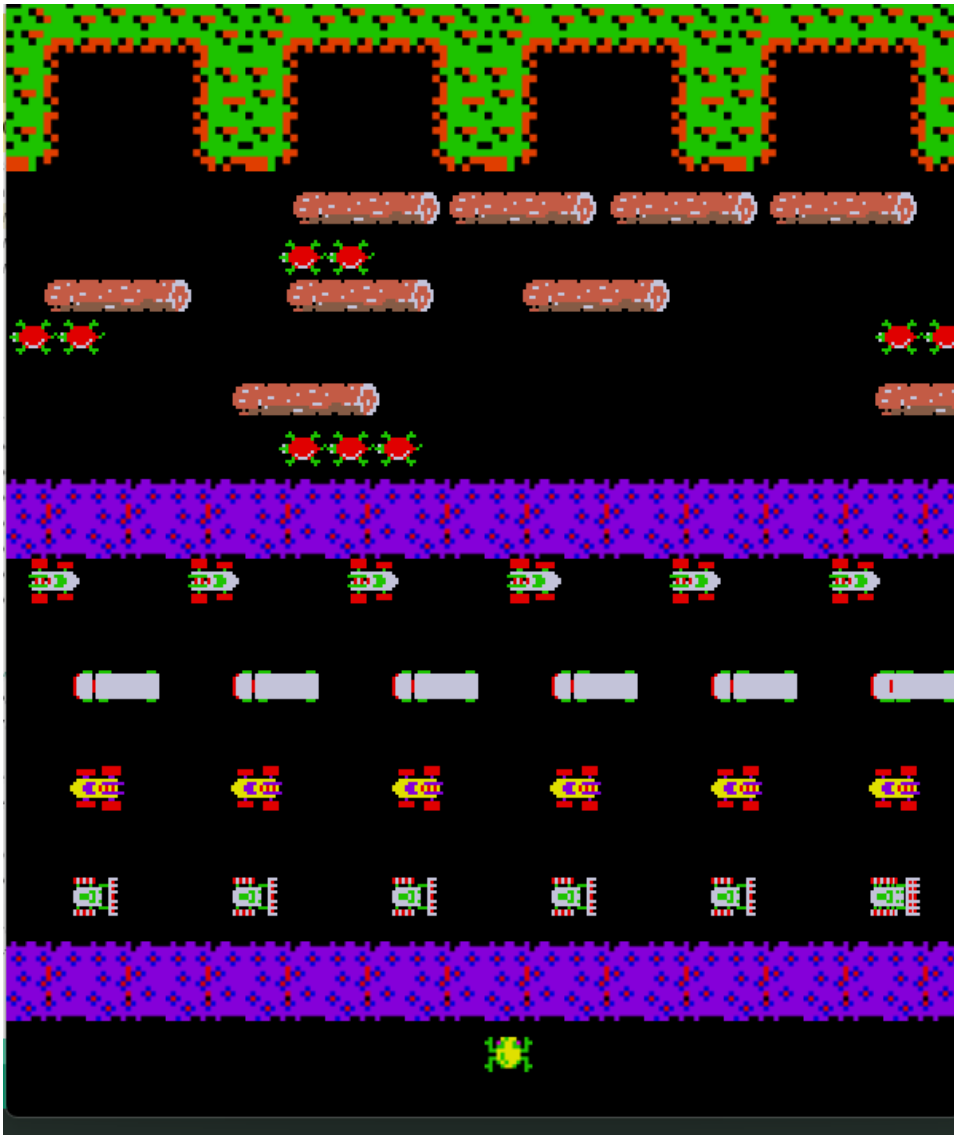
4

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.

Desarrollo del punto





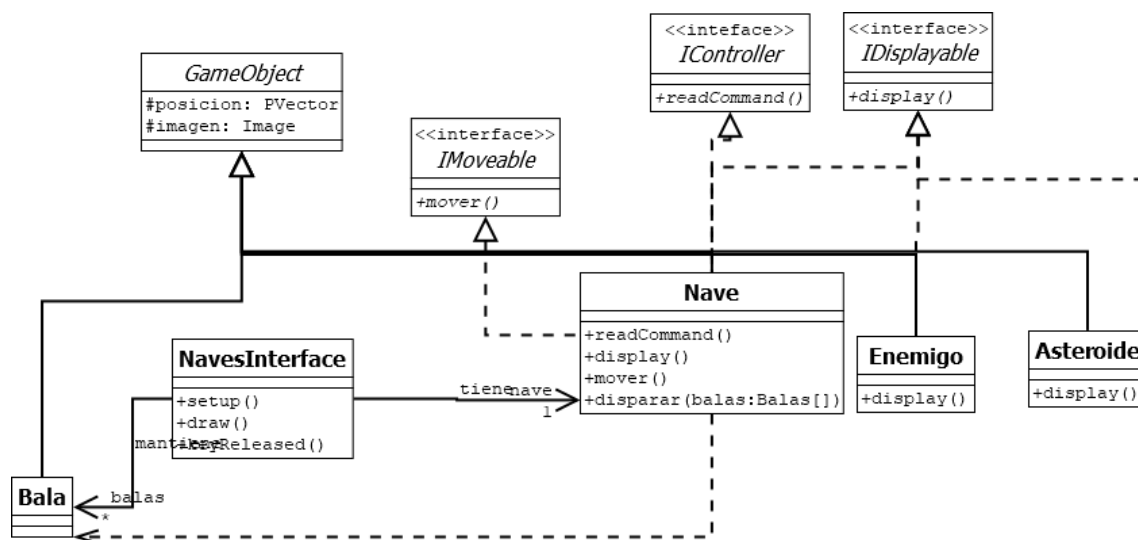
Punto 4:



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 IController 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.

Desarrollo del punto



	<p>FUNDAMENTOS DE PROGRAMACIÓN ORIENTADA A OBJETOS TECNICATURA UNIVERSITARIA EN DISEÑO INTEGRAL DE VIDEOJUEGOS FACULTAD DE INGENIERÍA Universidad Nacional de Jujuy PENSAMIENTO COMPUTACIONAL y PROGRAMACIÓN: Problema y Solución – PC y P – Algoritmos – Principio de la P</p>	
---	--	---

Conclusión

Párrafos de las conclusiones

Fuentes bibliográficas

Se deben enunciar las fuentes (apuntes de la materia, páginas web, videos de youtube, libro (nombre, autores, año), etc)