

### LABORATORIO 3

#### Requerimientos funcionales

<b>Nombre</b>	<b>R.# 1. Capturar Pac-Mac.</b>
<b>Resumen</b>	Permite que el usuario pueda atrapar un Pac-Mac. Esto lo puede realizar presionando clic justo en cima del Pac-Mac.
<b>Entradas</b>	
- clic en el Pac-Mac.	
<b>Resultados</b>	
Se atrapa y desaparece de la ventana el Pac-Mac seleccionado.	

<b>Nombre</b>	<b>R.# 2. Configurar nuevo juego.</b>
<b>Resumen</b>	Permite cargar un archivo de texto en el cuál se tendrá la configuración de un nuevo juego. Esto se realiza a través de un menú.
<b>Entradas</b>	
- Nivel de dificultad del juego - diámetro del Pac-Mac - posición en x - posición en y - tiempo de espera - dirección inicial - cantidad de rebotes - estado (¿detenido?)	
<b>Resultados</b>	
Se carga el archivo y se configura un nuevo juego con todos los valores introducidos en el.	

<b>Nombre</b>	<b>R.# 3. Guardar estado actual.</b>
<b>Resumen</b>	Permite guardar el estado actual del juego, mediante un archivo de texto con los valores actuales de los Pac-Mac's.
<b>Entradas</b>	
Ninguno	
<b>Resultados</b>	

Se guarda el estado actual de juego en un archivo de texto.

<b>Nombre</b>	<b>R.# 4. Guardar los mejores puntajes.</b>
<b>Resumen</b>	Permite generar un salón de la fama en el cual se guardará los mejores 10 puntajes que se vayan registrando en el juego. Esta información debe guardarse cuando se cierre la aplicación y se puede consultar al abrirla de nuevo.
<b>Entradas</b>	
Ninguno	
<b>Resultados</b>	
Se guardan los mejores 10 puntajes.	

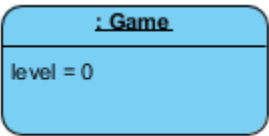
#### Requerimientos no funcionales

<b>Nombre</b>	<b>R.# 1. Interactuar con usuario.</b>
<b>Resumen</b>	Permite que, mediante una interfaz gráfica amigable, el usuario lleve a cabo los requerimientos y así mismo el desarrollo de la aplicación.
<b>Entradas</b>	
- Ninguna.	
<b>Resultados</b>	
Se muestra la interfaz gráfica al usuario.	

[Diagrama de clases del modelo y la interfaz](#)

## Diseño de casos de pruebas unitarias

### Configuración de los Escenarios

Nombre	Clase	Escenario
setupScenary1	Game	vacío
setupScenary2	Game	 <pre> classDiagram     class Game {         level = 0     }         </pre>
setupScenary1	PacMan	vacío
setupScenary1	Score	vacío

### Diseño de Casos de Prueba

Objetivo de la Prueba: Verificar la correcta creación de un juego y el correcto funcionamiento de los métodos triviales				
Clase	Método	Escenario	Valores de Entrada	Resultado
Game	game	setupScenary1	Level = 1	Se ha creado un nuevo juego exitosamente. La relación con PacMan representada como lista no es nula al momento de crear el juego. Así mismo sucede con la relación con Score.
PacMan	pacMan	setUpScenary1	radius = 50 posX = 150 posY = 69 waitTime = 50 direction = "RIGHT"	Se ha creado un nuevo PacMan exitosamente. Los métodos getters, devuelven el valor correcto. Igualmente, los métodos setters cambian correctamente los valores de

			bounces = 0 state = false	los atributos de la clase PacMan.
Score	score	setUpScenary1	Ninguno	Se ha creado un nuevo Score exitosamente. Los puntajes representado por medio de mapas, se inicializan correctamente, no son nulos.

**Objetivo de la Prueba:** Verificar la lectura correcta del archivo del juego para cargar

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	loadGame	setupScenary2	ninguno	El método efectivamente encuentra el archivo y lo lee correctamente, convirtiendo la información dados en atributos de un nuevo PacMan.

**Objetivo de la Prueba:** Verificar que el estado actual del juego se guarda correctamente en un archivo de texto

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	saveGame	setupScenary2	ninguno	El método efectivamente obtiene la información actual de la aplicación y la convierte en un archivo de texto que será leído más adelante por el método anteriormente escrito.

**Objetivo de la Prueba:** Verificar que el estado de las puntuaciones persiste cuando se cierra la aplicación

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	saveScore	setupScenary2	ninguno	El método efectivamente guarda la tabla de mejores puntajes (sala de fama) mediante serialización.

**Objetivo de la Prueba:** Verificar que se carga correctamente la tabla de los mejores puntajes

Clase	Método	Escenario	Valores de Entrada	Resultado
-------	--------	-----------	--------------------	-----------

Game	loadScore	setupScenary2	ninguno	El método efectivamente carga la tabla de mejores puntajes (sala de fama) mediante serialización.
------	-----------	---------------	---------	---

## Trazabilidad de análisis al diseño

Requerimiento funcional	Método	Clase
R1. Capturar Pac-Man	catchPac() getPac() desactive getPacMan() setState()	PacManController PacThread PacThread PacThread PacMan
R2. Configurar nuevo juego	initialize() loadSaveGame(ActionEvent event) level0(ActionEvent event) level1(ActionEvent event) level2(ActionEvent event) loadGame(String path, String sep) createPacMans(List<PacMans> pacM) start() run() catchPac : getPacMans()	PacManController PacManController PacManController PacManController PacManController PacMan PacManController PacThread PacThread PacManController PacMan
R3. Guardar estado actual	saveGame(Sting Path) saveGame(ActionEvent evento)	PacMan PacManController
R4. Guardar los mejores puntajes	loadScore() saveScore() viewScores(ActionEvent event) getScore() getHallOfFame0() getHallOfFame1() etHallOfFame2()	PacMan PacMan PacManController Game Score Score Score