Nicolás Penagos Montoya
A00347293

UNIDAD 3 – Laboratorio 3

1. Especificación de requerimientos.

Funcionales

Nombre:	**R-1 Mover pacmans**
Resumen	Al momento de iniciar/cargar el juego, los pacmans deben moverse en sentidos verticales u horizontales, adicionalmente estos deben rebotar cuando lleguen a los límites de la escena y por ultimo estos deben rebotar cuando choquen entre sí.
Entradas	
Resultado	Los pacmans se mueven, rebotan y colisionan correctamente toda vez el juego es iniciado.

Nombre:	**R-2 Detener pacmans**
Resumen	Cuando el jugador de clic sobre el área de un pacman , este debe detenerse.
Entradas	
Resultado	El pacman es detenido

Nombre:	**R-3 Guardar juego**
Resumen	El programa debe tener también la posibilidad de guardar el estado actual en un archivo de texto con un formato específico. (level, radio posX, posY, sleep, movement, direction , bouncings, caught).
Entradas	
Resultado	Un archivo de texto es creado, escrito y guardado en la carpeta correspondiente.

Nombre:	**R-4 Cargar juego**
Resumen	La configuración de un nuevo juego se puede (debe) cargar desde un archivo de texto a través de la opción del menú. Formato del archivo de texto: level, radio posX, posY, sleep, movement, direction , bouncings, caught.
Entradas	
Resultado	El juego es cargado e iniciado con los valores correspondientes al archivo correspondiente previamente guardado.

Nombre:	**R-5 Llevar puntuación**
Resumen	Cada que un pacman rebote o colisione la puntuación del jugador será aumentada
Entradas	
Resultado	La puntuación se aumenta

Nicolás Penagos Montoya
A00347293

Nombre:	**R-6 Terminar juego y guardar puntaje alto**
Resumen	Cuando el jugador detenga todos los pacmans, el programa debe terminar el juego, y este debe pedir el nombre del jugador para guardar su puntaje
Entradas	
Resultado	El mensaje de finalización es mostrado y el nombre del jugador es pedido.

Nombre:	**R-7 Mostrar puntaje**
Resumen	Cuando el jugador seleccione la opción correspondiente, el programa debe mostrar los mejores puntajes junto con el nivel en el que fueron conseguidos.
Entradas	
Resultado	Los puntajes son mostrados en pantalla.

Nombre:	**R-8 Serializar puntaje**
Resumen	Tanto la escritura y la lectura de los puntajes se llevará a cabo a través de serialización. El objeto que se serializará será el objeto de la clase score. Al momento de iniciar una nueva partida el objeto debe ser deserializado.
Entradas	
Resultado	El objeto puntaje ha sido serializado, el archivo ha sido guardado en el lugar correspondiente.

No funcionales

Nombre: Interfaz JavaFx	Implementar el programa de computador con interfaz gráfica de JavaFx

Nicolás Penagos Montoya
A00347293

2. Trazabilidad del análisis del diseño

Requerimientos	Clase	Método
R-1 Mover pacmans	Main	main(String): void
		Start(stage): void
	PacManController	Update( ): void
		Initialize( ): void
	PacManThread	run( ): void
		PacManThread(long, PacManModel)
	GUIUpdateControllThread	GUIUpdateControllThread(PacManController)
		run( ): void
	GUIUpdateRunnable	GUIUpdateRunnable(PacManController)
		run( ): void
	Game	Game(int)
	PacManModel	PacManModel( )
		Move( ): void
		Colision( )): void
R-2 Detener pacmans	Main	main(String): void
		Start(stage): void
	PacManController	Update( ): void
		Stop(Event): void
		Initialize( ): void
	PacManThread	run( ): void
		PacManThread(long, PacManModel)
	GUIUpdateControllThread	GUIUpdateControllThread(PacManController)
		run( ): void
	GUIUpdateRunnable	GUIUpdateRunnable(PacManController)
		run( ): void
	PacManModel	PacManModel( )
		caught( ): void

R-3 Guardar juego	Main	main(String): void
		Start(stage): void
	PacManController	Initialize( ): void
		SaveSlot1( ): void
		SaveSlot2( ): void
		SaveSlot3( ): void
	Game	Game(int)
		saveGame(int): void
R-4 Cargar juego	Main	main(String): void
		Start(stage): void
	PacManController	Initialize( ): void
		loadNewHardGame( ): void
		loadNewMediumGame( ): void
		loadNewEasyGame( ): void
		LaodSlot1( ): void
		LoadSlot2( ): void
		LaodSlot3( ): void
	Game	Game(int)
		loadNewFile(int): void
R-5 Llevar puntuaciion	Main	main(String): void
		Start(stage): void
	PacManController	Update( ): void
		Initialize( ): void
	PacManThread	run( ): void
		PacManThread(long, PacManModel)
	GUIUpdateControllThread	GUIUpdateControllThread(PacManController)
		run( ): void
	GUIUpdateRunnable	GUIUpdateRunnable(PacManController)
		run( ): void
	Game	Game(int)
	PacManModel	PacManModel( )
		Move( ): void
		Colision( ): void

Nicolás Penagos Montoya
A00347293

R-6 Terminar juego y guardar puntaje alto	Main	main(String): void
		Start(stage): void
	PacManController	Update( ): void
		Initialize( ): void
	PacManThread	run( ): void
		PacManThread(long, PacManModel)
	GUIUpdateControllThread	GUIUpdateControllThread(PacManController)
		run( ): void
	GUIUpdateRunnable	GUIUpdateRunnable(PacManController)
		run( ): void
	Game	Game(int)
		totalBouncings( ): int
		win( ): void
	Scores	Scores( )
		isBest(String, int, int)
		order(String [ ]): String[ ]
	PacManModel	PacManModel( )
		caught( ): void
R-7 Mostrar puntaje	Main	main(String): void
		Start(stage): void
	PacManController	initialize( ): void
		Show( )
	Scores	Scores( )
		getScores( )
R-8 Serializar puntaje	Main	main(String): void
		Start(stage): void
	PacManController	initialize( ): void
		save( ): void
		load( ): void