

## UNIDAD 5 – Laboratorio 5

### 1. Especificación de requerimientos.

Funcionales:

R-1 Generar vuelos aleatorios	
Resumen	Generar aleatoriamente una lista enlazada de vuelos en diferentes fechas, horarios, diferentes areolineas, diferentes números de vuelo (éste debe ser único), ciudades destino y puertas de embarque.
Entradas	
Resultado	Se generan diferentes vuelos, todos con sus respectivos atributos diferentes y correctamente asignados.

R-2 Mostrar vuelo e información en pantalla	
Resumen	Se introducirá el desarrollo de una nueva pantalla de información de vuelos, esta será interactiva y permitirá que el usuario busque un vuelo u reordene estos de acuerdo a diferentes criterios. Esta pantalla debe ser paginada y poseer los botones de control necesarios para navegar entre las diferentes pantallas. (En cada pantalla se muestra un número limitado de n vuelos).
Entradas	
Resultado	Los vuelos son correctamente desplegados en pantalla, esta permita correctamente su navegación e interacción.

R-3 Ordenamiento de vuelos	
Resumen	El programa debe permitir el ordenamiento de los vuelos por cualquier criterio que el usuario seleccione (fecha, aerolínea, número de vuelo, ciudad de destino, y puerta de embarque), este ordenamiento debe realizarse por medio de un algoritmo de ordenamiento diferente.
Entradas	
Resultado	Los vuelos son correctamente ordenados con base al criterio seleccionado.

R-4 Búsqueda de vuelos	
Resumen	El programa debe permitir el la búsqueda de un vuelo por cualquier criterio que el usuario seleccione.
Entradas	
Resultado	El vuelo es correctamente buscado y mostrado en pantalla.

No funcionales:

<b>Interfaz JavaFx</b>	Implementar el programa de computador con interfaz gráfica en JavaFx.
<b>Algoritmos de ordenamiento</b>	Utilizar los diferentes algoritmos de ordenamiento y búsqueda. Utilizar listas enlazadas como estructura de datos principal.

## 2. Tabla de Trazabilidad

Requerimientos	Clase	Método
R-1 Generar vuelos aleatorios	Main	main
		start
	Controller	generate
	Airport	loadTextFileToLinkedList
		isUnic
	Flight	generateFlights
	CustomHour	Flight
R-2 Mostrar vuelos e información en pantalla	CustomDate	CustomHour
	Main	CustomDate
		main
	Controller	start
		initialize
		Update
	GUIUpdateControllThread	UpdateList
R-3 Ordenamiento de los vuelos	GUIUpdateRunnable	run
	Airport	run
		getFlights
	Main	main
		start
		sortByHour
		sortByFlightNumber
		sortByAirline
	Controller cocktailSortDestine	sortByTerminal
		sortByDate
		sortByDestine
		cocktailSortComparatorFullHour
	Airport	cocktailSortGate
		cocktailSortComparatorDate
		cocktailSortFlightNumber
		cocktailSortDestine
		calculateTime
		calculateTime2
R-4 Busqueda de los vuelos	Flight	compareTo
	CustomHourComparator	compareTo
	DestineComparatoe	compareTo
	Main	main
		start
	Controller	search
	Airport	searchByTimeLinearS
		searchFlightLinearS
		searchDateLinearS
		searchDestineLinearS
		searchByGateBinaryS
		calculateTime
		calculateTime2
		printF

### 3. Diseño de pruebas unitarias

#### Airport

Nombre	Clase	Escenario
setUpScenary1	AirportTest	
setUpScenary2	AirportTest	<div>:Airport</div>
setUpScenary3	AirportTest	<div>:Flight</div>

**Objetivo de la Prueba:** Verificar la correcta creación de un objeto aeropuerto y el correcto funcionamiento de getters y setters,

Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	Airport	setupScenary1		Se ha creado un nuevo aeropuerto exitosamente.
Airport	getters	setupScenary1		Los getters retornan los valores correctamente
Airport	setters	setupScenary1		Los setters cambian los valores correctamente

**Objetivo de la Prueba:** Verificar la correcta creación de vuelos aleatorios, con número de vuelo únicos y verificar que se actualicen.

Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	generateFlights	setupScenary2		Se ha generado un nuevo arreglo de vuelos aleatorios exitosamente
Airport	isUnic	setupScenary2		Los el número de vuelo es son únicos

**Objetivo de la Prueba:** Verificar que se tome el tiempo correctamente.

Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	calculateTime1	setupScenary2		El tiempo es calculado y retornado
Airport	calculateTime2	setupScenary2		El tiempo es calculado y retornado

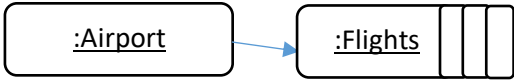
**Objetivo de la Prueba:** Verificar que el programa de computador ordene correctamente los vuelos de acuerdo a los diferentes parámetros

Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	sortByHour	setupScenary2		El arreglo esta ordenado por hora
Airport	sortByDate	setupScenary2		El arreglo esta ordenado por fecha
Airport	sortByAirline	setupScenary2		El arreglo esta ordenado por aerolínea
Airport	sortByDestine	setupScenary2		El arreglo esta ordenado por destino
Airport	sortByFlightNumber	setupScenary2		El arreglo esta ordenado por numero de vuelo

**Objetivo de la Prueba:** Verificar que el programa de computador busque y retorne correctamente los vuelos de acuerdo a los diferentes parámetros

Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	searchByHour	setupScenary2		El vuelo indicado es retornado correctamente
Airport	seacrByDate	setupScenary2		El vuelo indicado es retornado correctamente
Airport	searchByAirline	setupScenary2		El vuelo indicado es retornado correctamente
Airport	seacrByDestine	setupScenary2		El vuelo indicado es retornado correctamente
Airport	searchByFlightNumber	setupScenary2		El vuelo indicado es retornado correctamente

## Flight

Nombre	Clase	Escenario
setUpScenary1	AirportTest	
setUpScenary2	FlightTest	

**Objetivo de la Prueba:** Verificar la correcta creación de un objeto vuelo y el correcto funcionamiento de getters y setters,

Clase	Método	Escenario	Valores de Entrada	Resultado
Airport	Flight	setupScenary1		Se ha creado un nuevo aeropuerto exitosamente.
Airport	Flight	setupScenary1		Los getters retornan los valores correctamente
Airport	Flight	setupScenary1		Los setters cambian los valores correctamente

**Objetivo de la Prueba:** Verificar el correcto funcionamiento del método to string,

Clase	Método	Escenario	Valores de Entrada	Resultado
Flight	toString	setupScenary2		Se imprime el vuelo de manera correcta

**Objetivo de la Prueba:** Verificar el correcto funcionamiento del método compareTo,

Clase	Método	Escenario	Valores de Entrada	Resultado
Flight	compareTo	setupScenary2		El método retorna el valor esperado

### CustomHour

<b>Objetivo de la Prueba:</b> Verificar la correcta creación de un objeto hora y el correcto funcionamiento de getters y setters,				
Clase	Método	Escenario	Valores de Entrada	Resultado
CustomHour	CustomHour	setupScenary1		Se ha creado un nuevo aeropuerto exitosamente.
CustomHour	CustomHour	setupScenary1		Los getters retornan los valores correctamente
CustomHour	CustomHour	Flight		Los setters cambian los valores correctamente

<b>Objetivo de la Prueba:</b> Verificar el correcto funcionamiento del método to string,				
Clase	Método	Escenario	Valores de Entrada	Resultado
CustomHour	toString	setupScenary2		Se imprime la hora de manera correcta

### CustomDate

<b>Objetivo de la Prueba:</b> Verificar la correcta creación de un objeto fecha y el correcto funcionamiento de getters y setters,				
Clase	Método	Escenario	Valores de Entrada	Resultado
CustomDate	CustomDate	setupScenary1		Se ha creado un nuevo aeropuerto exitosamente.
CustomDate	CustomDate	setupScenary1		Los getters retornan los valores correctamente
CustomDate	CustomDate	setupScenary1		Los setters cambian los valores correctamente

<b>Objetivo de la Prueba:</b> Verificar el correcto funcionamiento del método to string,				
Clase	Método	Escenario	Valores de Entrada	Resultado
CustomHour	toString	setupScenary2		Se imprime la hora de manera correcta