Pruebas Unitarias:

Configuración de los Escenarios de la clase Game:

Nombre	Clase	Escenario
setUpStage1	GameTest	User name= “Fernanda”, nickname= “fernandarojas152”, gender=”femenine”, password= “fernanda”

Nombre	Clase	Escenario
setUpStage2	GameTest	No se ha creado ningún jugador, la lista está vacía y solo se inicializa el juego.

Nombre	Clase	Escenario
setUpStage3	GameTest	User name= “Fernanda”, nickname= “fernandarojas152”, gender=”femenine”, password= “fernanda” User2 name= “Angelica”, nickname= “angelica2013”, gender=”femenine”, password= “angelica”

Nombre	Clase	Escenario
setUpStage4	GameTest	User name= “Fernanda”, nickname= “fernandarojas152”, gender=”femenine”, password= “fernanda” User2 name= “Angelica”, nickname= “angelica2013”, gender=”femenine”, password= “angelica” User3 name= “Amanda”, nickname= “amandaR”, gender=”femenine”, password= “angelica” User4 name= “Saru”, nickname= “saruhiko”, gender=”male”, password= “angelica”

Nombre	Clase	Escenario
setUpStage5	GameTest	User name= “Fernanda”, nickname= “fernandarojas152”, gender=”femenine”, password= “fernanda”

Nombre	Clase	Escenario
setUpStage6	GameTest	Artist a= new Artist("Ed Sheeran", "England", "Atlantic Records"); Artist b= new Artist("One Direction", "England", "SYCO Music"); Artist c= new Artist("Taylor Swift", "USA", "Universal Music");

Diseño de Casos de Prueba:

Objetivo de la Prueba:. El programa es capaz de verificar que existen usuarios registrados previamente dentro del usuario y no permite ingresar uno con el mismo apodo.

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	addUser()	setUpStage1	User2 name= “Fernanda”, nickname= “fernandarojas152”, gender=”femenine”, password= “fernanda”	UserAlreadyExistsException

Objetivo de la Prueba:. El programa busca un usuario dentro del sistema y si no lo encuentra avisa al usuario de que este no existe aún.

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	searchUser()	setUpStage1	ninguno	UserDoesntExistException

Objetivo de la Prueba:. El programa verifica si la lista que almacena a los usuarios tiene alguno dentro de esta, por lo que busca dentro de una lista vacía a una persona y manda un mensaje avisando al usuario

Clase	Método	Escenario	Valores de Entrada	Resultado

Game	searchUser()	setUpStage2	ninguno	UserDoesntExistException

Objetivo de la Prueba:. Se verifica que el usuario no pueda agregar una nueva cuenta si este no rellena todos los campos de registro.

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	addUser()	setUpStage5	User3 name= “”, nickname= “amandaR”, gender=””, password= “amanda”	RequiredFieldsException

Objetivo de la Prueba:. El programa es capaz de ordenar una lista de usuarios en orden alfabetico.

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	sortUsers()	setUpStage4	ninguno	Angelica2013

Objetivo de la Prueba:. El programa es capaz de buscar a un usuario y verificar que este existe

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	searchUser()	setUpStage3	ninguno	True, angelica2013 existe

Objetivo de la Prueba:. El programa es capaz de buscar un usuario y al no ser encontrado notificarlo por medio de un mensaje.

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	searchUser()	setUpStage3	ninguno	UserDoesntExistException

Objetivo de la Prueba:. El programa añade artistas

Clase	Método	Escenario	Valores de Entrada	Resultado
Game	addArtist()	setUpStage2	Artist a= new Artist("Ed Sheeran", "England", "Atlantic Records");	true

Objetivo de la Prueba:. El programa es capaz de ordenar un artista en preorden				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
Game	preOrderSort()	setUpStage6	ninguno	true

Objetivo de la Prueba:. El programa es capaz de ordenar un artista en posorden				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
Game	posOrderSort()	setUpStage6	ninguno	true

Objetivo de la Prueba El programa es capaz de ordenar un artista en inorden				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
Game	inOrderSort()	setUpStage6	ninguno	true

Configuración de los Escenarios de la clase User:

Nombre	Clase	Escenario

setUpStage1	User	User name= “Fernanda”, nickname= “fernandarojas152”, gender=”femenine”, password= “fernanda” User.accumulatePoints(5)

Nombre	Clase	Escenario
setUpStage2	User	User name= “Fernanda”, nickname= “fernandarojas152”, gender=”femenine”, password= “fernanda” User.accumulatePoints(10) User.setPoints(user.getPoints()-20)

Diseño de Casos de Prueba:

Objetivo de la Prueba:. Verifica que se vaya acumulando los puntos obtenidos por el usuario.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
User	acumulatePoints()	setUpStage1	User.accumulatePoints(10)	25

Objetivo de la Prueba:. Verifica que se eliminen puntos cuando estos se gastan.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
User	acumulatePoints()	setUpStage2	ninguno	0

Configuración de los Escenarios de la clase de la clase MusicLibrary:

Nombre	Clase	Escenario
setUpStage1	MusicLibrary	Se instancia la nueva librería musical.

Nombre	Clase	Escenario
setUpStage2	MusicLibrary	Se añade una nueva canción. "./src/give_me_love.mp3"

Diseño de Casos de Prueba:

Objetivo de la Prueba:. El programa es capaz de agregar una nueva canción				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
MusicLibrary	addSong()	setUpStage1	"./src/give_me_love.mp3"	True, se agrega correctamente

Objetivo de la Prueba:. El programa no permite ingresar dos canciones iguales dentro de la lista de reproducción para la ruleta				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
MusicLibrary	addSong()	setUpStage2	"./src/give_me_love.mp3"	SongAlreadyExistException

Objetivo de la Prueba:. El programa es capaz de buscar una canción dentro de la lista de reproducción.				
Clase	**Método**	**Escenario**	**Valores de Entrada**	**Resultado**
MusicLibrary	search()	setUpStage2	ninguno	True, la canción existe.