# *ESPECIFICACIÓN DE REQUERIMIENTOS FUNCIONALES*

| ***Nombre:*** | ***RF1 – Registrar un clan*** |
|---|---|
| ***Resumen:*** | *El sistema permite registrar un clan, no pueden existir dos clanes con el mismo nombre.* |
| ***Entrada:*** | *El* ***nombre*** *del clan.* |
| ***Salida:*** | *Se ha registrado satisfactoriamente al clan.* |

| ***Nombre:*** | ***RF2 – Registrar un personaje*** |
|---|---|
| ***Resumen:*** | *El sistema permite registrar un clan, no pueden existir dos personajes con el mismo nombre.* |
| ***Entrada:*** | *(1) El* ***nombre*** *del personaje, (2) su* ***personalidad****, (3) su* ***fecha de creación****, (4) su* ***poder*** *y (5) sus* ***técnicas****.* |
| ***Salida:*** | *Se ha registrado satisfactoriamente al personaje* |

| ***Nombre:*** | ***RF3 – Registrar una técnica*** |
|---|---|
| ***Resumen:*** | *El sistema permite registrar una técnica. El mismo personaje no puede tener dos técnicas con el mismo nombre.* |
| ***Entrada:*** | *(1) El* ***nombre*** *de la técnica y (2) su* ***factor****.* |
| ***Salida:*** | *Se ha registrado satisfactoriamente la técnica.* |

| ***Nombre:*** | ***RF4 – Eliminar un clan*** |
|---|---|
| ***Resumen:*** | *El sistema permite eliminar un clan.* |
| ***Entrada:*** | *El **nombre** del clan.* |
| ***Salida:*** | *El clan ha sido eliminado satisfactoriamente.* |

| ***Nombre:*** | ***RF5 – Eliminar un personaje*** |
|---|---|
| ***Resumen:*** | *El sistema permite eliminar a un personaje.* |
| ***Entrada:*** | *El **nombre** del personaje.* |
| ***Salida:*** | *El personaje ha sido eliminado satisfactoriamente.* |

| ***Nombre:*** | ***RF6 – Eliminar una técnica*** |
|---|---|
| ***Resumen:*** | *El sistema permite eliminar una técnica.* |
| ***Entrada:*** | *El **nombre** de la técnica.* |
| ***Salida:*** | *La técnica ha sido eliminada satisfactoriamente.* |

| ***Nombre:*** | ***RF7 – Actualizar un clan*** |
|---|---|
| ***Resumen:*** | *El sistema permitir actualizar los atributos de un clan (nombre)* |
| ***Entrada:*** | *(1) El atributo que desea cambiar (Expresado como un número entero) y (2) El nuevo valor del atributo* |

| | |
|---|---|
| ***Salida:*** | *El clan fue actualizado satisfactoriamente* |

| ***Nombre:*** | ***RF8 – Actualizar un personaje*** |
|---|---|
| ***Resumen:*** | *El sistema permitir actualizar los atributos de un personaje (nombre, personalidad, fecha de creación y nivel de poder)* |
| ***Entrada:*** | *(1) El atributo que desea cambiar (Expresado como un número entero) y (2) El nuevo valor del atributo* |
| ***Salida:*** | *El personaje fue actualizado satisfactoriamente* |

| ***Nombre:*** | ***RF9 – Actualizar una técnica*** |
|---|---|
| ***Resumen:*** | *El sistema permitir actualizar los atributos de una técnica (nombre y factor)* |
| ***Entrada:*** | *(1) El atributo que desea cambiar (Expresado como un número entero) y (2) El nuevo valor del atributo* |
| ***Salida:*** | *La técnica fue actualizada satisfactoriamente* |

| ***Nombre:*** | ***RF10 – Mostrar clanes*** |
|---|---|
| ***Resumen:*** | *El sistema permite mostrar todos los clanes.* |
| ***Entrada:*** | *Ninguna* |
| ***Salida:*** | *Una lista con todos los clanes* |

| ***Nombre:*** | ***RF11 – Mostrar un personaje*** |
|---|---|
| ***Resumen:*** | *El sistema permite mostrar todos los personajes de un clan, se pueden mostrar ordenados por nombre, personalidad y nivel de poder.* |
| ***Entrada:*** | *Tipo de ordenamiento* |
| ***Salida:*** | *Una lista con todos los personajes del clan* |

| ***Nombre:*** | ***RF12 – Mostrar una técnica*** |
|---|---|
| ***Resumen:*** | *El sistema permite mostrar todas las técnicas de un personaje.* |
| ***Entrada:*** | *Ninguna* |
| ***Salida:*** | *Una lista con todas las técnicas del personaje* |

## ¿Qué es la trazabilidad en el desarrollo de software?

La trazabilidad en el desarrollo de software es un registro que nos permite enlazar los requerimientos con los métodos o código que los satisface, haciendo más fácil su mantenimiento. Nos permite además calcular el impacto (en términos de código) que implica cambiar un requerimiento del programa o algún método. Es una muy buena práctica, especialmente útil en medianos o grandes proyectos, donde saber qué métodos satisfacen un requerimiento no es tan evidente, lo que nos permite ahorrar muchísimo tiempo.

# *ESPECIFICACIÓN DE REQUERIMIENTOS NO FUNCIONALES*

| ***Nombre:*** | ***RNF1 – Listas doblemente enlazadas*** |
|---|---|
| ***Resumen:*** | *Los personajes deben ser implementados usando listas doblemente enlazadas.* |

| ***Nombre:*** | ***RNF2 – Listas sencillas*** |
|---|---|
| ***Resumen:*** | *Las técnicas deben ser implementadas usando listas sencillas.* |

| ***Nombre:*** | ***RNF3 – Métodos de ordenamiento*** |
|---|---|
| ***Resumen:*** | *El programa debe implementar los 3 métodos de ordenamiento clásicos: burbuja, selección, inserción.* |

| ***Nombre:*** | ***RNF4 – Comparable*** |
|---|---|
| ***Resumen:*** | *El programa debe implementar la interfaz Comparable.* |

| ***Nombre:*** | ***RNF5 – Comparator*** |
|---|---|
| ***Resumen:*** | *El programa debe implementar la interfaz Comparator.* |

| ***Nombre:*** | ***RNF6 – Búsquedas*** |
|---|---|
| ***Resumen:*** | *El programa debe implementar algoritmos de búsqueda secuencial.* |

| *Nombre:* | *RNF7 – El programa debe persistir* |
| --- | --- |
| ***Resumen:*** | *EL programa debe restaurar su estado anterior cada vez que se ejecute.* |

# *TRAZABILIDAD*

| *RF1 – Registrar un clan* | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***NarutoGame*** | addClan() |
| ***NarutoGame*** | checkIfExistClanWithThisName() |
| ***Clan*** | getName() |

| *RF2 – Registrar un personaje* | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***Clan*** | addCharacter() |
| ***Clan*** | checkIfExistCharacterWithThisName() |
| ***Clan*** | addFirst() |
| ***Clan*** | getCharacter() |
| ***Clan*** | getFirst() |
| ***Clan*** | getSize() |
| ***Clan*** | getNext() |
| ***Clan*** | setSize() |
| ***Clan*** | setPrev() |

| | |
|---|---|
| ***Clan*** | setNext() |
| ***Clan*** | setFirst() |
| ***GameCharacter*** | getName() |

| ***RF3 – Registrar una técnica*** | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***GameCharacter*** | addTechnique() |
| ***GameCharacter*** | checkIfExistTechniqueWithThisName() |
| ***GameCharacter*** | addFirst() |
| ***GameCharacter*** | sortTechniquesByFactor() |
| ***GameCharacter*** | getFirst() |
| ***GameCharacter*** | setFirst() |
| ***GameCharacter*** | getTechnique() |
| ***Technique*** | getName() |
| ***GameCharacter*** | getSize() |
| ***Technique*** | compareTo() |
| ***GameCharacter*** | setTechnique() |

| *RF4 – Eliminar un clan* | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***NarutoGame*** | deleteClan() |
| ***Clan*** | getName() |

| *RF5 – Eliminar un personaje* | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***Clan*** | deleteCharacter() |
| ***Clan*** | getFirst() |
| ***GameCharacter*** | getName() |
| ***Clan*** | setFirst() |
| ***GameCharacter*** | getNext() |
| ***GameCharacter*** | getPrev() |
| ***GameCharacter*** | setPrev() |

| *RF6 – Eliminar una técnica* | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***GameCharacter*** | deleteTechnique() |

| | |
|---|---|
| ***Technique*** | getName() |
| ***Terchnique*** | getNext() |
| ***GameCharacter*** | setFirst() |
| ***GameCharacter*** | getTechnique() |

| ***RF7 – Actualizar un clan*** | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***Clan*** | setName() |

| ***RF8 – Actualizar un personaje*** | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***Clan*** | updateCharacter() |
| ***GameCharacter*** | setName() |
| ***GameCharacter*** | setPersonality() |
| ***GameCharacter*** | setCreationDate() |
| ***GameCharacter*** | setPowerLevel |
| ***Clan*** | sortCharactersByName() |
| ***GameCharacter*** | compareTo() |

| | |
|---|---|
| ***Clan*** | getCharacter() |
| ***Clan*** | setCharacter() |

| ***RF9 – Actualizar una técnica*** | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***GameCharacter*** | updateTechnique() |
| ***Technique*** | setName() |
| ***Technique*** | setFactor() |
| ***GameCharacter*** | sortTechniquesByFactor() |
| ***Technique*** | compareTo() |
| ***GameCharacter*** | getTechnique() |
| ***GameCharacter*** | setTechnique |

| ***RF10 – Mostrar un clan*** | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***NarutoGame*** | printClans() |
| ***Clan*** | toString() |

| *RF11 – Mostrar un personaje* | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***Clan*** | printCharacters() |
| ***GameCharacter*** | toString() |
| ***Clan*** | getSize() |
| ***Clan*** | getCharacter() |
| ***Clan*** | sortCharacterByName() |
| ***Clan*** | sortCharactersByPersonality() |
| ***Clan*** | sortCharactersByPowerLevel() |
| ***Clan*** | setCharacterByName() |
| ***GameCharacter*** | compareTo() |
| ***GameCharacter*** | compare() |
| ***GameCharacter*** | compareByPowerLevel() |

| *RF12 – Mostrar una técnica* | |
|---|---|
| ***CLASE*** | ***MÉTODO*** |
| ***GameCharacter*** | showTechniques() |
| ***GameCharacter*** | getSize() |

| | |
|---|---|
| ***GameCharacter*** | getTechnique() |
| ***Technique*** | toString() |

ui

**Menu**

- scanner : Scanner

+ Menu()
+ systemMenu(): void
+ systemOptionMenu(): int
+ registerClanMenu(): void
+ deleteClanMenu(): void
+ enterInAClanMenu(): void
+ clanMenu(): void
+ clanOptionMenu(): int
+ registerCharacterMenu(): void
+ validateInt(): int
+ validateDouble(): double
+ deleteCharacterMenu(): void
+ showCharactersMenu(): void
+ updateClan(): void
+ enterInAClanMenu(): void
+ characterMenu(): void
+ characterOptionMenu(): int
+ registerTechniqueMenu(): void
+ deleteTechnniqueMenu(): void
+ updateCharacterMenu(): void
+ enterInATechniqueMenu(): void
+ techniqueMenu(): void
+ techniqueOptionMenu(): int
+ updateTechniqueMenu(): void

**Main**

+ main(String[] arg): void

<<use>>

- currentClan

model

- game

<<use>>

**<<Exception>>**
**IllegalNameException**

IllegalNameException(msg : String)

<<use>>

<<use>>

- currentCharacter

- currentTechnique

0..1

<<use>>

0..1

0..1

1

**NarutoGame**

+ NarutoGame()
+ addClan(name : String): boolean
+ checkIfExistClanWithThisName(String : name): boolean
+ saveClans(): void
+ loadClans(): void
+ printClans(): String
+ deleteClan(): boolean
+ getClan(): Clan
+ getClans(): ArrayList<Clan>
+ setClans(clans : ArrayList<Clan>): void

- clans

0..*

**Clan**

- name : String

+ Clan(name : String)
+ getName(): String
+ getFirst(): GameCharacter
+ setName(): void
+ setFirst(): void
+ getSize(): int
+ addCharacter(name : String, personality : String, creationDate : String, powerLevel : int): boolean
+ checkIfExistCharacterWithThisName(name : String) : boolean
+ addFirst(character : GameCharacter): void
+ getCharacter(index : int): GameCharacter
+ setCharacter(index : int, character : GameCharacter): void
+ sortCharactersByName(): void
+ deleteCharacter(name : String): boolean
+ toString(): String
+ showCharacters(type : int): String
+ printCharacters(): String
+ getCharacterByName(): GameCharacter
+ updateCharacter(name : String, character : GameCharacter): boolean
+ getTechniqueByName(name : String): Technique

- firstCharacter

0..1

0..1

**GameCharacter**

- name : String
- personality : String
- creationDate : String
- powerLevel : int

+ GameCharacter(name : String, personality : String, creationDate : String, powerLevel : int)
+ getName(): String
+ getPersonality(): String
+ getCreationDate(): String
+ getPowerLevel(): int
+ getFirst(): Technique
+ setName(name : String): void
+ setPersonality(personality : String): void
+ setCreationDate(creationDate : String): void
+ setPowerLevel(powerLevel : int): void
+ setFirst(firstTechnique : Technique): void
+ compareTo(character : GameCharacter): int
+ compare(character : GameCharacter, character2 : GameCharacter): int
+ compareByPowerLevel(character : GameCharacter): int
+ toString(): String
+ getSize(): int
+ getTechnique(index : int): Technique
+ setTechnique(index : int): void
+ checkIfExistTechniqueWithThisName(name : String) : boolean
+ addTechnique(name : String, factor : double): boolean
+ addFrist(technique : Technique): void
+ showTechniques() : String
+ sortTechniquesByFactor(): void
+ updateTechnique(type : int, arg : String, technique : Technique): void
+ deleteTechnique(name : String): boolean

- prev

0..1

- next

0..1

- firstTechnique

0..1

0..1

**Technique**

- name : String
- factor : double

+ Technique(name: String, factor : double)
+ toString(): String
+ compareTo(technique : Technique): int
+ updateTechnique(type : int, arg : String)

0..1

- next

# CASOS DE PRUEBA

| Caso de prueba 1 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Clan | setCharacter() | 3 personajes (en una lista doblemente enlazada) cuyas posiciones (en la lista doblemente enlazada) intercambiaré | Las nuevas posiciones de los personajes | Verdadero, los personajes cambian a las posiciones esperadas. |

| Caso de prueba 2 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Clan | sortCharactersByName() | 6 personajes (en una lista doblemente enlazada) que sólo tiene nombre y están desordenados | Ninguna | Verdadero, los personajes fueron ordenados de menor a mayor por el nombre |

| Caso de prueba 3 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |

| | | | | |
|---|---|---|---|---|
| Clan | deleteCharacter() | 6 personajes (en una lista doblemente enlazada) que eliminaré de la lista doblemente enlazada. | El nombre del personaje que será eliminado | Verdadero, el personaje fue eliminado de la lista doblemente enlazada |

| **Caso de prueba 4** | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Clan | getCharacterByName() | 3 personajes (en una lista doblemente enlazada), con los cuales verificaré que se obtenga el clan con el nombre | El nombre del clan que se quiere obtener | Verdadero, el clan que se obtiene es el esperado |

| **Caso de prueba 5** | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| NarutoGame | addClan() | Un ArrayList vacío de clanes | Los nombres de los clanes que serán agregados | Verdadero, los clanes fueron agregados al ArrayList |

| **Caso de prueba 6** | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |

| | | | | |
|---|---|---|---|---|
| NarutoGame | deleteClan() | Un ArrayList con 4 clanes | Los nombres de los clanes que desea eliminar | Verdadero, los clanes fueron eliminados del ArrayList |

| Caso de prueba 7 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Clan | addCharacter() | Un clan con su primer personaje null | Los clanes que se van a agregar (sus nombres) | Verdadero, los clanes se han agregado a la lista enlazada |

| Caso de prueba 8 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Clan | sortCharactersByPersonality() | 6 personajes (en una lista doblemente enlazada) que sólo tiene personalidad y están desordenados | Ninguna | Verdadero, los personajes están ordenados de menor a menor por la personalidad |

| Caso de prueba 9 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| Clan | sortCharactersByPowerLevel() | 6 personajes (en una lista doblemente enlazada) que sólo | Ninguna | Verdadero, los personajes están ordenados de |

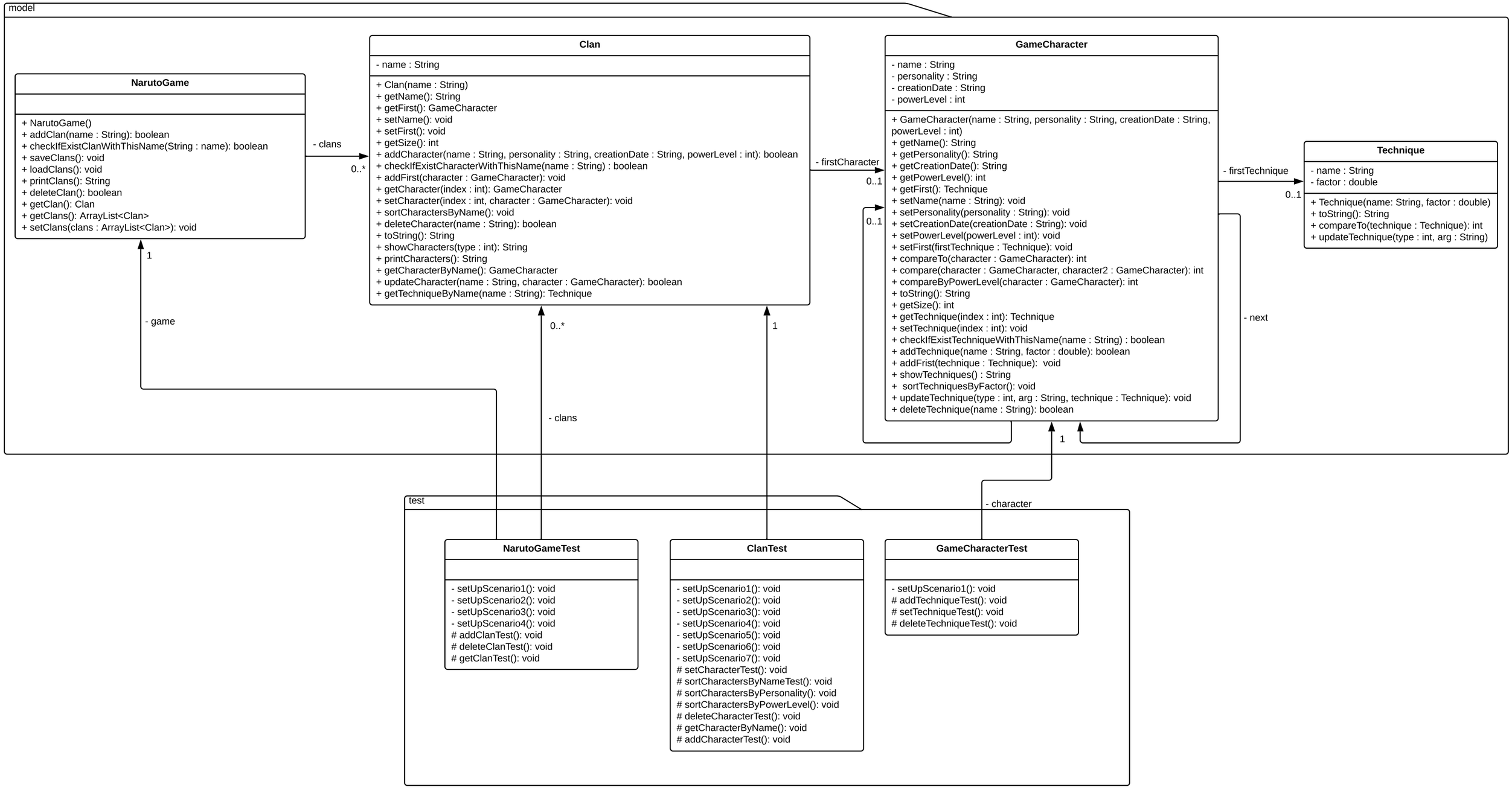
| | | | | |
|---|---|---|---|---|
| | | tiene nivel de poder y están desordenados | | menor a mayor por el nivel de poder |

| Caso de prueba 10 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| GameCharacter | addTechnique() | 6 técnicas | Las técnicas que se van a agregar (su nombre y su factor) | Verdadero, las técnicas fueron agregadas a la lista enlazada y están ordenadas de menor a mayor por el factor |

| Caso de prueba 11 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |
| GameCharacter | setTechnique() | 6 técnicas cuyas posiciones (En la lista enlazada) intercambiaré | Las técnicas que se van a intercambiar y sus nuevas posiciones | Verdadero, las técnicas cambian a las posiciones esperadas. |

| Caso de prueba 12 | | | | |
|---|---|---|---|---|
| **Clase** | **Método** | **Escenario** | **Valores de entrada** | **Resultado** |

| | | | | |
|---|---|---|---|---|
| GameCharacter | deleteTechnique() | 6 técnicas que eliminaré (de la lista enlazada) | El nombre de la técnica que se desea eliminar | Verdadero, las técnicas fueron eliminadas |


model
NarutoGame
+ NarutoGame()
+ addClan(name : String): boolean
+ checkIfExistClanWithThisName(String : name): boolean
+ saveClans(): void
+ loadClans(): void
+ printClans(): String
+ deleteClan(): boolean
+ getClan(): Clan
+ getClans(): ArrayList<Clan>
+ setClans(clans : ArrayList<Clan>): void
- clans
0..*
Clan
- name : String
+ Clan(name : String)
+ getName(): String
+ getFirst(): GameCharacter
+ setName(): void
+ setFirst(): void
+ getSize(): int
+ addCharacter(name : String, personality : String, creationDate : String, powerLevel : int): boolean
+ checkIfExistCharacterWithThisName(name : String) : boolean
+ addFirst(character : GameCharacter): void
+ getCharacter(index : int): GameCharacter
+ setCharacter(index : int, character : GameCharacter): void
+ sortCharactersByName(): void
+ deleteCharacter(name : String): boolean
+ toString(): String
+ showCharacters(type : int): String
+ printCharacters(): String
+ getCharacterByName(): GameCharacter
+ updateCharacter(name : String, character : GameCharacter): boolean
+ getTechniqueByName(name : String): Technique
- firstCharacter
0..1
GameCharacter
- name : String
- personality : String
- creationDate : String
- powerLevel : int
+ GameCharacter(name : String, personality : String, creationDate : String, powerLevel : int)
+ getName(): String
+ getPersonality(): String
+ getCreationDate(): String
+ getPowerLevel(): int
+ getFirst(): Technique
+ setName(name : String): void
+ setPersonality(personality : String): void
+ setCreationDate(creationDate : String): void
+ setPowerLevel(powerLevel : int): void
+ setFirst(firstTechnique : Technique): void
+ compareTo(character : GameCharacter): int
+ compare(character : GameCharacter, character2 : GameCharacter): int
+ compareByPowerLevel(character : GameCharacter): int
+ toString(): String
+ getSize(): int
+ getTechnique(index : int): Technique
+ setTechnique(index : int): void
+ checkIfExistTechniqueWithThisName(name : String) : boolean
+ addTechnique(name : String, factor : double): boolean
+ addFrist(technique : Technique): void
+ showTechniques() : String
+ sortTechniquesByFactor(): void
+ updateTechnique(type : int, arg : String, technique : Technique): void
+ deleteTechnique(name : String): boolean
- firstTechnique
0..1
Technique
- name : String
- factor : double
+ Technique(name: String, factor : double)
+ toString(): String
+ compareTo(technique : Technique): int
+ updateTechnique(type : int, arg : String)
1
- game
0..*
- clans
1
- next
1
- character
test
NarutoGameTest
- setUpScenario1(): void
- setUpScenario2(): void
- setUpScenario3(): void
- setUpScenario4(): void
# addClanTest(): void
# deleteClanTest(): void
# getClanTest(): void
ClanTest
- setUpScenario1(): void
- setUpScenario2(): void
- setUpScenario3(): void
- setUpScenario4(): void
- setUpScenario5(): void
- setUpScenario6(): void
- setUpScenario7(): void
# setCharacterTest(): void
# sortCharactersByNameTest(): void
# sortCharactersByPersonality(): void
# sortCharactersByPowerLevel(): void
# deleteCharacterTest(): void
# getCharacterByName(): void
# addCharacterTest(): void
GameCharacterTest
- setUpScenario1(): void
# addTechniqueTest(): void
# setTechniqueTest(): void
# deleteTechniqueTest(): void