**Objetivos**

**Unidad 3: Estructuras Lineales Enlazadas**

Al finalizar esta unidad, el estudiante estará en capacidad de:

OE5.1 Utilizar estructuras enlazadas de objetos para modelar grupos de atributos no primitivos de tamaño flexible.

OE5.2 Escribir los algoritmos necesarios para manipular estructuras lineales que almacenan sus elementos enlazándolos entre ellos.

**Preparación**

* Lea cuidadosamente el enunciado, la documentación suministrada y cada uno de los puntos que debe desarrollar antes de empezar su desarrollo. Pregunte a su profesor cualquier duda respecto al enunciado o a los requerimientos funcionales que debe desarrollar.
* El trabajo debe ser realizado individualmente.
* El trabajo será entregado en la fecha y hora establecida en Moodle.
* Usted debe utilizar git para manejar el versionamiento de su trabajo desarrollado localmente y manejar como repositorio remoto **privado**, un repositorio en GitHub o un proyecto GitLab. Su trabajo debe ser gestionado con git desde el inicio del desarrollo, los commit deben hacerse regularmente, así como los push al remoto. Esto se verificará con las fechas de los commits. En el momento de la fecha de entrega máxima, usted debe hacer público su repositorio. Recuerde las convenciones de nombre y estructura de directorios indicadas en esta [presentación de Git](https://drive.google.com/open?id=1RmergYAoyEvs-Q8g6gklipoKqeHyRhWNW8N77ewTQDQ).

**Enunciado**

Naruto es un manga escrito e ilustrado por Masashi Kishimoto y su adaptación al anime es dirigida por Hayato Date. La obra narra la historia de un ninja adolescente hiperactivo, posteriormente adulto llamado Naruto Uzumaki, quien aspira a convertirse en Hokage, con el propósito de ser reconocido como alguien importante dentro de su aldea y entre sus amigos.

La universidad tiene un proyecto de crear un videojuego y para ello le ha solicitado a usted que le ayude a construir algunas de las funcionalidades. Para esto es importante saber que todos los personaje de la Naruto shippuden tiene un puntaje para determinar su poder, por ejemplo: Naruto: 899, Naruto sennin: 1200, Naruto kyubi mode: 3.000m, Naruto sabio de los 6 caminos: 4.200, Naruto sabio de los 6 caminos Fusionado con kurama:14.500, Sasuke: 900, Sasuke susanno: 1.900, Sasuke sharingan eternal: 2.300, Kakachi: 1.400, Yiraiya: 3.000, kiba: 740, akamaru: 230, konohamaru: 101, tsunade: 890, Neji: 830, etc.

Del juego se desea tener clanes, por cada clan se tienen los personajes y de ellos se sabe su nombre, su personalidad, la fecha de creación del personaje, su poder y las técnicas usadas para este personaje. De las técnicas nos interesa saber el nombre de la técnica y el factor que influye en el poder de cada personaje, también es importante tener en orden ascendentemente todos las técnicas de un personaje por el factor. Se debe tener en cuenta que el nombre del clan, del personaje y de la técnica no se repitan. El inversionista del proyecto ha solicitado que los personajes sean implementados usando una lista doblemente enlazada y las técnicas una lista sencilla.

En su programa usted deberá:

1. Realizar el CRUD de cada una de las clases del sistema (del original en inglés: Create, Read, Update and Delete) y realizar la respectiva persistencia.
2. Implementar y utilizar los tres métodos de ordenamiento clásicos: burbuja, selección e inserción.
3. Implementar y utilizar las dos estrategias de búsqueda clásicas (secuencial).
4. Utilizar la interface Comparable.
5. Utilizar la interface Comparator.

Ya que en el programa hay que hacer diferentes ordenamientos y búsquedas, para cumplir con los requisitos anteriores, usted utilizará un algoritmo de ordenamiento y búsqueda en un caso y otro algoritmo ordenamiento búsqueda en otro caso, implementando todos los vistos hasta la fecha.

**Entregables.**

**1.** Requerimientos Funcionales y no funcionales.

**2.** Diagrama de clases de modelo y de la interfaz del usuario (no generado automáticamente).

**3.** Implementación completa de todos los requerimientos en Java.

**4.** Tabla de trazabilidad de requerimientos vs métodos (tabla con una columna de los requerimientos, tal que, por cada requerimiento se indica en la columna siguiente todos los métodos que contribuyen a resolverlo).

**5.** Investigue e incluya en su reporte de laboratorio qué es trazabilidad en desarrollo de software.

**Fecha de Entrega Máxima:** martes 24 de septiembre de 2019 a las 11:59 AM a través de Moodle. El laboratorio debe trabajarse y entregarse **individualmente**. Lo que usted debe entregar de su trabajo es la url de su repositorio en GitHub o proyecto en GitLab. Recuerde que el repositorio o proyecto debe ser privado durante el desarrollo del laboratorio y hacerse público solo en el momento justo de la entrega máxima indicada aquí.

**DESARROLLO**

**1.**

|  |  |
| --- | --- |
| **Nombre** | **R.F. 1. Registrar un clan nuevo** |
| **Resumen** | El usuario debe ingresar el nombre y fecha de apertura del clan, para que registrar los datos en el sistema. |
| **Entradas** | |
| -Nombre del clan  -Fecha de apertura | |
| **Resultados** | |
| Clan registrado. | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 2. Registrar un ninja nuevo** |
| **Resumen** | El usuario debe ingresar el nombre, fecha de creación, personalidad y poder del ninja, para que registrar los datos en el sistema. |
| **Entradas** | |
| -Nombre del clan  -Nombre del ninja  -Fecha de creación del ninja  -Personalidad del ninja  -Poder del ninja | |
| **Resultados** | |
| Ninja registrado. | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 3. Registrar un jutsu nueva** |
| **Resumen** | El usuario debe ingresar el nombre, genero y fecha de nacimiento de la mascota, para que registrar los datos en el sistema. |
| **Entradas** | |
| -ID del club del dueño  -ID del dueño  -ID de la mascota  -Nombre de la mascota  -Genero de la mascota  -Tipo de mascota  -Fecha de nacimiento de la mascota | |
| **Resultados** | |
| Mascota registrada. | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 4. Eliminar un club** |
| **Resumen** | El usuario debe ingresar el ID o nombre del club para que el programa lo busque y lo elimine consigo a los clientes y mascotas que tenga ligados. |
| **Entradas** | |
| -ID del club  -Nombre del club | |
| **Resultados** | |
| Club eliminado. | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 5. Eliminar un cliente** |
| **Resumen** | El usuario debe ingresar el ID o nombre de un cliente para que el programa lo busque y lo elimine consigo a las mascotas que tenga ligados. |
| **Entradas** | |
| -ID del cliente  -Nombre del cliente | |
| **Resultados** | |
| Cliente eliminado. | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 6. Eliminar una mascota** |
| **Resumen** | El usuario debe ingresar el ID o nombre de una mascota para que el programa lo busque y lo elimine |
| **Entradas** | |
| -ID de la mascota  -Nombre de la mascota | |
| **Resultados** | |
| Mascota eliminado. | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 7. Leer archivos** |
| **Resumen** | El programa debe poder leer archivos de texto plano y serializados para que su contenido pueda ser utilizado dentro del problema |
| **Entradas** | |
| -Archivos de texto plano  -Archivos serializados | |
| **Resultados** | |
| Contenido de archivos cargado en el sistema | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 8. Guardar cambios en el programa** |
| **Resumen** | El programa debe poder guardar cambios del modelo del problema para que estos datos puedan ser utilizados de nuevo en el futuro |
| **Entradas** | |
| -Archivos de texto plano  -Archivos serializados | |
| **Resultados** | |
| Estado final del sistema es guardado en archivos | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 9. Buscar un club** |
| **Resumen** | El programa debe poder buscar un club según las entradas dadas por el usuario y bajo que críterio quiere hacer la búsqueda |
| **Entradas** | |
| -ID del club  -Nombre del club  -Fecha de apertura | |
| **Resultados** | |
| Club encontrado | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 10. Buscar un cliente** |
| **Resumen** | El programa debe poder buscar un cliente de un club según las entradas dadas por el usuario y bajo que críterio quiere hacer la búsqueda |
| **Entradas** | |
| -ID del club del cliente  -ID del cliente  -Nombre del cliente  -Apellidos del cliente  -Fecha de nacimiento del cliente  -Tipo de mascota favorita del cliente | |
| **Resultados** | |
| Cliente encontrado | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 11. Buscar una mascota** |
| **Resumen** | El programa debe poder buscar una mascota de un cliente de un club según las entradas dadas por el usuario y bajo que críterio quiere hacer la búsqueda |
| **Entradas** | |
| -ID del club del cliente  -ID del cliente  -ID de la mascota  -Nombre de la mascota  -Genero de la mascota  -Tipo de mascota  -Fecha de nacimiento de la mascota | |
| **Resultados** | |
| Mascota encontrada | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 12. Ordenar clubes** |
| **Resumen** | El programa debe poder ordenar los clubles bajo cualquiera de los críterios posibles |
| **Entradas** | |
| -Selección del críterio de ordenamiento | |
| **Resultados** | |
| Clubes ordenados bajo algún críterio | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 13. Ordenar clientes** |
| **Resumen** | El programa debe poder ordenar los clientes de un club bajo cualquiera de los críterios posibles |
| **Entradas** | |
| -Selección del críterio de ordenamiento | |
| **Resultados** | |
| Clientes de un club ordenados bajo algún críterio | |

|  |  |
| --- | --- |
| **Nombre** | **R.F. 14. Ordenar mascotas** |
| **Resumen** | El programa debe poder ordenar las mascotas de una persona bajo cualquiera de los críterios posibles |
| **Entradas** | |
| -Selección del críterio de ordenamiento | |
| **Resultados** | |
| Mascotas de un cliente ordenados bajo algún críterio | |

|  |  |
| --- | --- |
| **Nombre** | **R.N.F.1 Ninjas únicos en cada clan del programa** |
| **Resumen** | **Por cada clan debe haber ninjas con nombres únicos** |

|  |  |
| --- | --- |
| **Nombre** | **R.N.F.2 Mascotas con el mismo nombre o ID en una persona** |
| **Resumen** | **No podrán haber mascotas con el mismo nombre ni ID registradas bajo una persona** |