**Algoritmos y programación ll**

**Jhonatan Steven Arboleda Velez - A00358993**

**Laboratorio 2.**

Requerimientos funcionales

|  |  |
| --- | --- |
| Nombre | R1. Buscar clubes, dueños y mascotas. |
| Resumen | Se requiere que el programa sea capaz de buscar los clubes, dueños y mascotas por cualquier criterio que el usuario escoja, de la manera tradicional y binaria, además de mostrarle al usuario el tiempo que se tardó en buscar en cada caso. |
| Entradas | Atributo por el que se quiere buscar el dueño, club o mascota. |
| Salida | Un mensaje diciendo si el elemento existe o no. |

|  |  |
| --- | --- |
| Nombre | R2. Ordenar clubes, dueños y mascotas. |
| Resumen | Se requiere que el programa sea capaz de ordenar los clubes, dueños y mascotas por cualquier criterio que el usuario escoja. |
| Entradas | <Ninguna> |
| Salida | Listado ordenado de los clubes, dueños o mascotas. |

|  |  |
| --- | --- |
| Nombre | R3. Agregar clubes, dueños y mascotas. |
| Resumen | Se requiere que el programa añada clubes, dueños y mascotas al sistema. |
| Entradas | Atributos del elemento por agregar |
| Salida | Ninguna |

|  |  |
| --- | --- |
| Nombre | R4. Eliminar clubes, dueños y mascotas. |
| Resumen | Se requiere que el programa elimine los clubes, dueños o mascotas. |
| Entradas | Identificación o nombre del elemento a eliminar. |
| Salida | Un mensaje diciendo si fue posible eliminarlo o no. |

Requerimientos no funcionales

1. No deben existir dos dueños con la misma identificación en el programa.
2. Un dueño no puede tener dos mascotas con el mismo nombre.
3. Crear y utilizar los tres métodos de ordenamiento.
4. Utilizar las interfaces Comparable y Comparator.
5. Guardar la información de los clubes en archivos planos y la información de los dueños y sus mascotas en archivos serializados.

Trazabilidad

|  |  |  |
| --- | --- | --- |
| Requerimiento | Clase | Método |
| R1. Buscar clubes, dueños y mascotas. | Sponsor | +tradSearch\*(String):void  +binSearch\*(String):void |
| +tradSearch\*Owner(String): void  +binSearch\*Owner(String): void |
| +tradSearch\*Pet(\*): void  +binSearch\*Pet(\*): void |
| Club | +tradSearch\*(String): void  +binSearch\*(String): void |
| +tradSearch\*Pet(\*): void  +binSearch\*Pet(\*): void |
| Owner | +tradSearch\*(\*): void  +binSearch\*(\*): void |
| R2. Ordenar clubes, dueños y mascotas. | Sponsor | +orderClubs(): void  +orderClubs\*(): void |
| +orderOwner\*(): void |
| +orderPet\*(): void |
| Club | +orderOwners(): void  +orderOwner\*(): void |
| +orderPet\*(): void |
| Owner | +orderPet\*(): void |
| R3. Agregar clubes, dueños y mascotas. | Sponsor | +addClub(Club): void |
| +addOwner(Owner,String): void |
| +addPet(Pet,String): void |
| Club | +addOwner(Owner): void  +sameOwner(Owner): void  +sameOwnerId(String): void |
| +addPet(Pet,String): void |
| Owner | +addPet(Pet): void |
| R4. Eliminar clubes, dueños y mascotas. | Sponsor | +deleteClub(String): void |
| +deleteOwner(String): void |
| +deletePet(String): void |
| Club | +eliminateOwner(String): void |
| +eliminatePet(String): void |
| Owner | +eliminatePet(String): void |