Diseño de pruebas.

Clase: *ClubAdministration.*

- **Prueba**: El método permite registrar agregar un club a la lista de clubes.

Clase	Método probado	Escenario	Valores de entrada	Resultado
ClubAdministration	registerClubs(String id, String clubName, String dateCreation, String petType)	Se crea una nueva clase con una lista de clubes los cuales son cargados desde una archivo .txt llamado “ClubesTest”.	-La id del Club -El nombre del Club -La fecha de creación del Club -El tipo de Mascotas que maneja el Club.	Verdadero: Un nuevo club se debió añadir a la lista de clubes.

- **Prueba**: El método busca un club y si lo encuentra lo retorna

Clase	Método probado	Escenario	Valores de entrada	Resultado
ClubAdministration	searchClub ()	Se crea una nueva clase con una lista de clubes los cuales son cargados desde una archivo .txt llamado “ClubesTest”. Los archivos de serialización de mascotas y de dueños no están habilitados para las pruebas de esta clase.	No requiere entradas.	No null: El club buscado debe existir porque ya se registró.
ClubAdministration	searchClub ()	Se crea una nueva clase con una lista de clubes los cuales son cargados desde una archivo .txt llamado “ClubesTest”. Los archivos de serialización de mascotas y de dueños no están habilitados para las pruebas de esta clase.	No requiere entradas.	Null: El club buscado no debe existir puesto que nunca se agrego a la lista de clubes.

- **Prueba**: El método ordena una lista de clubes de acuerdo a su nombre.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	orderClubsByClubName ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro “clubName” en desorden.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.

ClubAdministration	orderClubsByClubName ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “clubName” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsByClubName ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “clubName” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de clubes de acuerdo a su id.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	orderClubsById ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro “id” en desorden.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsById ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “id” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsById ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “id” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de clubes de acuerdo con su fecha de creación.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	orderClubsByDate ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro “dateCreation” en desorden.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsByDate ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “dateCreation” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsByDate ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “dateCreation” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de clubes de acuerdo con su tipo de mascotas permitidas.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	orderClubsByPet ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro “petType” en desorden.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsByPet ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “petType” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsByPet ()	Se crea una nueva clase con una lista de clubes con 5 clubes, los cuales se ingresaron con su parámetro de “petType” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de clubes de acuerdo con su número de dueños.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	orderClubsBy NumberOwners ()	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les ingresaron un numero de dueños de acuerdo con el orden lexicográfico de su id, por lo cual el menor tendrá menos mascotas que el mayor. Cada lista de clubes esta en desorden siguiendo el criterio lexicografico	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsBy NumberOwners ()	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les ingresaron un numero de dueños de acuerdo con el orden lexicográfico de su id, por lo cual el menor tendrá menos mascotas que el mayor. Cada lista de clubes esta ordenada de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.
ClubAdministration	orderClubsBy NumberOwners ()	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les ingresaron un numero de dueños de acuerdo con el orden lexicográfico de su id, por lo cual el menor tendrá menos mascotas que el mayor. Cada lista de clubes esta ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de clubes que será ordenada debe coincidir con una lista de clubes diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método busca y encuentra un club por medio de su id, si la id ingresada no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	secuencialSearch ById(String id)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id del club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	secuencialSearch ById(String id)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id de un club que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un club por medio de su nombre, si el nombre ingresado no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	secuencialSearch ByClubName(String clubName)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre del club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	secuencialSearch ByClubName(String clubName)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre de un club que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un club por medio de su fecha de creación, si la fecha de creación ingresada no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	secuencialSearch ByClubDate(String date)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	La fecha de creación del club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	secuencialSearch ByClubDate(String date)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	La fecha de creación de un club que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un club por medio del tipo de mascotas que admite el club, si el tipo de mascota ingresada no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	secuencialSearch ByPet(String pet)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El tipo de mascotas permitidas por el club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	secuencialSearch ByPet(String pet)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	Un tipo de mascotas que no existe en ningún club.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un club por medio de su id, si la id ingresada no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	binarySearch ById(String id)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id del club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	binarySearch ById(String id)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id de un club que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un club por medio de su nombre, si el nombre ingresado no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	binarySearch ByClubName(String clubName)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre del club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	binarySearch ByClubName(String clubName)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre de un club que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un club por medio de su fecha de creación, si la fecha de creación ingresada no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	binarySearch ByClubDate(String date)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	La fecha de creación del club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	binarySearch ByClubDate(String date)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	La fecha de creación de un club que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un club por medio del tipo de mascotas que admite el club, si el tipo de mascota ingresada no corresponde a un club no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
ClubAdministration	binarySearch ByPet(String pet)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El tipo de mascotas permitidas por el club que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
ClubAdministration	binarySearch ByPet(String pet)	Se crea una nueva clase con una lista de clubes con 5 clubes, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	Un tipo de mascotas que no existe en ningún club.	El método no debe retornar nada

Diseño de pruebas.

Clase: *Club*

- **Prueba:** El método retorna true en caso de que encuentre un dueño con la identificación ingresada, falso en caso contrario

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	giveOwner(String identification)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	La id de un dueño que existe en la lista de dueños.	Verdadero, el dueño existe por lo cual retorna true.
Club	giveOwner(String identification)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	La id de un dueño que no existe en la lista de dueños	Falso, el dueño no existe por lo cual retorna false;

- **Prueba:** El método permite agregar un dueño a la lista de dueños

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	registerOwner((String id, String name, String lastName, String birthDate, String favoritePet)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	-La id del dueño -Su nombre -Su apellido -Su fecha de nacimiento -Su tipo de mascota favorita	Se añade un nuevo dueño al arreglo de dueños. La lista de dueños ahora tiene 2 de ellos.

- **Prueba:** El método retorna el un objeto de tipo dueño en caso de que encuentre un dueño con la identificación ingresada, null en caso contrario

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	searchOwner (String id)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	La id de un dueño que existe en la lista de dueños	No nulo, dado que el dueño existe retorna el dueño encontrado

Club	searchOwner (String id)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	La id de un dueño que NO existe en la lista de dueños	nulo, dado que el dueño NO existe retorna un objeto nulo.

- **Prueba:** El método retorna un numero positivo si la id a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	compareTo (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero negativo
Club	compareTo (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero positivo
Club	compareTo (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si el nombre del club a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales
-

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	Compare (Club o1, Club o2)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero negativo

Club	Compare (Club o1, Club o2)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero positivo
Club	Compare (Club o1, Club o2)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si la fecha de creación a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	compareByDate (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero negativo
Club	compareByDate (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero positivo
Club	compareByDate (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si el tipo de mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	compareByPet (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero negativo
Club	compareByPet (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero positivo
Club	compareByPet (Club o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si la id a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	compareByIdBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El id a comparar	Numero negativo
Club	compareByIdBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El id a comparar	Numero positivo

Club	compareByIdBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El id a comparar	Numero 0

- **Prueba:** El método retorna un numero positivo si el nombre del club a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	compareByNameBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El nombre del club a comparar.	Numero negativo
Club	compareByNameBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El nombre del club a comparar.	Numero positivo
Club	compareByNameBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El nombre del club a comparar.	Numero 0

- **Prueba:** El método retorna un numero positivo si la fecha de creación a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	compareByDateBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	La fecha de creación a comparar.	Numero negativo

Club	compareByDate (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	La fecha de creación a comparar.	Numero positivo
Club	compareByDate (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	La fecha de creación a comparar.	Numero 0

- **Prueba:** El método retorna un numero positivo si el tipo de mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Club	compareByPetBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El tipo de mascota a comparar.	Numero negativo
Club	compareByPetBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El tipo de mascota a comparar.	Numero positivo
Club	compareByPetBS (String o)	Se crea un nuevo Club con valores arbitrarios y se le añade un dueño a su lista de dueños.	El tipo de mascota a comparar.	Numero 0

- **Prueba**: El método ordena una lista de dueños de acuerdo con su nombre.

Clase	Método probado	Escenario	Entradas	Resultado
Club	orderOwnersByName ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “name” en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderOwnersByName ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “name” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderOwnersByName ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “name” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de dueños de acuerdo con su id.

Clase	Método probado	Escenario	Entradas	Resultado
Club	orderOwnersById ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “id” en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderOwnersById ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “id” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

Club	orderOwnersById ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “id” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba**: El método ordena una lista de dueños de acuerdo con su apellido.

Clase	Método probado	Escenario	Entradas	Resultado
Club	orderOwnersByLastName ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “lastName” en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderOwnersByLastName ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “lastName” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderOwnersByLastName ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “lastName” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de dueños de acuerdo con su fecha de nacimiento.

Clase	Método probado	Escenario	Entradas	Resultado
Club	orderOwnersByDate ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “birthDate” en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

Club	orderOwnersByDate ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “birthDate” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderClubsByDate ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “birthDate” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de dueños de acuerdo con su mascota favorita.

Clase	Método probado	Escenario	Entradas	Resultado
Club	orderOwnersByPet ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “favoritePet” en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderOwnersByPet ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “favoritePet” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderClubsByPet ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “favoritePet” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de clubes de acuerdo con su número de dueños.

Clase	Método probado	Escenario	Entradas	Resultado
Club	orderClubsBy NumberPets ()	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les ingresaron un numero de mascotas de acuerdo con el orden lexicográfico de su id, por lo cual el menor tendrá menos mascotas que el mayor. Cada lista de dueños esta en desorden siguiendo el criterio lexicografico	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderClubsBy NumberPets ()	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les ingresaron un numero de mascotas de acuerdo con el orden lexicográfico de su id, por lo cual el menor tendrá menos mascotas que el mayor. Cada lista de dueños esta ordenada de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Club	orderClubsBy NumberPets ()	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les ingresaron un numero de mascotas de acuerdo con el orden lexicográfico de su id, por lo cual el menor tendrá menos mascotas que el mayor. Cada lista de dueños esta ordenada de mayor a menor siguiendo el criterio lexicográfico3.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método busca y encuentra un dueño por medio de su id, si la id ingresada no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	secuencialSearch ById(String id)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id del dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.

Club	secuencialSearch ById(String id)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio de su nombre, si el nombre ingresado no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	secuencialSearch ByName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre del dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.
Club	secuencialSearch ByName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio de su apellido, si el apellido ingresado no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	secuencialSearch ByLastName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El apellido del dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.
Club	secuencialSearch ByLastName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El apellido de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio de su fecha de nacimiento, si la fecha de nacimiento ingresada no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	secuencialSearch ByOwnerDate(String date)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	La fecha de nacimiento de un dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.
Club	secuencialSearch ByOwnerDate(String date)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	Una fecha de nacimiento apellido de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio del tipo de mascotas que admite prefiere, si el tipo de mascota ingresada no corresponde a un tipo preferido por un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	secuencialSearch ByPet(String pet)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El tipo de mascota preferida de un dueño que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
Club	secuencialSearch ByPet(String pet)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	Un tipo de mascotas que no existe en ningún dueño.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio de su id, si la ingresada no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	binarySearch ById(String id)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id del dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.

Club	binarySearch ById(String id)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El id de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio de su nombre, si el nombre ingresado no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	binarySearch ByName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre del dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.
Club	binarySearch ByName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El nombre de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio de su apellido, si el apellido ingresado no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	binarySearch ByLastName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El apellido del dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.
Club	binarySearch ByLastName(String name)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El apellido de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio de su fecha de nacimiento, si la fecha de nacimiento ingresada no corresponde a un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	binarySearch ByDate(String date)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	La fecha de nacimiento de un dueño que se quiere buscar	Se debe devolver la información correspondiente al dueño buscado.
Club	binarySearch ByDate(String date)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	Una fecha de nacimiento apellido de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra un dueño por medio del tipo de mascotas que admite prefiere, si el tipo de mascota ingresada no corresponde a un tipo preferido por un dueño no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Club	binarySearch ByPet(String pet)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	El tipo de mascota preferida de un dueño que se quiere buscar	Se debe devolver la información correspondiente al club buscado.
Club	binarySearch ByPet(String pet)	Se crea una nueva clase con una lista de dueños con 3 dueños, a los cuales se les asigno diversos parámetros para realizar la búsqueda de acuerdo con lo requerido por el método.	Un tipo de mascotas que no existe en ningún dueño.	El método no debe retornar nada

Diseño de pruebas.

Clase: *Owner*

- **Prueba:** El método retorna true en caso de que encuentre una mascota con la identificación ingresada, falso en caso contrario

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	givePet(String pN)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	La id de una mascota que existe en la lista de mascotas.	Verdadero, la mascota existe por lo cual retorna true.
Owner	givePet(String pN)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	La id de una mascota que no existe en la lista de mascotas.	Falso, la mascota no existe por lo cual retorna false;

- **Prueba:** El método retorna agregar una mascota a la lista de mascotas del dueño.

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	registePet((String id, String petName, String petBirthDay, String gender, String type)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	-La id de la mascota -El nombre de la mascota -Su fecha de nacimiento -El tipo de mascota	Se añade una nueva mascota al arreglo de mascotas. La lista de mascotas ahora tiene 2 de ellas.

- **Prueba:** El método retorna un numero positivo si la id a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareTo (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	No tiene ninguna entrada	Numero negativo
Owner	compareTo (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	No tiene ninguna entrada	Numero positivo

Owner	compareTo (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si el nombre del dueño a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	Compare (Owner o1, Owner o2)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero negativo
Owner	Compare (Owner o1, Owner o2)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero positivo
Owner	Compare (Owner o1, Owner o2)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si el apellido del dueño a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	CompareByLastName (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero negativo

Owner	CompareByLastName (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero positivo
Owner	CompareByLastName (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si la fecha de nacimiento del dueño a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareByDate (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero negativo
Owner	compareByDate (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero positivo
Owner	compareByDate (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si el tipo de mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareByPet (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero negativo
Owner	compareByPet (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero positivo
Owner	compareByPet (Owner o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si la id a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareByIdBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El id a comparar	Numero negativo
Owner	compareByIdBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El id a comparar	Numero positivo

Owner	compareByIdBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El id a comparar	Numero 0

- **Prueba:** El método retorna un numero positivo si el nombre del dueño a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareByNameBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El nombre para comparar	Numero negativo
Owner	compareByNameBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El nombre para comparar	Numero positivo
Owner	compareByNameBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El nombre para comparar	Numero 0

- **Prueba:** El método retorna un numero positivo si el nombre del dueño a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareByLastNameBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El apellido para comparar	Numero negativo

Owner	compareByLastNameBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El apellido para comparar	Numero positivo
Owner	compareByLastNameBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El apellido para comparar	Numero 0

- **Prueba:** El método retorna un numero positivo si la fecha de nacimiento del dueño a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareByDateBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	La fecha de nacimiento del dueño a comparar.	Numero negativo
Owner	compareByDateBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	La fecha de nacimiento del dueño a comparar.	Numero positivo
Owner	compareByDateBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	La fecha de nacimiento del dueño a comparar.	Numero 0

- **Prueba:** El método retorna un numero positivo si el tipo de mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Owner	compareByPetBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El tipo de mascota preferida del dueño a comparar.	Numero negativo
Owner	compareByPetBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El tipo de mascota preferida del dueño a comparar.	Numero positivo
Owner	compareByPetBS (String o)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El tipo de mascota preferida del dueño a comparar.	Numero 0

- **Prueba**: El método ordena una lista de mascotas de acuerdo con su nombre.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	orderPetsByName ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro “name” en desorden.	No requiere entradas	La lista de mascotas que será ordenada debe coincidir con una lista de mascotas diferente que ya se ordenó lexicográficamente de menor a mayor.
Owner	orderPetsByName ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro “name” en desorden.	No requiere entradas	La lista de mascotas que será ordenada debe coincidir con una lista de mascotas diferente que ya se ordenó lexicográficamente de menor a mayor.

Owner	orderPetsByName ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro "name" en desorden.	No requiere entradas	La lista de mascotas que será ordenada debe coincidir con una lista de mascotas diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de mascotas de acuerdo con su id.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	orderPetsById ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro "id" en desorden.	No requiere entradas	La lista de mascotas que será ordenada debe coincidir con una lista de mascotas diferente que ya se ordenó lexicográficamente de menor a mayor.
Owner	orderPetsById ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro "id" en desorden.	No requiere entradas	La lista de mascotas que será ordenada debe coincidir con una lista de mascotas diferente que ya se ordenó lexicográficamente de menor a mayor.
Owner	orderPetsById ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro "id" en desorden.	No requiere entradas	La lista de mascotas que será ordenada debe coincidir con una lista de mascotas diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba**: El método ordena una lista de mascotas de acuerdo con su fecha de nacimiento.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	orderPetsByDate ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro "petBirthDate" en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Owner	orderPetsByDate ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente

		ingresaron con su parámetro “petBirthDate” en desorden.		que ya se ordenó lexicográficamente de menor a mayor.
Owner	orderPetsByDate ()	Se crea una nueva clase con una lista de mascotas con 3 mascotas, los cuales se ingresaron con su parámetro “petBirthDate” en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método ordena una lista de mascotas de acuerdo a su especie.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	orderPetsByType ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “petType” en desorden.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Owner	orderPetsByType ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “petType” ordenado de menor a mayor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.
Owner	orderPetsByType ()	Se crea una nueva clase con una lista de dueños con 3 dueños, los cuales se ingresaron con su parámetro “petType” ordenado de mayor a menor siguiendo el criterio lexicográfico.	No requiere entradas	La lista de dueños que será ordenada debe coincidir con una lista de dueños diferente que ya se ordenó lexicográficamente de menor a mayor.

- **Prueba:** El método busca y encuentra una mascota por medio de su id, si la id ingresada no corresponde a una mascota no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	secuencialSearch ById(String id)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El id de la mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	secuencialSearch ById(String id)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El id de una mascota que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra una mascota por medio de su nombre, si el nombre ingresado no corresponde a una mascota no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	secuencialSearch ByName(String name)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El nombre de una mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	secuencialSearch ByName(String name)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	El nombre de una mascota que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra una mascota por medio de su fecha de nacimiento, si la fecha de nacimiento ingresada no corresponde a ninguna mascota no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	secuencialSearch ByDate(String date)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas	La fecha de nacimiento de la mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	secuencialSearch ByDate(String date)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	Una fecha de nacimiento de un dueño que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra una mascota por medio de su especie, si el tipo de mascota ingresada no corresponde a ninguna mascota registrada no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	secuencialSearch ByPet(String pet)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	El tipo de mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	secuencialSearch ByPet(String pet)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	Un tipo de mascotas que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra una mascota por medio de su id, si la id ingresada no corresponde a una mascota no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	binarySearch ById(String id)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	El id de la mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	binarySearch ById(String id)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	El id de una mascota que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra una mascota por medio de su nombre, si el nombre ingresado no corresponde a ninguna mascota no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	binarySearch ByName(String name)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	El nombre de la mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	binarySearch ByName(String name)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	El nombre de una mascota que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra una mascota por medio de su fecha de nacimiento, si la fecha de nacimiento ingresada no corresponde a una mascota registrada no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	binarySearch ByDate(String date)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	La fecha de nacimiento de la mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	binarySearch ByDate(String date)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	Una fecha de nacimiento que no existe.	El método no debe retornar nada

- **Prueba:** El método busca y encuentra una mascota por medio de su especie, si el tipo de mascota ingresada no corresponde a ninguna mascota registrada no retorna nada.

Clase	Método probado	Escenario	Entradas	Resultado
Owner	binarySearch ByPet(String pet)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	El tipo de mascota que se quiere buscar	Se debe devolver la información correspondiente a la mascota buscada.
Owner	binarySearch ByPet(String pet)	Se crea un nuevo Dueño con valores arbitrarios y se le añade una mascota a su lista de mascotas.	Un tipo de mascotas que no existe.	El método no debe retornar nada

Diseño de pruebas.

Clase: *Pet*

- **Prueba:** El método retorna un numero positivo si la id a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	compareTo (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero negativo

Pet	compareTo (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero positivo
Pet	compareTo (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si el nombre de la mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	Compare (Pet o1, Pet o2)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero negativo
Pet	Compare (Pet o1, Pet o2)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero positivo
Pet	Compare (Pet o1, Pet o2)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si la fecha de nacimiento de la mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	compareByDate (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero negativo
Pet	compareByDate (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero positivo
Pet	compareByDate (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si el tipo de mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales.

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	compareByType (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero negativo
Pet	compareByType (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero positivo
Pet	compareByType (Pet o)	Se crea una nueva mascota con valores arbitrarios.	No tiene ninguna entrada	Numero 0

- **Prueba:** El método retorna un numero positivo si la id a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	compareByIdBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El id a comparar	Numero negativo
Pet	compareByIdBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El id a comparar	Numero positivo
Pet	compareByIdBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El id a comparar	Numero 0

- **Prueba:** El método retorna un numero positivo si el nombre del dueño a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	compareByNameBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El nombre para comparar	Numero negativo
Pet	compareByNameBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El nombre para comparar	Numero positivo

Pet	compareByNameBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El nombre para comparar	Numero 0

- **Prueba:** El método retorna un numero positivo si la fecha de nacimiento de la mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	compareByDateBS (String o)	Se crea una nueva mascota con valores arbitrarios.	La fecha de nacimiento de la mascota a comparar.	Numero negativo
Pet	compareByDateBS (String o)	Se crea una nueva mascota con valores arbitrarios.	La fecha de nacimiento de la mascota a comparar.	Numero positivo
Pet	compareByDateBS (String o)	Se crea una nueva mascota con valores arbitrarios.	La fecha de nacimiento de la mascota a comparar.	Numero 0

- **Prueba:** El método retorna un numero positivo si el tipo de mascota a comparar es lexicográficamente mayor, negativo si la cadena es menor y 0 si son iguales

Clase	Método probado	Escenario	Valores de entrada	Resultado
Pet	compareByPetBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El tipo de mascota a comparar.	Numero negativo
Pet	compareByPetBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El tipo de mascota a comparar.	Numero positivo
Pet	compareByPetBS (String o)	Se crea una nueva mascota con valores arbitrarios.	El tipo de mascota a comparar.	Numero 0