

**ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL**

|  |
| --- |
| **FACULTAD DE INGENIERIA EN ELECTRICIDAD Y COMPUTACION** |

**PROGRAMACIÓN ORIENTADA A OBJETOS**

**INFORME**

“Proyecto I - Primer Parcial”

**INTEGRANTES:**

Benalcázar Rodríguez Jeanpiere Santiago

Moran Zavala Justo Bautista

López Matute Felix Xavier

**GRUPO:**

7

**CARRERA:**

Ingeniería Mecatrónica

**PROFESOR(A):**

Gladys Eliana Carrillo Bastidas

**PARALELO:** 1

**TÉRMINO:** 2021 - 2S

**GUAYAQUIL – ECUADOR**

Tabla de Índice

[1 Introducción 3](#_Toc76418747)

1.1 Objetivo General………….. ……………………………………………………3

1.2 Objetivos Específico …………………………….. …………………………….3

[2 Marco Teórico 3](#_Toc76418748)

[3 Contenido 4](#_Toc76418749)

[4 Referencias 12](#_Toc76418753)

[5 Anexos 13](#_Toc76418756)

1. **Introducción**

(En esta parte ira el objetivo del Sistema y un poco de los antecedentes. Una breve descripción de que se desarrolló.)

La Fundación Amigos Peludos EC, desea un sistema que le permita administrar los concursos de mascotas (solo perros y gatos) que se realizan en diferentes ciudades del país.

En este proyecto crearemos un programa que simule el manejo del concurso, que haga el razonamiento de la data obtenida por el árbitro y crear condiciones para procesar la información.

* 1. **Objetivo General**
* Desarrollar una programación que simule la administración de los concursos de mascotas para diferentes ciudades del país.

**1.2. Objetivos Específicos**

* Realizar paquetes adecuados para agrupar clases y el uso necesario de herencia y asociación.
* Subir toda documentación y programación de proyecto en un repositorio en la web junto con la asesoría del docente del proyecto.
* Verificar el cumplimiento de toda la funcionalidad requerida, así como las correctas validaciones de los datos.

1. **Marco Teórico**

**Interfaz de Usuario:** Es la parte del programa que permite al usuario interaccionar con él.

**Clases:** “Son la base de la Programación Orientada a Objetos. Una clase es una plantilla que define la forma de un objeto; en ella se agrupan datos y métodos que operarán sobre esos datos.” (Ing. Palacios, 2015).

**Diagrama de Clase:** “Es una representación gráfica de una clase, en el que se especifica el nombre de la clase, sus atributos y métodos.” (Fundamentos de la Programación Orientada a Objetos, 2007)

**Objeto:** “Corresponden a cosas que se encuentran en el mundo real.” (Tokio School, 2020)

**Paquetes:** “Son el mecanismo que usa Java para facilitar la modularidad del código. Un paquete puede contener una o más definiciones de interfaces y clases, distribuyéndose habitualmente como un archivo.” (Charte, 2019)

**Herencia:** “Es un mecanismo que permite la definición de una clase a partir de la definición de otra ya existente. La herencia permite compartir automáticamente métodos y datos entre clases, subclases y objetos.” (Carlos, 2016)

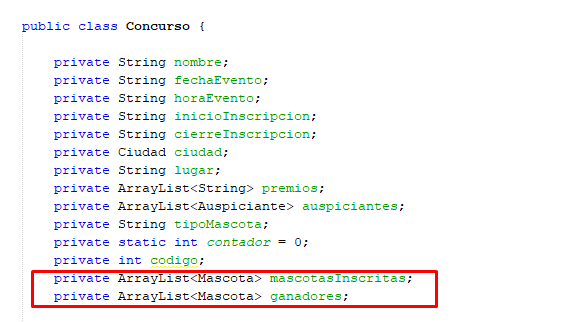
**Asociación:** “Expresa una relación (unidireccional o bidireccional) entre las instancias a partir de las clases conectadas. El sentido en que se recorre la asociación se denomina navegabilidad de la asociación.” (Garro, 2015)

**Instancia:** “Instanciar objetos es el proceso de generar un ejemplar de una clase, es decir, la clase es como una declaración de una forma y el objeto es un caso o elemento concreto que responde a esa forma.” (Ángel, 2008)

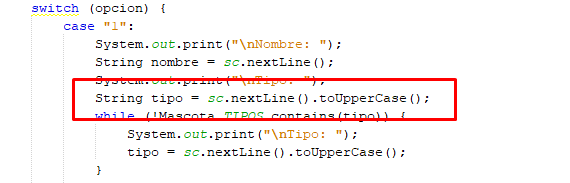
**Javadoc:** “Es una utilidad de Oracle para la generación de documentación de APIs en formato HTML a partir de código fuente Java. Cuando programamos una clase, debemos generar documentación lo suficientemente detallada sobre ella como para que otros programadores sean capaces de usarla sólo con su interfaz.” (Rodríguez, 2017)

1. **Contenido**

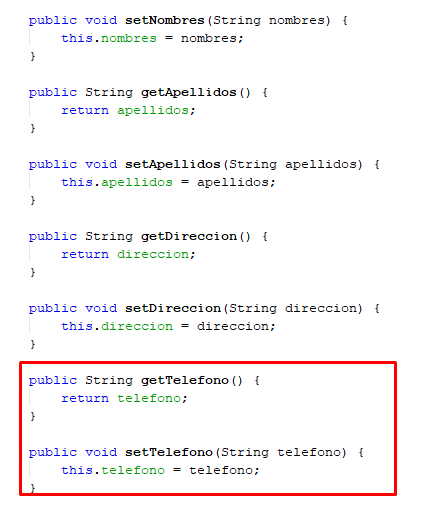
El proyecto “Amigos Peludos” implemento “ArraysList” para el almacenamiento de los datos de las diferentes clases (concursos, animales, auspiciante, dueño, mascota, etc.)

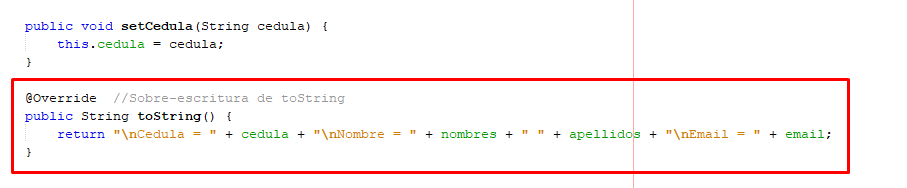


El sistema tiene una validación de errores muy sencillo que nos permite validar los datos, indiferente si es (mayúsculas y minúsculas, singular o plural).

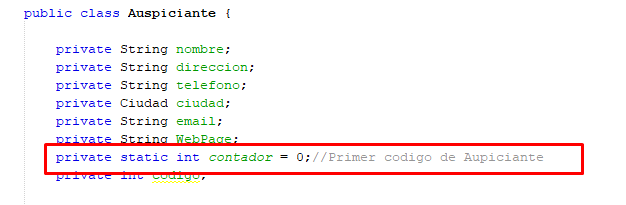


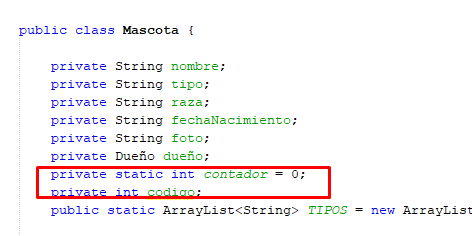
Adicional el sistema de menú es intuitivo, en cada clase se inicializaron setters y getters como también la sobreescritura de algunos métodos como el toString,



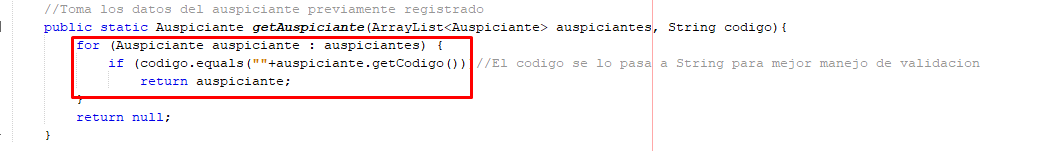


El sistema usa un variable inicializada de carácter privada y estática para poder asignar un código diferente a cada objeto (concurso, mascota, auspiciante) sin embargo para poder identificar el objeto dueño se utilizo el mismo dato ingresado por el usuario(cedula).

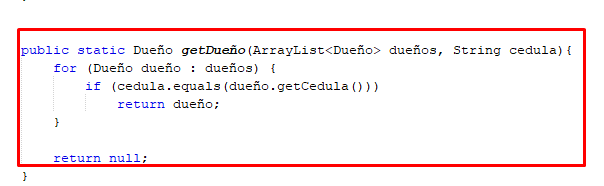


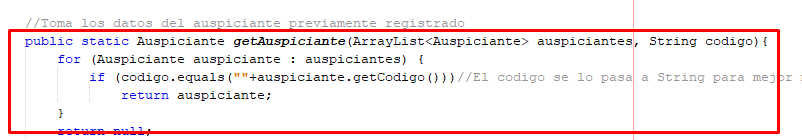


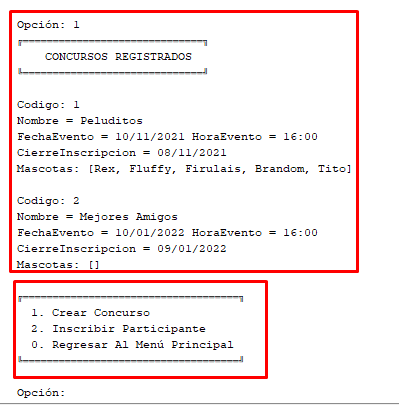
Las validaciones de identificadores y datos fueron transformados a tipo String para un mejor manejo.

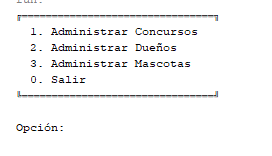


Se implementaron diferentes métodos para el mejor manejo del acceso de los datos de los diferentes arreglos de objetos, por lo que cada clase implemento los métodos necesarios.

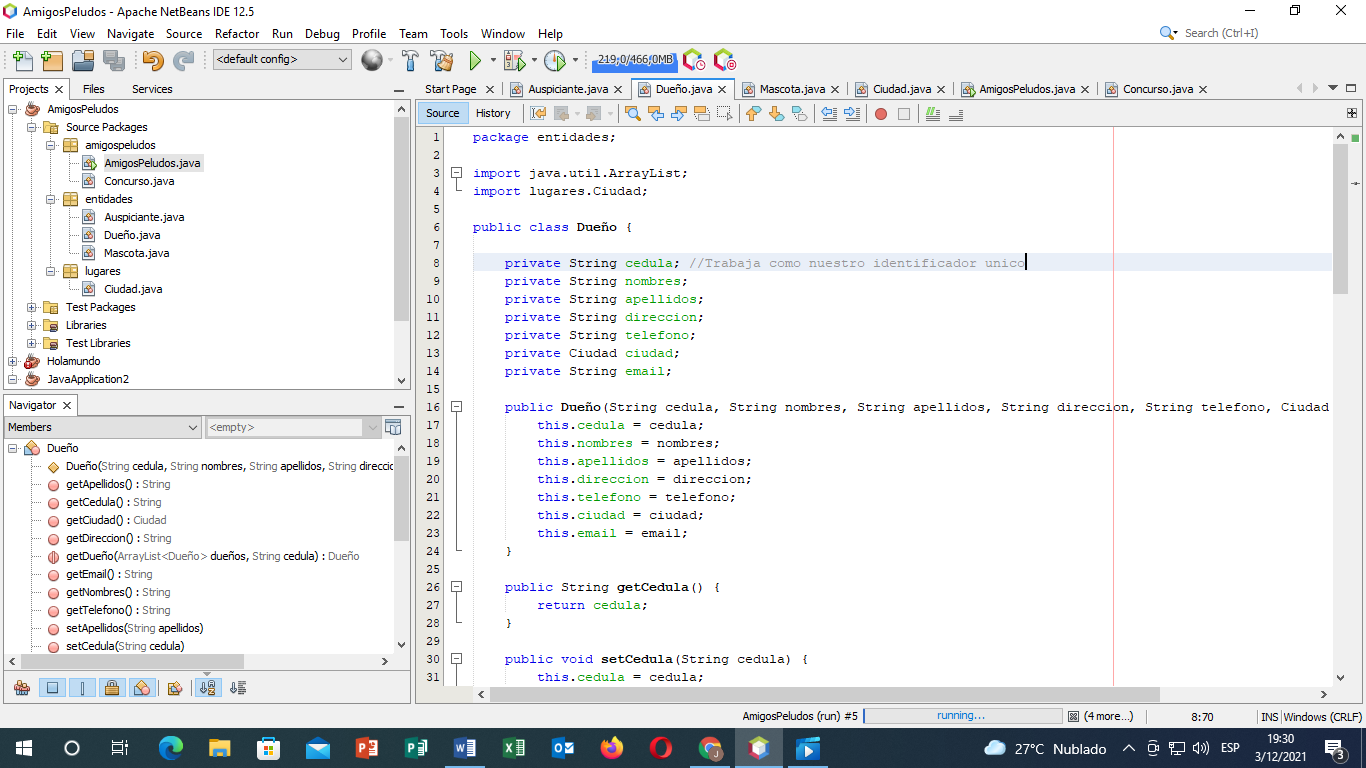




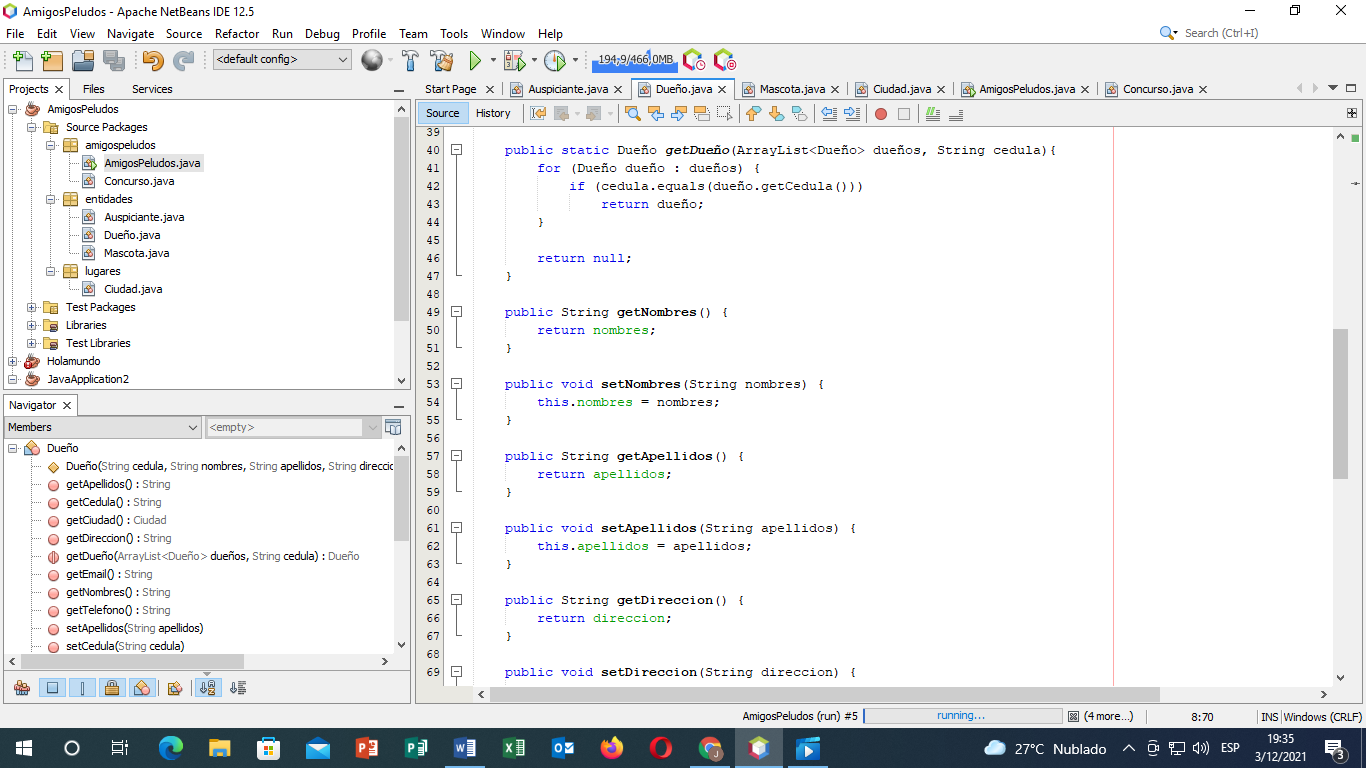
Para un mejor desarrollo de la ejecución del proyecto se elaboraron menús de fácil y agradable manejo, implementados con bucles que permiten la correcta validación de las opciones, así como, también nos permite un simple manejo de errores, como se explicó anteriormente.



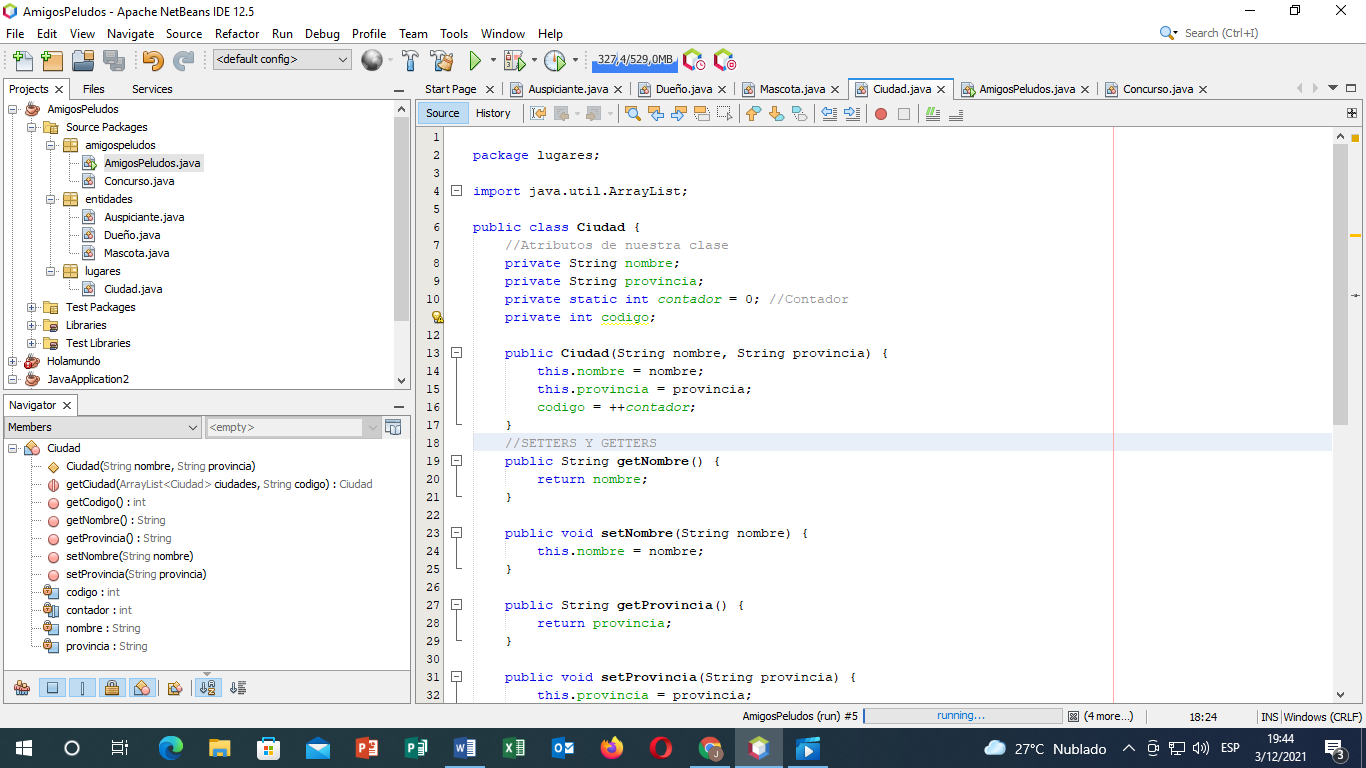
Creación de la clase **“Dueño”** con sus respectivos atributos, teniendo en cuenta que el número de cedula es aquello que distingue a la persona como única.



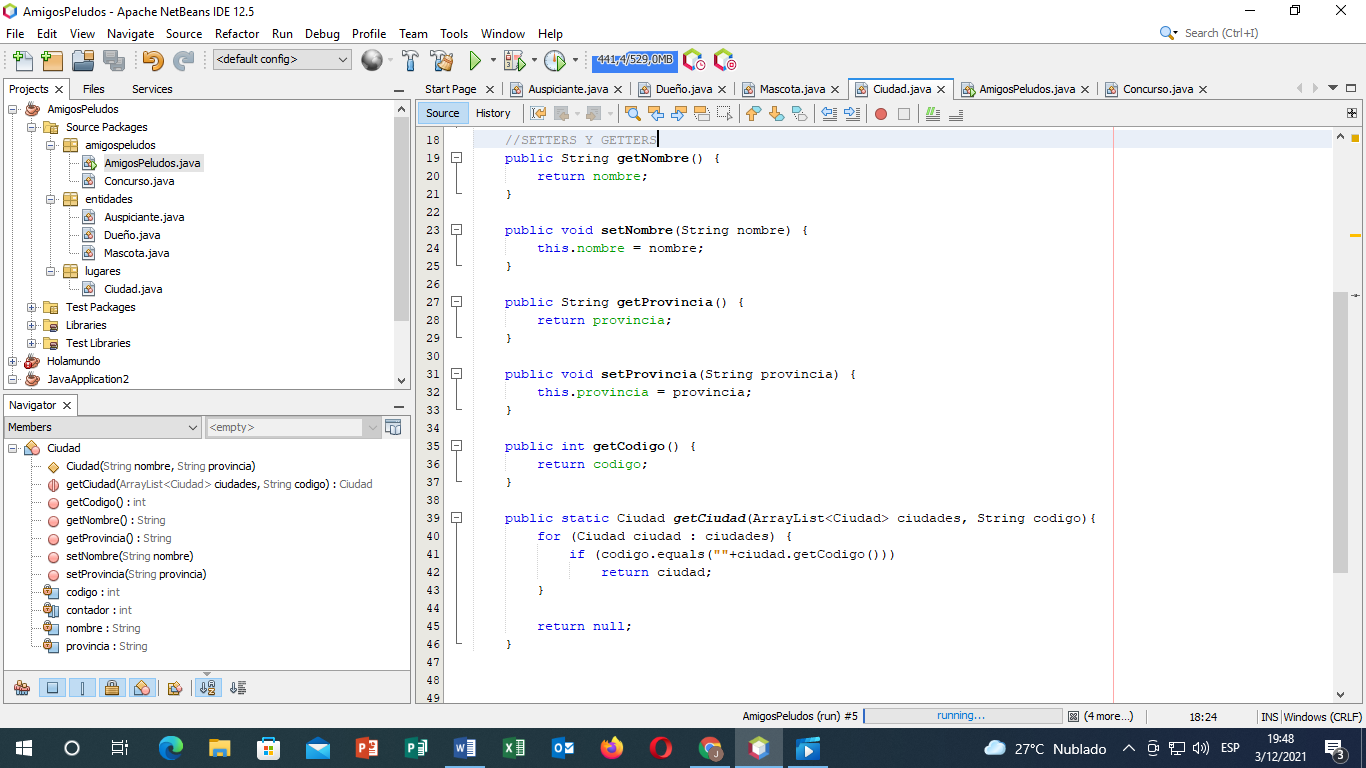
Implementación de los diferentes getters and setters necesarios para nuestro trabajo



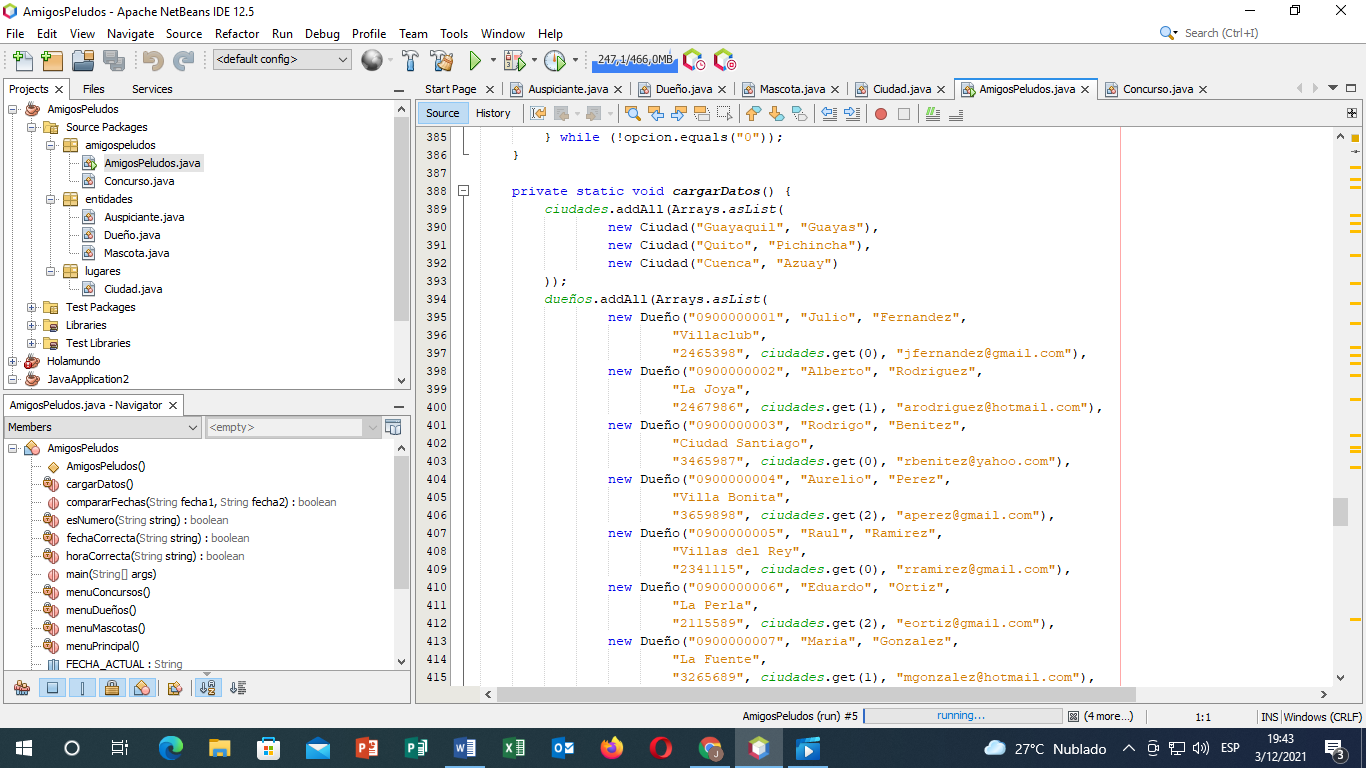
Clase **“Ciudad”** con sus respectivos atributos tomando en cuenta un contador que nos ayudara para llevar a cabo el proyecto



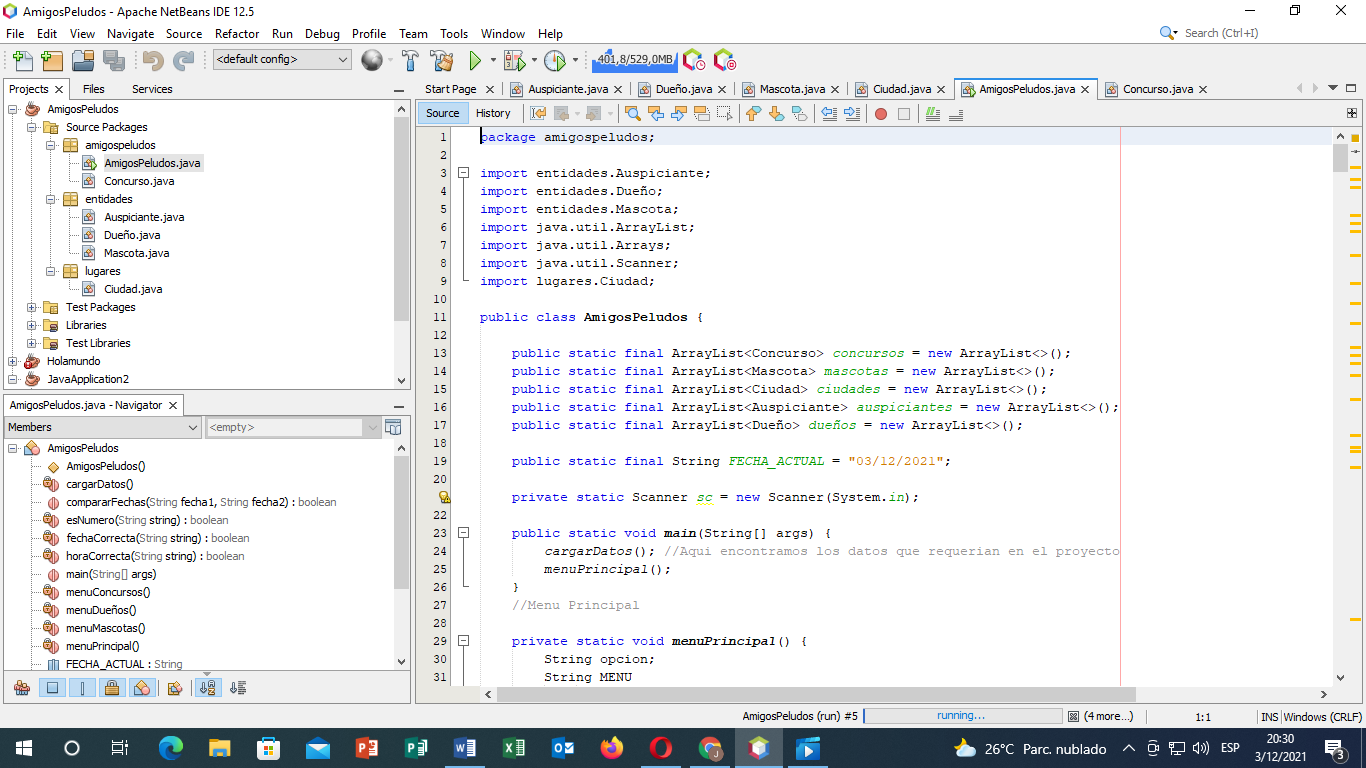
Getters y Setters respectivos que son métodos de acceso a los campos/atributos de una clase la cual se está trabajando.



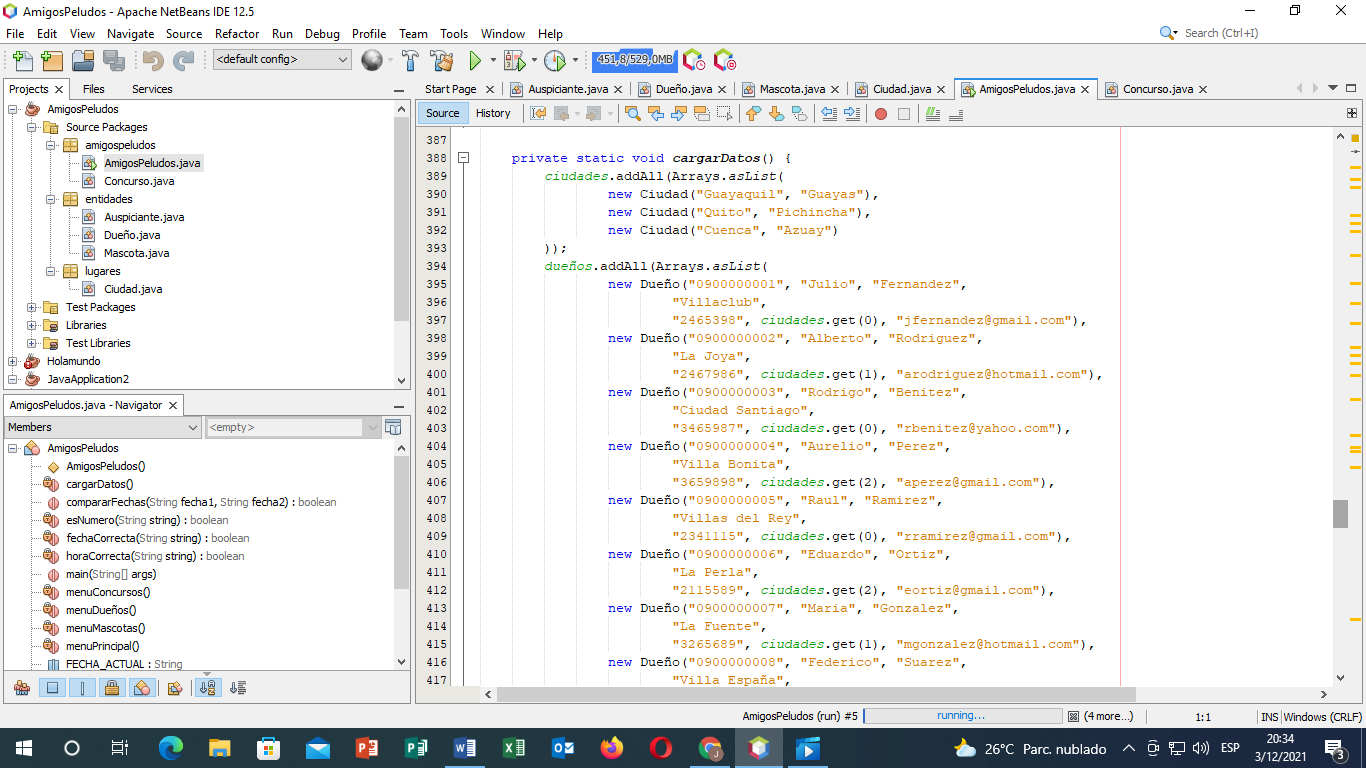
A cada uno se le agrega una ciudad de la que se cargó anteriormente



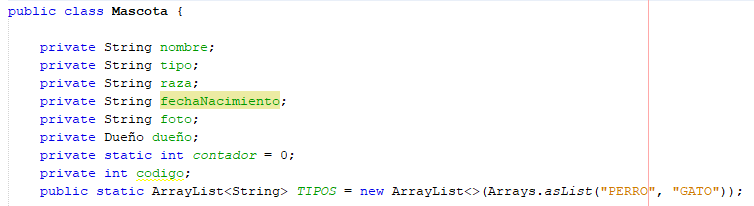
Creación de los ArraysList en nuestro programa principal, que es donde se guardara toda la información.



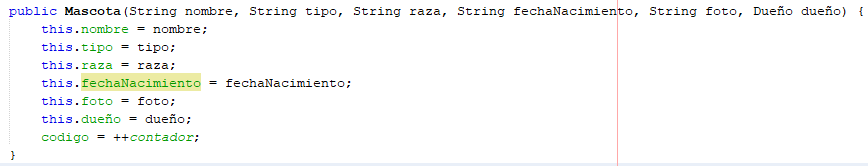
Datos por defecto que debían estar pre-cargados.



Para la clase **“Mascota”** del mismo modo creamos los atributos respectivos tales como nombre, tipo, raza, entre otros y con una respectiva “Lista de arreglo” vacío para la clasificación de perro o gato.

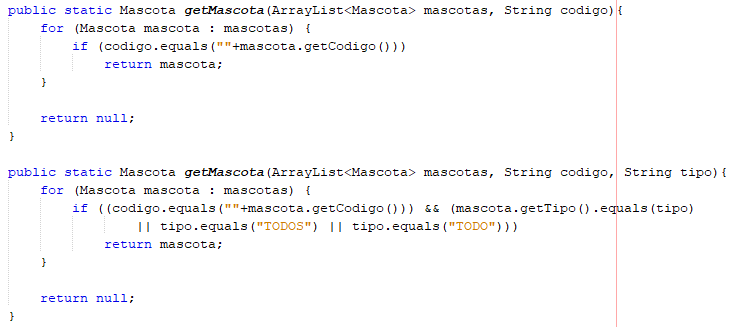


Gato Realizamos el constructor público para “Mascota” de tal modo que así podamos crear nuevos objetos de instancias en otra clase y así ir almacenando los datos de los concursantes de acuerdo si clasifican a la categoría mencionada anteriormente.



Luego procedemos a crear los getters and setters de los atributos con la opción de click derecho “Insert code”, luego “Getters and Setters…” y proceder a seleccionar los atributos.

Del mismo realizamos los constructores para obtener los datos específicos de la mascota de acuerdo al usuario.



# **Referencias**

Ing. Palacios, A.L. (2015) Introducción a la Programación en Java, dirección obtenida en: <http://fcqi.tij.uabc.mx/usuarios/palacios/java4.pdf>

Fundamentos de la Programación Orientada a Objetos (2007) Diagrama de clase, dirección obtenida en: <http://www.utn.edu.ec/reduca/programacion/poo/diagrama_de_clase.html#:~:text=Un%20Diagrama%20de%20Clase%20es,las%20relaciones%20entre%20las%20clases>.

Tokio School (2020) Objeto en Java, dirección obtenida en: <https://www.tokioschool.com/noticias/que-es-objeto-en-java-aprende-programar-java/#:~:text=Pero%2C%20%C2%BFqu%C3%A9%20es%20un%20objeto,de%20compras%2C%20cliente%20y%20producto>.

Charte, Francisco (2019) Paquetes en Java, dirección obtenida en: <https://www.campusmvp.es/recursos/post/paquetes-en-java-que-son-para-que-se-utilizan-y-como-se-usan.aspx>

Rodríguez, Alex (2017) Documentar proyectos Java con Javadoc, dirección obtenida en: <https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=646:documentar-proyectos-java-con-javadoc-comentarios-simbolos-tags-deprecated-param-etc-cu00680b&catid=68&Itemid=188>

Carlos (2016) Programación en Java – Capítulo 3, dirección obtenida en: http://profesores.fi-b.unam.mx/carlos/java/java\_basico3\_4.html#:~:text=La%20herencia%20es%20un%20mecanismo,entre%20clases%2C%20subclases%20y%20objetos.&text=Java%20s%C3%B3lo%20permite%20herencia%20simple.

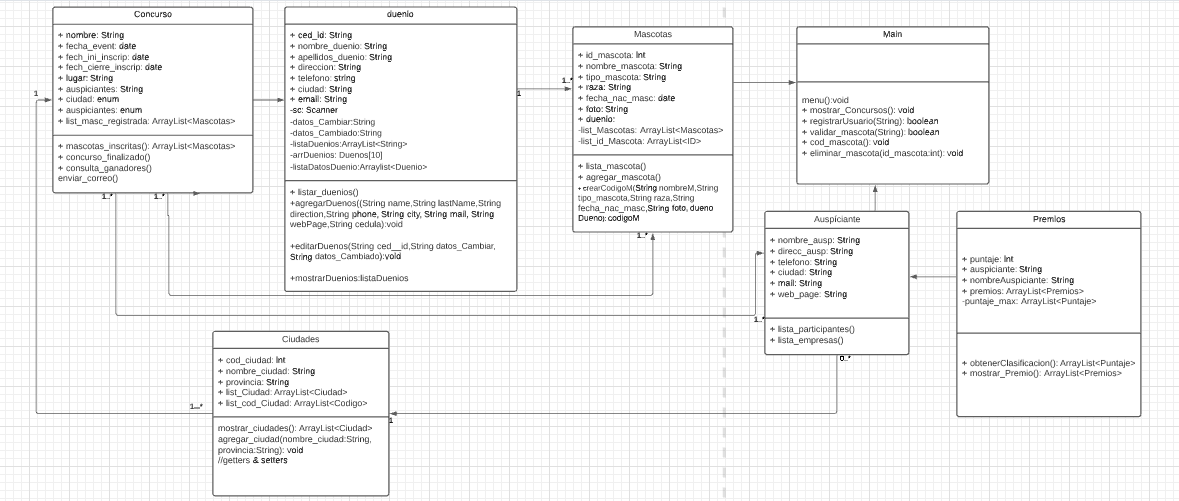
Garro, Arkaitz (2015) Capítulo 17 - Otras relaciones entre objetos, dirección obtenida en: <https://www.arkaitzgarro.com/java/capitulo-17.html#:~:text=En%20una%20asociaci%C3%B3n%2C%20dos%20instancias,s%C3%AD%20existen%20de%20forma%20independiente.&text=La%20relaci%C3%B3n%20de%20asociaci%C3%B3n%20expresa,denomina%20navegabilidad%20de%20la%20asociaci%C3%B3n>.

Ángel Álvarez, Miguel (2008) Instanciar y trabajar con objetos de clases creadas con Mootools, dirección obtenida en: <https://desarrolloweb.com/articulos/instanciar-trabajar-con-objetos-clases-mootools.html#:~:text=Instanciar%20objetos%20es%20el%20proceso,que%20responde%20a%20esa%20forma.&text=Podemos%20crear%20infinitos%20objetos%20a,un%20ejemplar%20de%20esa%20clase>.

1. **Anexos**

* **Diagrama de Clase – UML:**

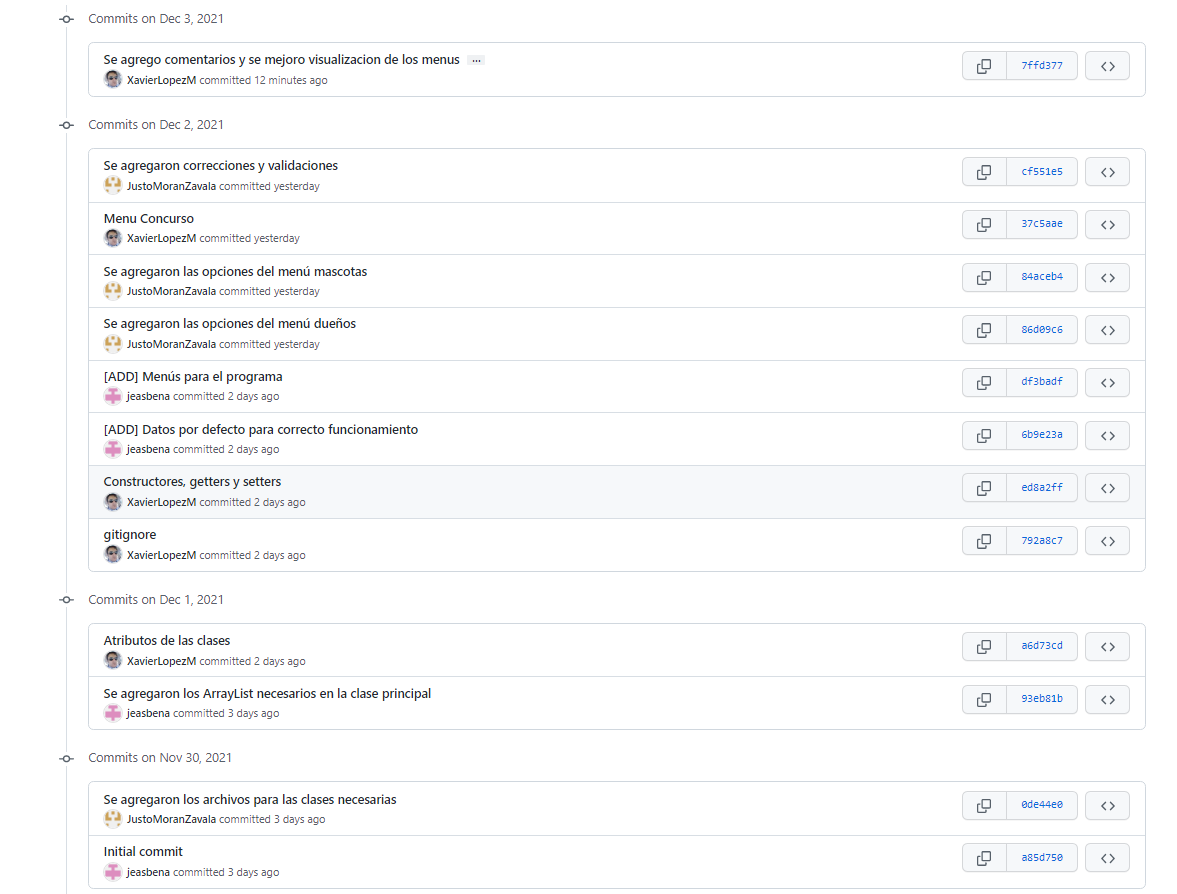
<https://lucid.app/lucidchart/bcb8e353-6298-49c1-8561-c64817d8e35f/edit?invitationId=inv_dd7b9a11-e616-45a2-95bb-f30dd044b6e3&page=0_0#>



* **Distribución de tareas:**

|  |  |
| --- | --- |
| Nombre | Tareas |
| Jeanpiere Benalcázar R. | Creación de Clase Dueños, Ciudad, Setters Getters, Constructores, Comentarios, Objetos, ArrayList, Menús, Precarga de Información dada. |
| Justo Moran Zavala | Menú principal de la Clase “AmigosPeludos” usando Do-While, al igual que el constructor “menuConcursos” en colaboración con Jeanpiere, Clase Mascota, Setters and Getters y comentarios. |
| Xavier Lopez M. | Menú concurso, Constructores, Setters Getters, Mejora Visualización Menús, Documentación, Clase Concurso, Atributos, Observaciones y comentarios. |

* **Commits realizados por los integrantes:**



* **Reporte del Repositorio:**

<https://github.com/jeasbena/AmigosPeludos.git>