**CORHUILA**

**ESTRUCTURA DE DATOS**

**RESUMEN CLASE ESPEJO**

**HERENCIA-POLIMORFISMO**

Presenta:

**JUAN CAMILO BERNAL GORDILLO**

Profesor:

**Jesús Ariel Gonzales Bonilla**

Facultad del estudiante:

**Ing. De Sistemas**

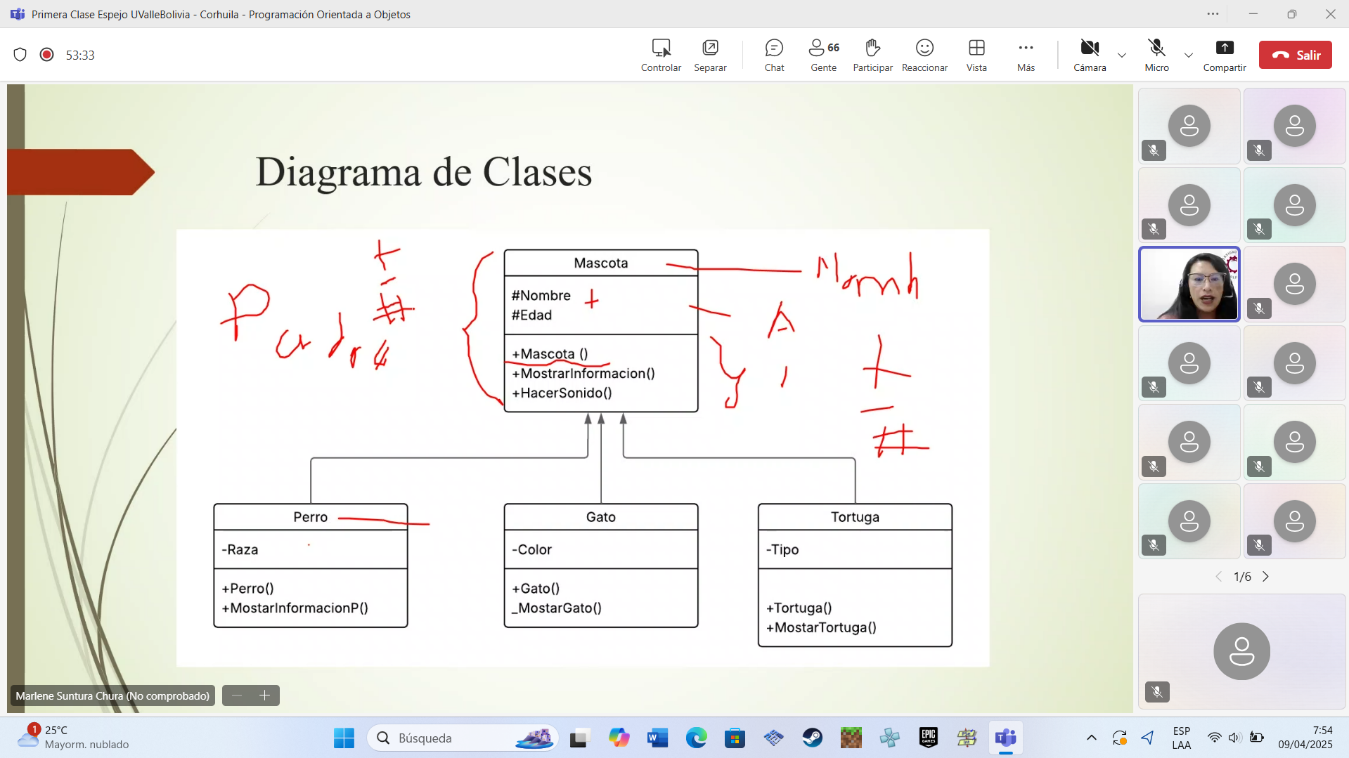
**Neiva-Huila Abril,2025**

**HERENCIA**

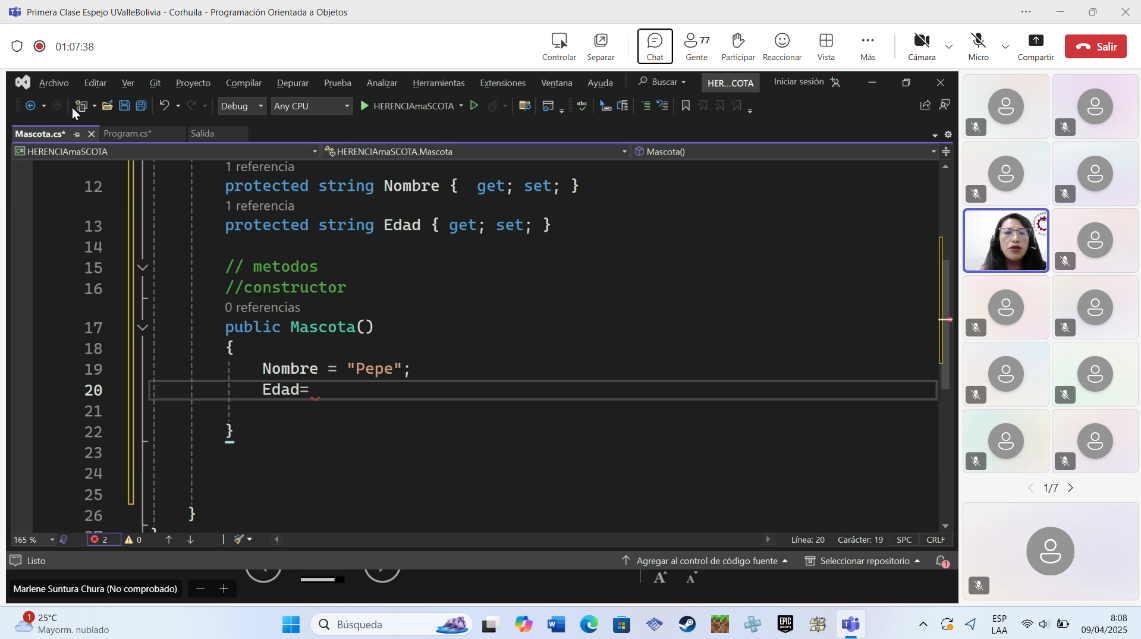
La clase inició 10-15 minutos después de lo programado mientras se daba espera al ingreso de los estudiantes que iban a participar en la reunión, la reunión se llevaría acabo por la maestra Marlene Suntura con estudiantes de la Universidad de Bolivia (UniValle).

La maestra explico primero el concepto de POO, que básicamente es un paradigma enfocado en clases y objetos. Nos explico que la POO tiene cuatro bases fundamentales, una es la herencia (cuando las subclases heredan atributos de una clase superior); encapsulamiento (ocultar los datos internos de un objeto y permitir el acceso solo a través de métodos controlados); polimorfismo (tener los mismos métodos pero representados de manera diferente); abstracción (mostrar lo importante y ocultar lo innecesario.).

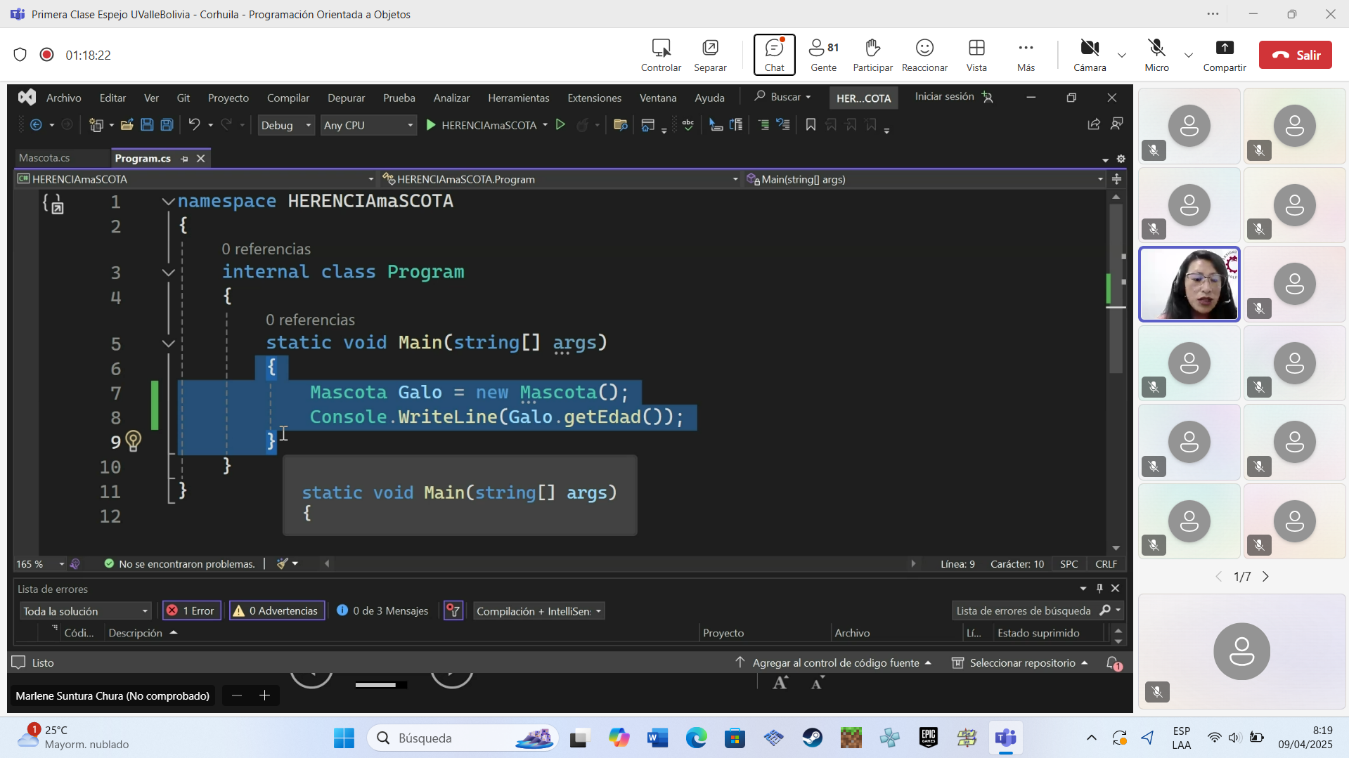
Como la clase estaba enfocada en herencia y polimorfismo, iniciamos con herencia donde la maestra mostrándonos una imagen (de un personaje de Mario y su hijo), nos explicó de manera clara que la herencia es aquellas características que la clase hija hereda de la clase padre. Para hacernos entender esto a nivel de programación nos elaboró este diagrama de clase para representar el ejercicio que íbamos a realizar como manera de ejemplo:

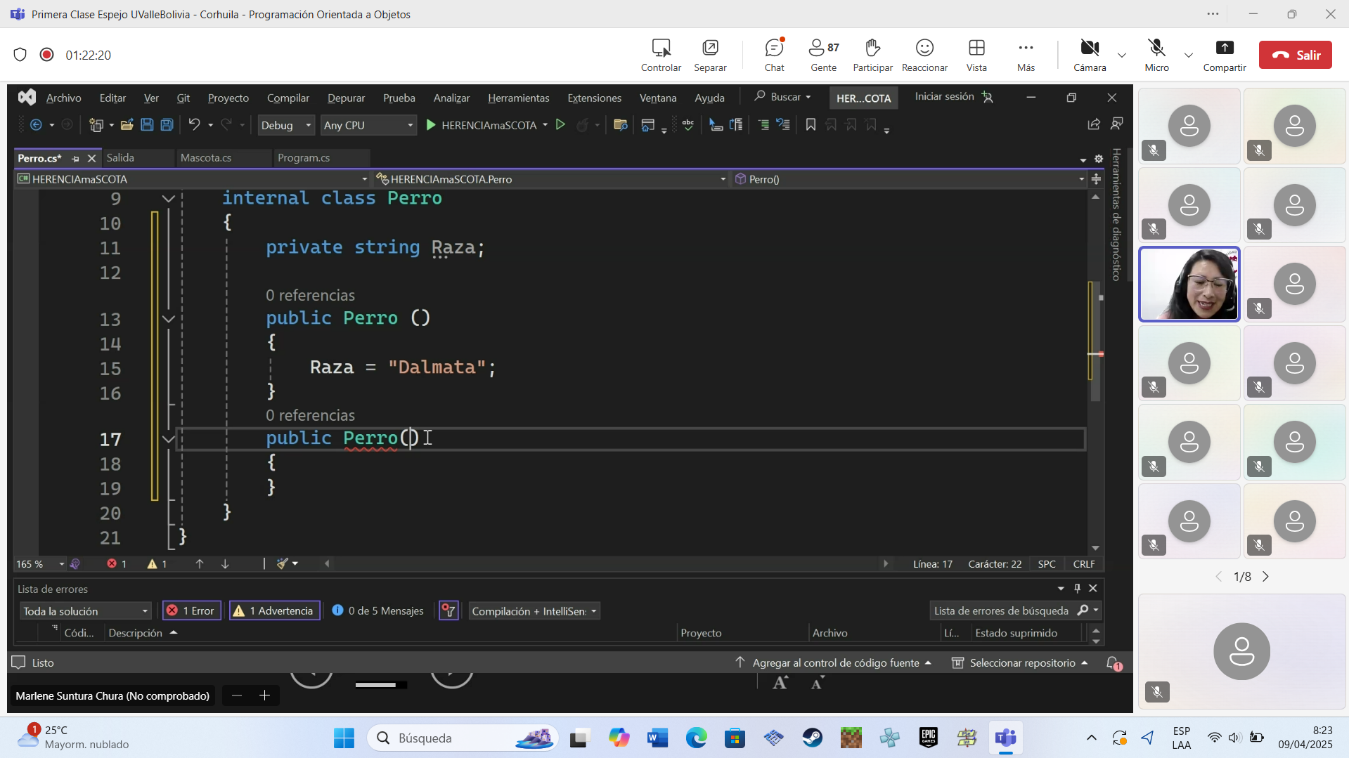
En este diagrama se evidencia que hay una clase padre llamada Mascota la cual tiene como atributos el nombre y la edad, también tiene métodos como el constructor, el de mostrar la información y hacer sonidos. A su vez, de esta clase padre se derivan 3 subclases hijas (perro, gato, tortuga), estas clases tendrían tanto método constructor como de mostrar información.

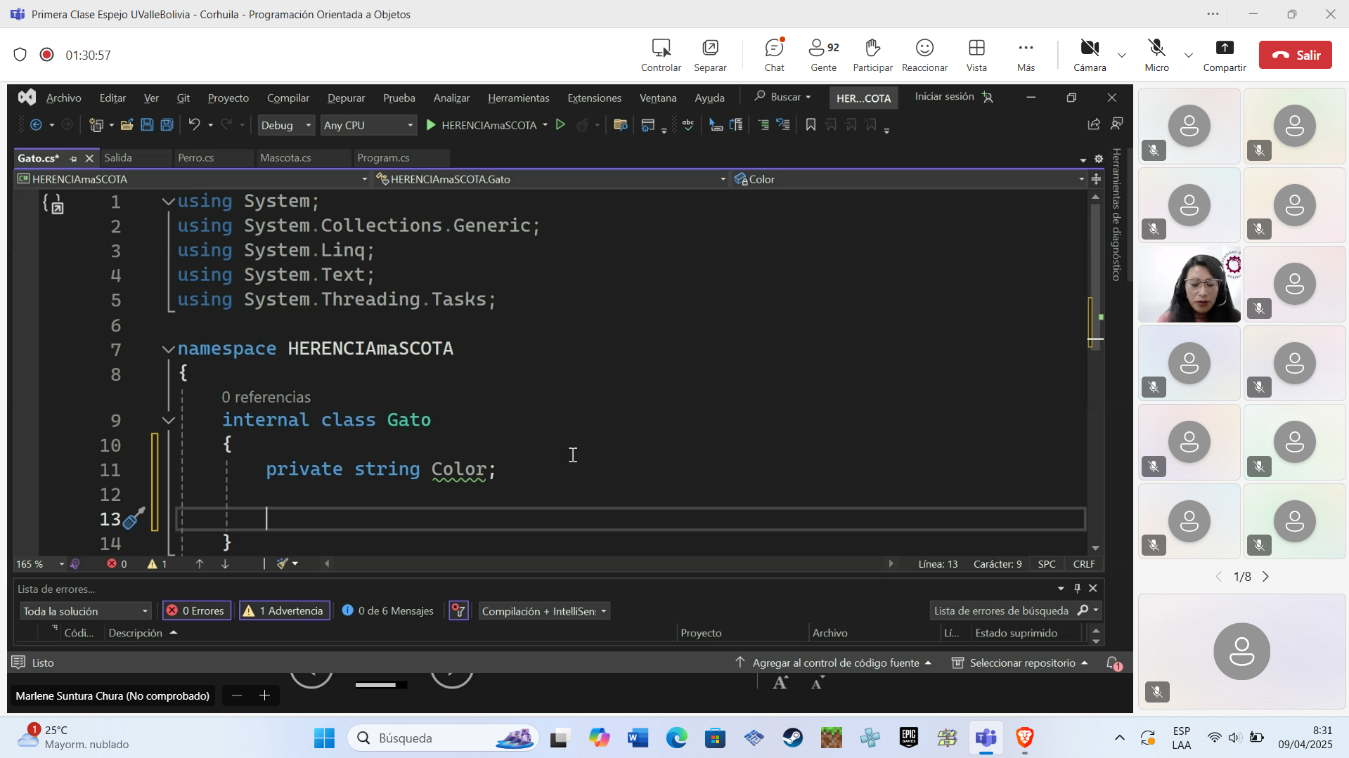
En cuanto el código trabajamos en Visual Studio (IDE) con el lenguaje de programación C#, nos enseñó como usarlo, creando clases, haciendo ejecuciones, etc.



Primeramente, iniciaría creando la clase padre que como se ve en el diagrama de clases sería la clase Mascota donde se le asignarían los atributos de nombre y edad, en cuanto al método de acceso lo puso en protected para que las subclases pudieran acceder a él, nos explicó más adelante que no los puso en private porque las subclases no podrían acceder a esos atributos como si lo hacen con protected. Después de ello, armó el método constructor de la clase (uno definido y el otro parametrizado) procedente del get y set de cada atributo, luego de ello armó un método donde iba a mostrar la información.

Posteriormente, crearía una clase llamada programa donde ejecutaría el código, dentro de esta clase crearía el objeto llamado Galo y mostraría la información en pantalla, en la impresión de pantalla saldría únicamente la edad que sería en este caso 5.

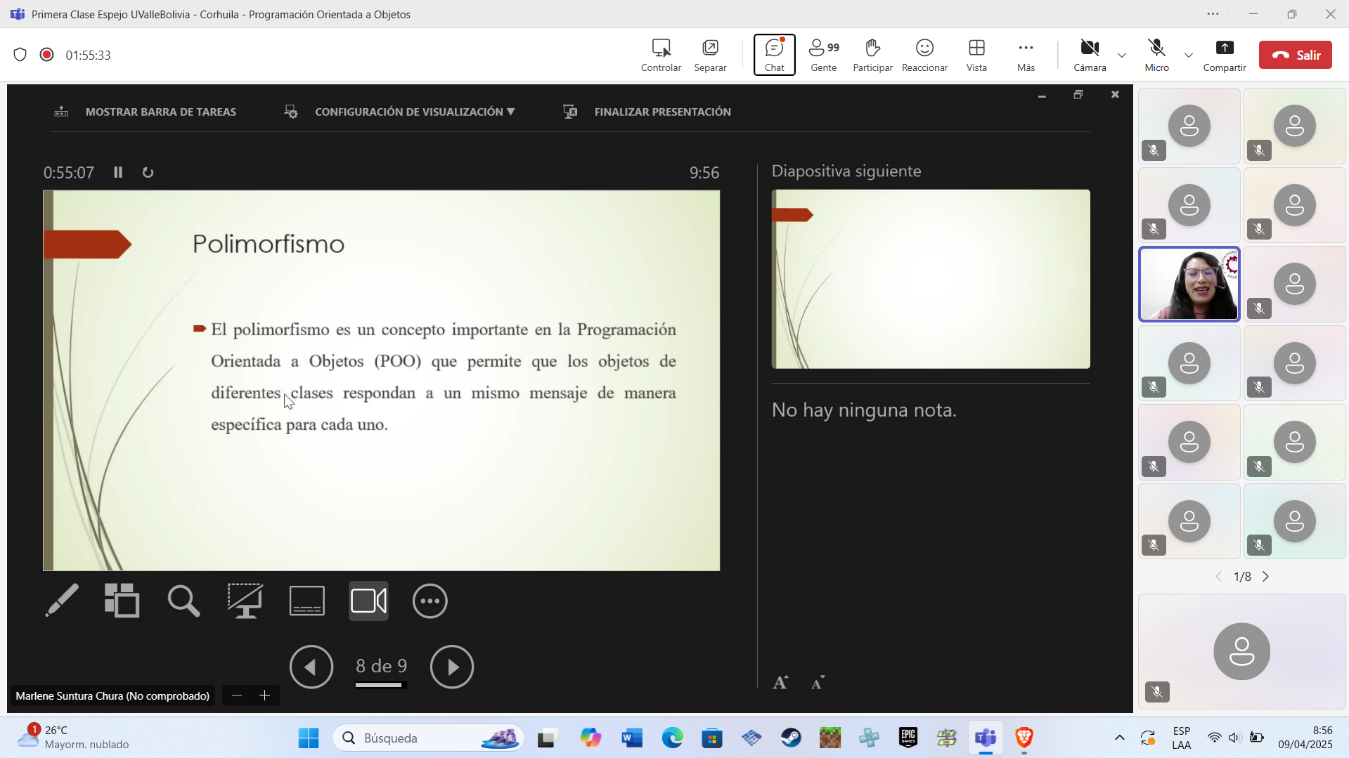


Luego de habernos explicado eso, creó una clase hija llamada Perro en la cual iba a quedar demostrado el concepto de herencia, a esta clase le asignó un atributo propio que sería Raza, luego haría el método Constructor uno definido y otro parametrizado, destacando los atributos que iba a heredar esta clase y el atributo que esta tenía, aquí nos explicó que en C# para aplicar la herencia se escribe enseguida de la clase **:** y en seguida la clase padre. Luego del método constructor, realizaría el encapsulamiento del atributo raza (get y set) y realizaría otro método en esta subclase para imprimir la información, dentro de este método llamaría el método de información de la clase padre para aplicar de ese modo la herencia. De igual manera, se iría a la clase programa donde crearía un objeto para mostrar la información de la subclase perro, en esta se imprimiría por pantalla no solo la edad, sino también el nombre y la raza del animal.

De la misma manera construiría una subclase llamada Gato que tendría los atributos de la clase padre y un atributo propio que sería el color, para este haría lo mismo de armar el método constructor (definido, parametrizado), hacer el encapsulamiento y crear un método para mostrar información que contenga el método de mostrar información de la clase padre. De aquí, se iría para la clase programa y crearía un objeto para mostrar la información que estaba dentro del método de información de la subclase gato, pero esta vez quería que se imprimiera en pantalla un gato con edad 4 y otro con 7 para mostrar la herencia, para ello en el objeto que creó definiría el nombre del animal, la edad y el color, seguido de esto llamaría al método de información del gato el cual mostraría el nombre del gato, la edad 4, y el color. De la misma forma, para mostrar la herencia del método uso un setEdad para actualizar la edad a 7, y esta vez iba a salir la misma información pero con la edad 7 donde quedaría comprobado de como la herencia permite a las subclases acceder a los atributos de la clase padre (mascota). Cabe aclarar, que acá hubo una pequeña complicación que sirvió para darnos cuenta que debemos tener cuidado al momento de escribir el código, ya que al ser lenguajes estrictos en sintaxis no puedes dejar de cerrar una llave, un punto y coma o un paréntesis, también resalto que la subclase tortuga no la realizamos por el tiempo y porque ya con las dos anteriores quedaba claro el concepto de herencia.

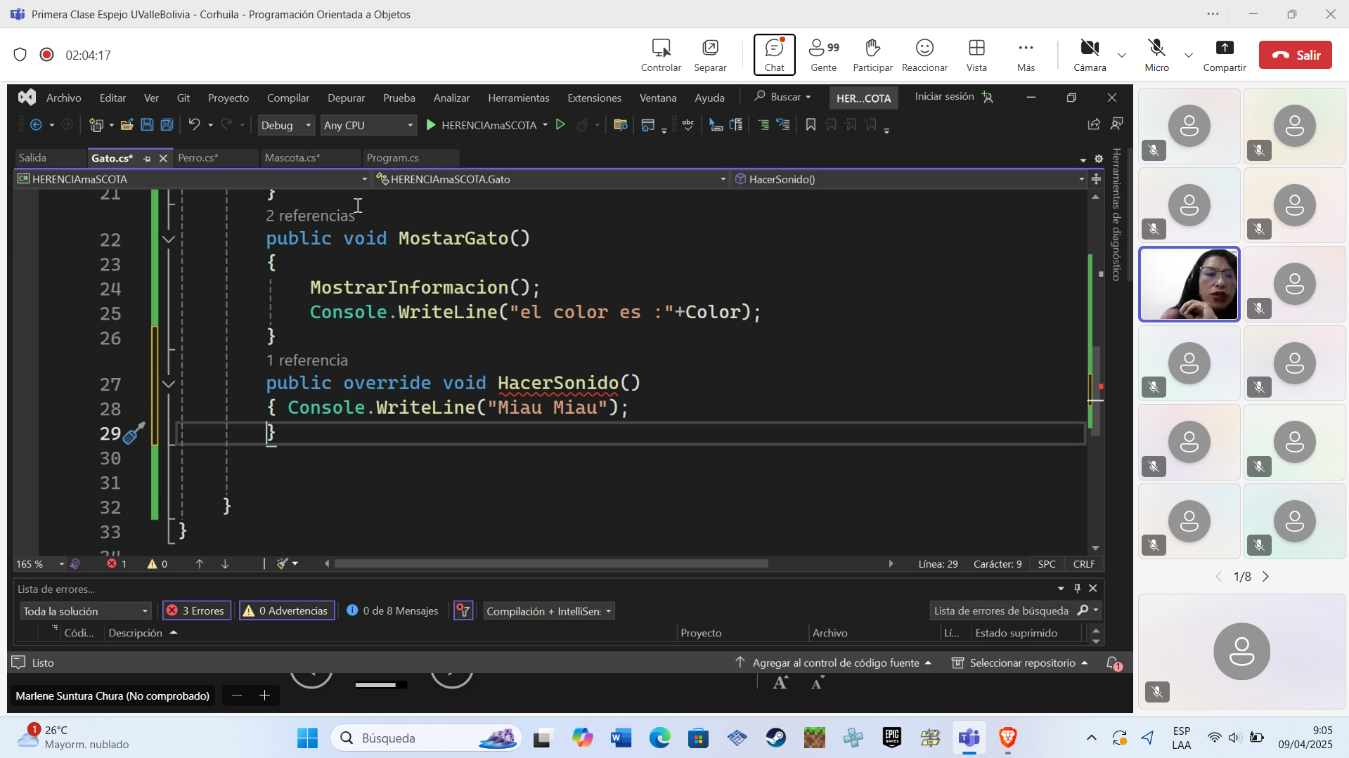
Antes de pasar a polimorfismo, nos explico que la herencia es conveniente aplicarla cuando se tienen atributos parecidos.

**POLIMORFISMO**



En polimorfismo nos explicó que es donde se tienen métodos con el mismo nombre pero que se comportan de diferentes maneras. Nos dijo que en la herencia ya habíamos usado el polimorfismo en los métodos constructor, ya que usábamos para cada clase un método definido y otro parametrizado, que tenían el mismo nombre pero que se comportaban de diferente forma.

A parte de esto nos dio un ejemplo claro, donde en la clase padre creó un método llamado HacerSonido en el cual imprimiría un mensaje diciendo que la mascota emite sonido. Posterior a ello, se iría para programa y nos mostraría que por Herencia cada subclase podría utilizar este nuevo método. Aplicando Polimorfismo definiría el sonido que hace cada animal, que en este caso sería para el perro un ladrido y para el gato un maullido.



Para aplicar el polimorfismo utilizaría una palabra reservada llamada override donde utilizaría el mismo nombre del método de hacer sonido de la clase padre pero que este se iba a comportar de acuerdo a su naturaleza, para poder utilizar el mismo nombre de la clase padre y que no generara error, se le antepuso al void del método de la clase padre la palabra reservada virtual, de esta forma las subclases podrían heredar el mismo método, pero comportándose de acuerdo a su naturaleza.

Por último, en el programa llamo el método de hacer sonido heredado de la clase padre, pero al momento de mostrar en pantalla se evidenció como cada subclase heredó efectivamente el método de hacer sonido, pero implementado de diferente forma (ej: miau miau). Ya aquí finalizaría la explicación y nos daría el espacio para aclarar dudas para posteriormente despedirse de forma muy respetuosa.