

**PROYECTO DE DISEÑO DE SOFTWARE**

# TEMA DEL PROYECTO:

Realizar un sistema estructurado para la empresa MyHome S.A diseñado para elaborar casas para sus clientes

**INTEGRANTES:**

* Manzano Manzano Bryan Andrés
* Portilla Salazar Christian Fabricio
* Riera Naranjo Genesis Juliana

**PROFESOR:** MsC. Mosquera Jurado David

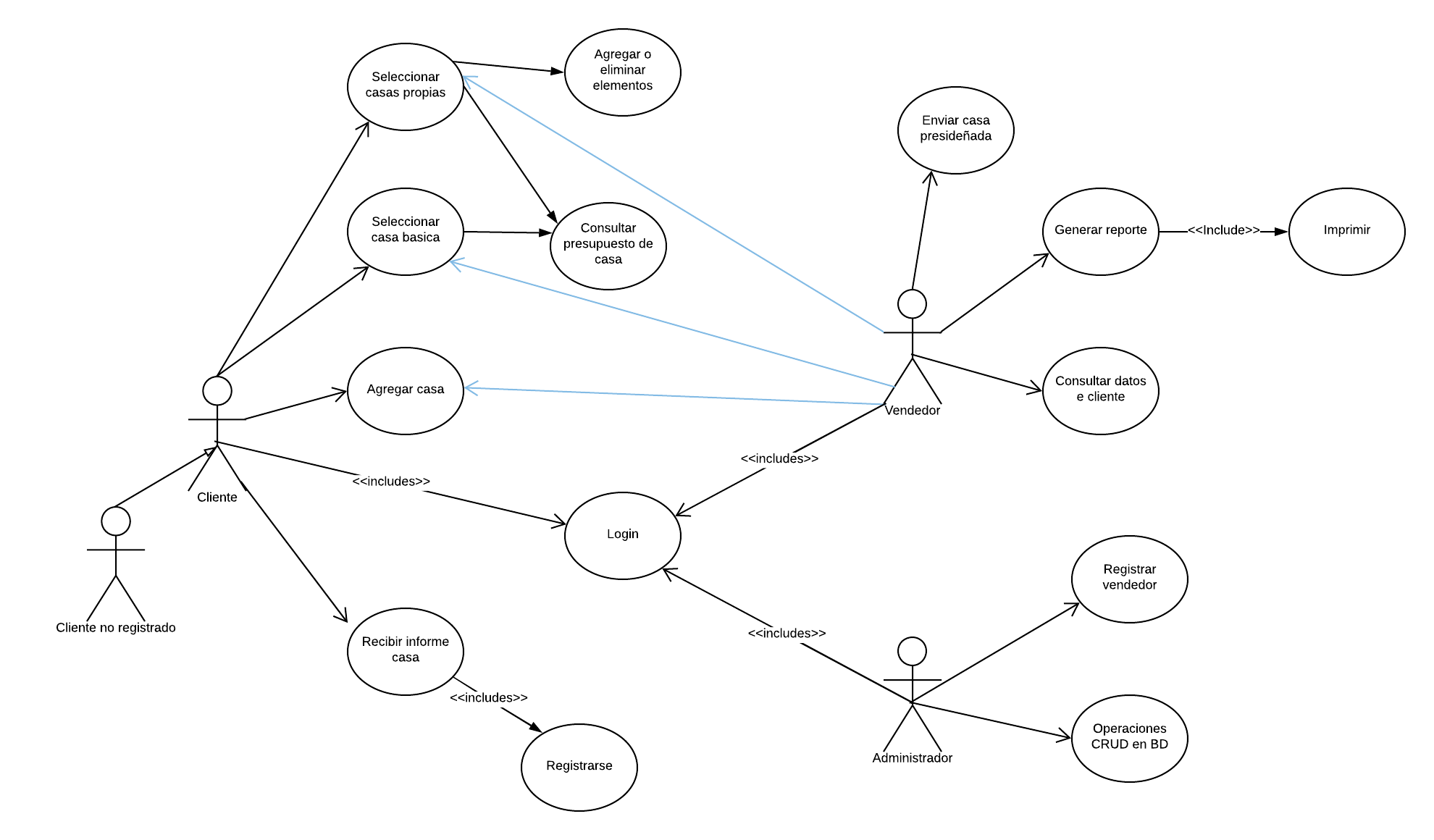
**PARALELO:** 1

**GUAYAQUIL – ECUADOR**

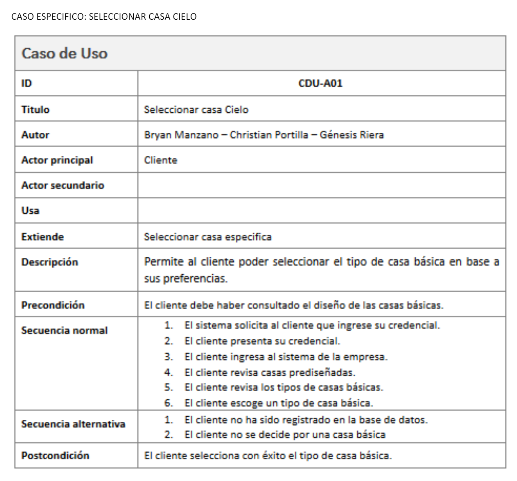
## **Descripción del proyecto**

El presente proyecto propone realizar un sistema para la empresa “MyHome S.A” que permita a sus clientes realizar diseños de casas, el diseño del sistema debe presentar diferentes interfaces para cada usuario (cliente, vendedor, administrador) de esta manera podrán ingresar y elegir entre las opciones disponibles dependiendo al rol que desempeñen. La empresa en la actualidad consta con una sola oficina, pero desea expandirse colocando quioscos en los centros comerciales para poder obtener más clientes. Para realizar el sistema de la empresa “MyHome S.A” se aplicaron conceptos aprendidos durante el primer y segundo parcial de la materia Diseño de Software.

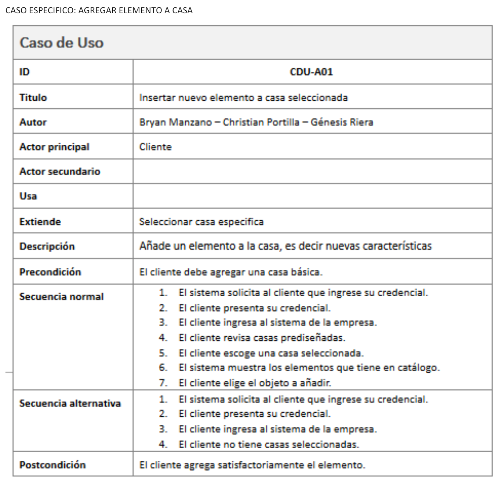
## **Diagrama de Casos de Usos general**



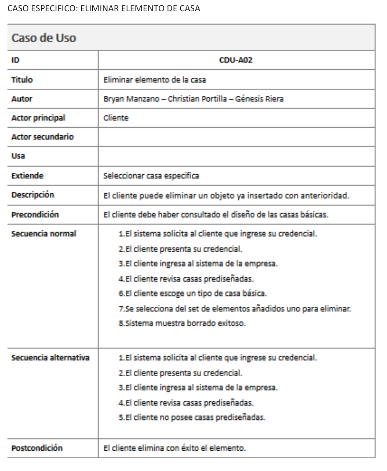
## **Caso de Uso: Seleccionar casa cielo**



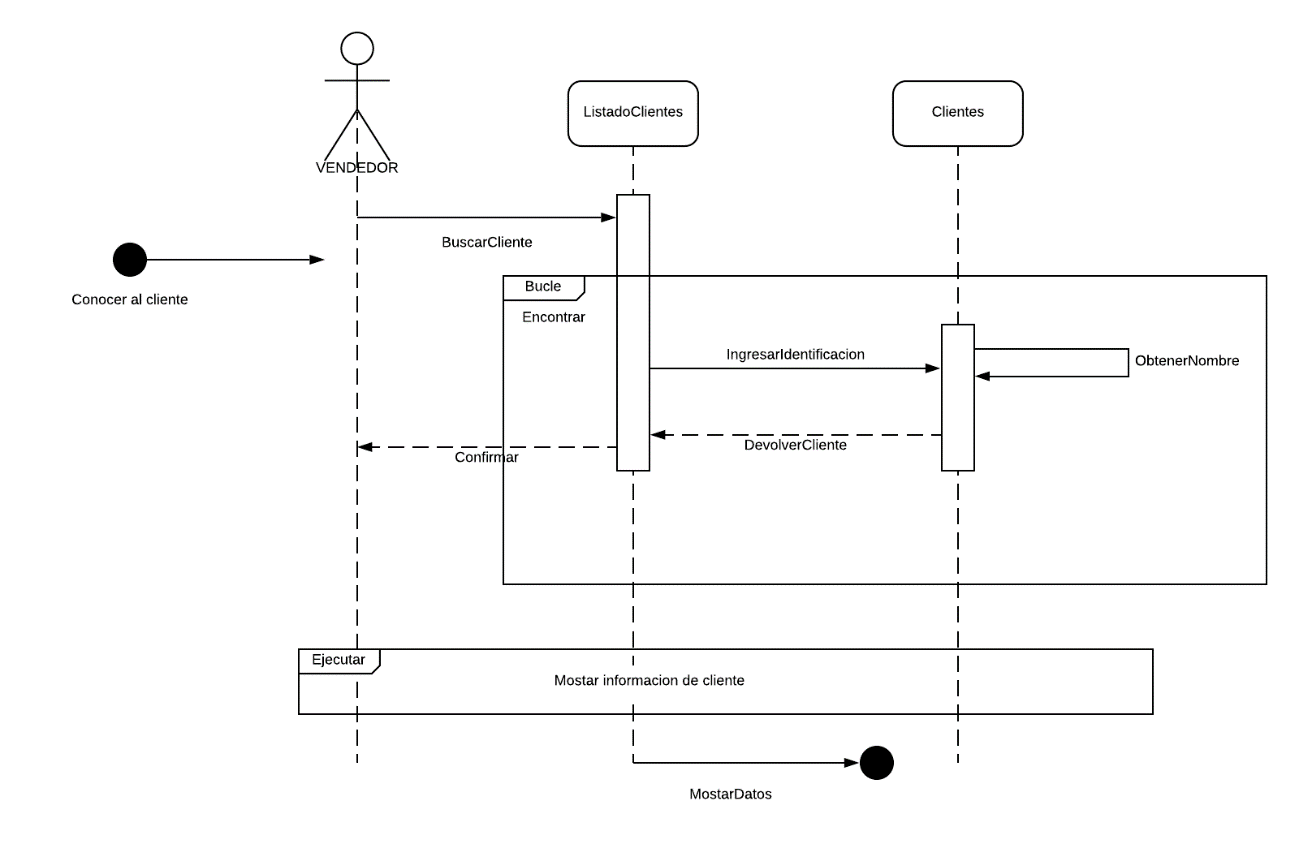
## **Caso de uso: Agregar elemento a casa**



## **Caso de uso: Eliminar elemento de casa**

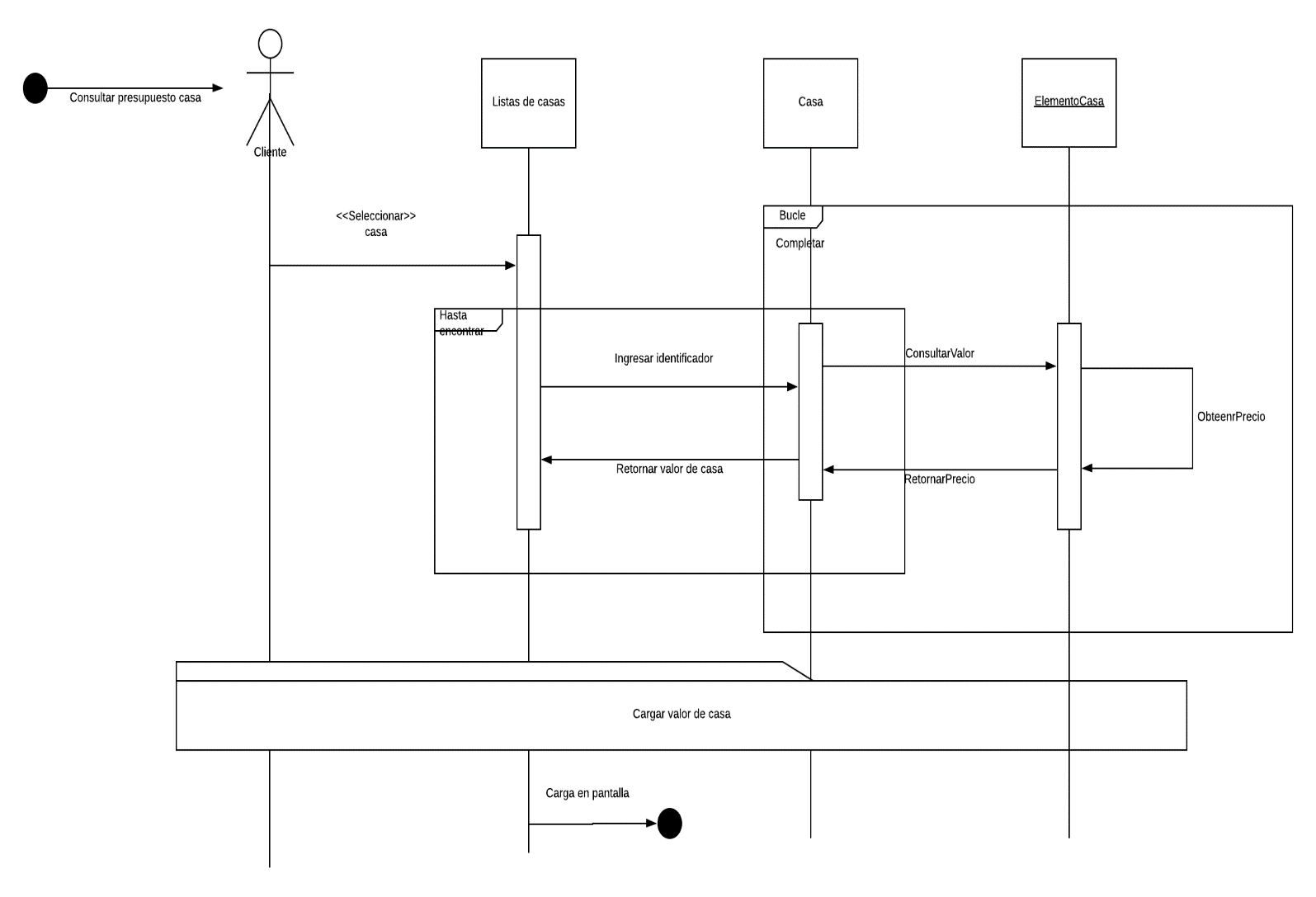


## **Diagrama de secuencias: Datos Cliente**

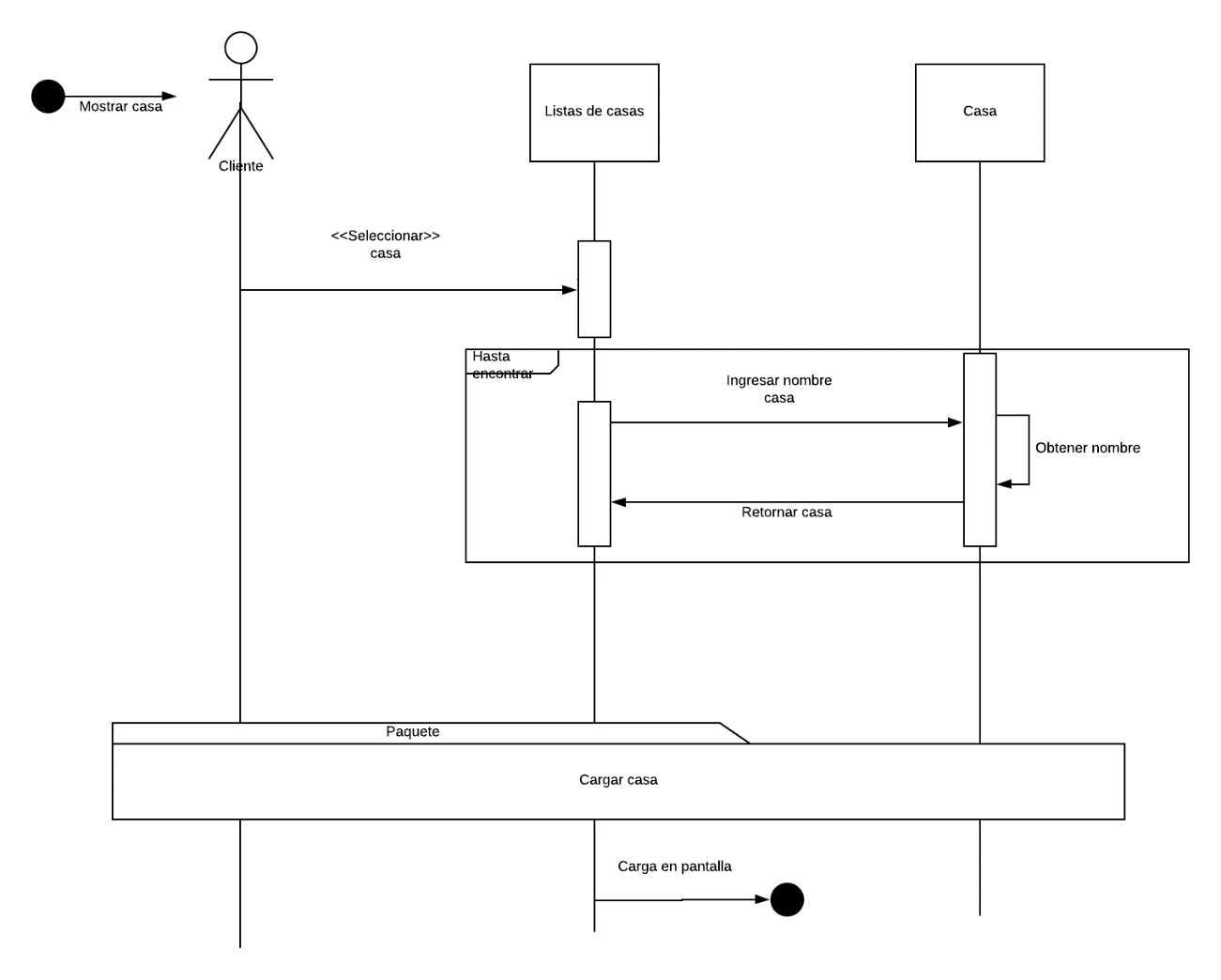


## **Diagramas de secuencias: Ofertar casa**

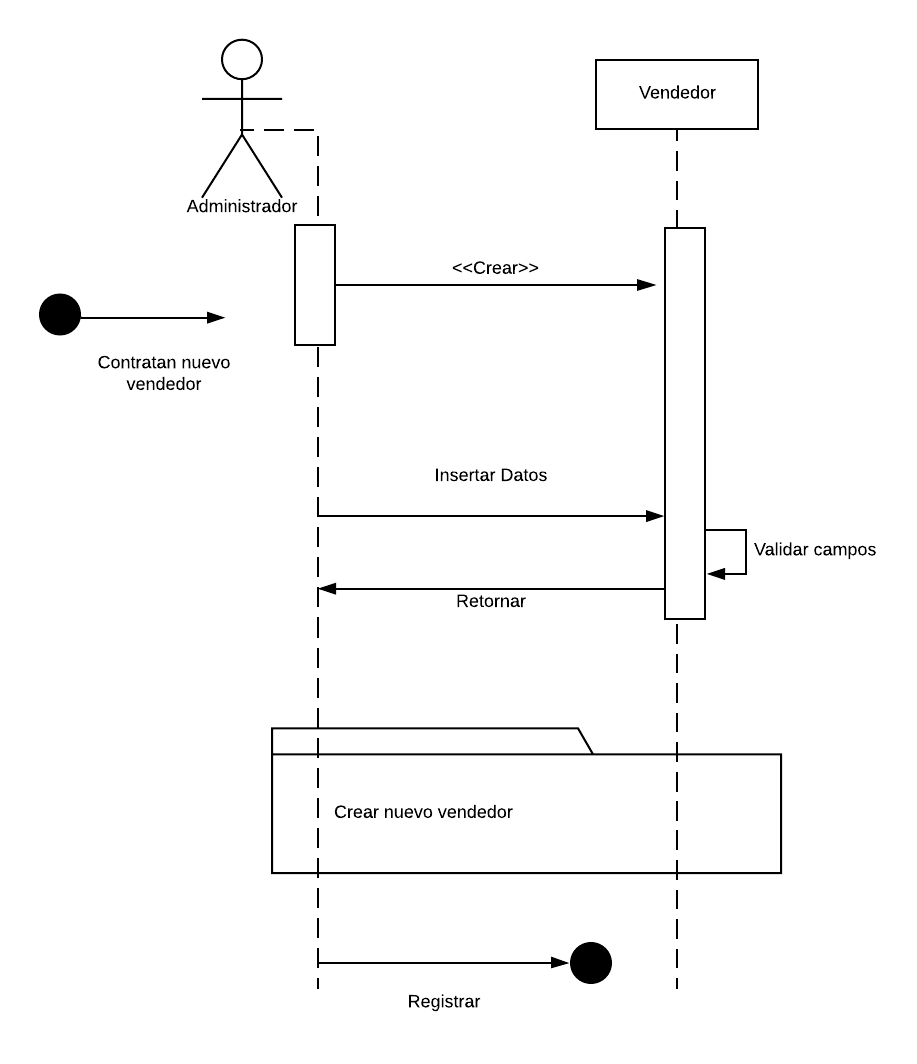
## **Diagrama de secuencias: Presupuesto de casa**



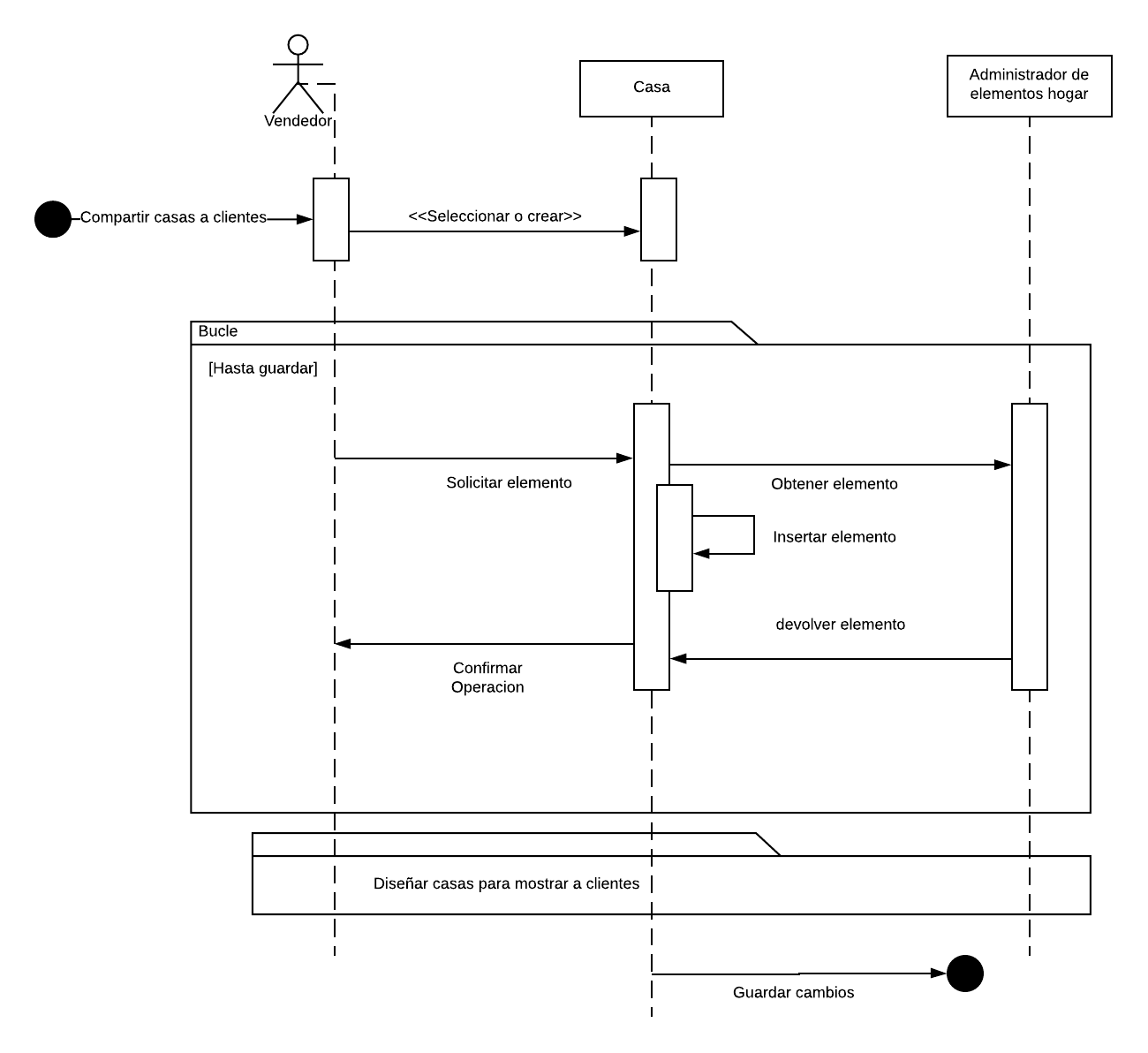
## **Diagrama de secuencias: Mostrar casa**



## **Diagrama de secuencias: Crear vendedor**

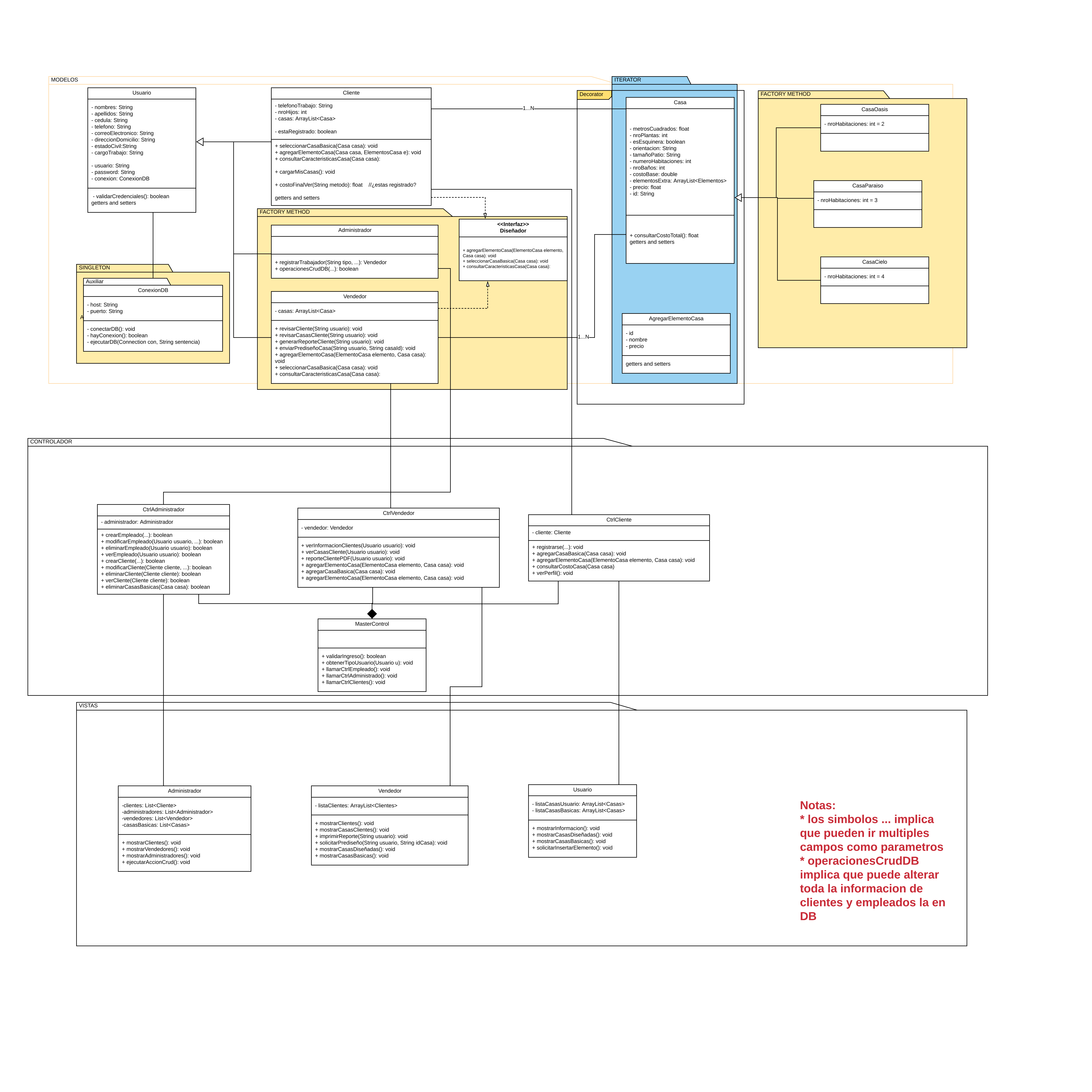


## **Diagrama de secuencias: Vendedor diseña casas**



## **Debido a que no se puede observar correctamente el diagrama de clases, se lo subirá aparte.**

## 



## **Patrones de diseño**

### **Facthory Method**

Debido a que el patrón de diseño Factory Method aparece como un modelo para crear objetos se lo implementó en las clases: Casa, Casa Cielo, Casa Paraíso, Casa Oasis. En este caso la clase que se usaría como patrón es la clase **Casa,** la cual permitirá crear nuevos objetos en este caso serian casa cielo, casa paraíso y casa oasis. Este patrón de diseño además nos proporciona un acoplamiento flexible y alta cohesión.

## **Singleton**

El patrón de diseño Singleton permite crear una sola instancia de una clase virtual de Java para poder instanciar diversos objetos, en nuestro proyecto se usó este patrón de diseño en el paquete Auxiliar debido a que se requiere controlar una acción durante su ejecución en este caso conectar la base de datos con el programa, puesto que permitirá leer desde el archivo de configuración la primera vez y en la segunda llamada leerá los datos de los objetos en memoria.

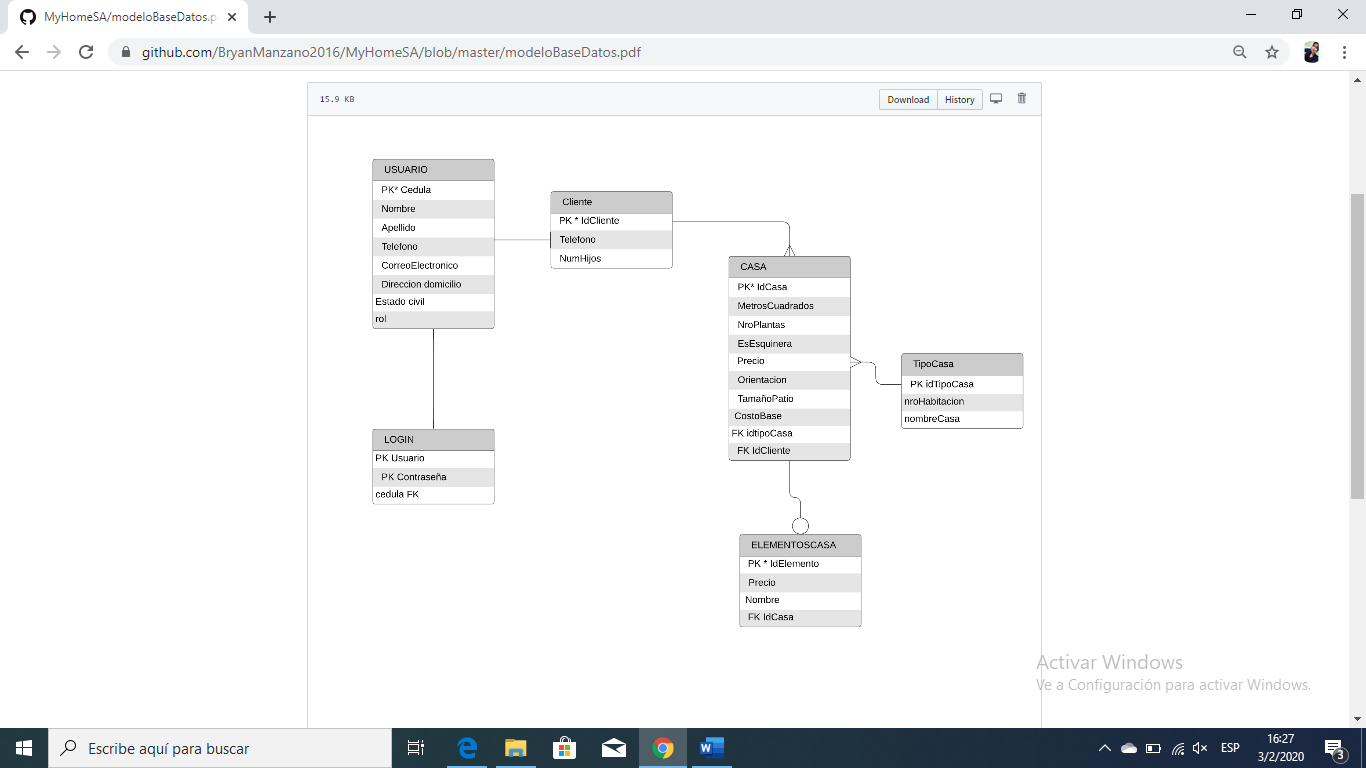
## **Iterator**

Este patrón de diseño es comúnmente usado para para poder acceder a los elementos que presenta un objeto de forma secuencial, en este caso se uso este patrón de diseño debido a que se requiere conocer los elementos que vaya a presentar las casas cada vez que un usuario lo requiera, además de que este patrón permite pasar una colección de objetos de manera definida, lo cual es útil para aplicaciones en la que interactúan clientes ya que se pueden realizar diversas operaciones en los elementos.

## **Decorator**

Este patrón de diseño nos permite agregar de manera dinámica funcionalidades y comportamientos a un objeto sin que este afectando el comportamiento que tengan los otros objetos de la misma clase. Por lo cual se lo usó para agregar los elementos a las casas sin que se modifiquen o altere a las casas que ya existen.

## Modelo Base de datos



## Pruebas unitarias

Resulta que el valor esperado para la línea 50 es true, así que se hará el respectivo cambio para que pasen todas las pruebas quedando así:

