Escuela Colombiana de Ingeniería Julio Garavito

Andrés Cardozo

Tulio Riaño

Laboratorio 04

Programacion Orientada a Objetos

FECHA:

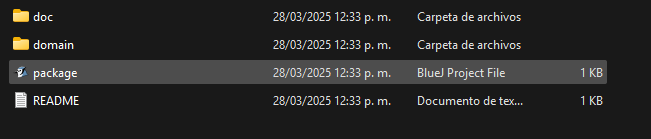
5/04/2025

# Unidades

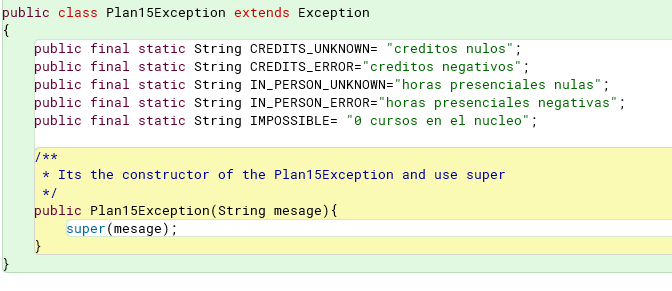
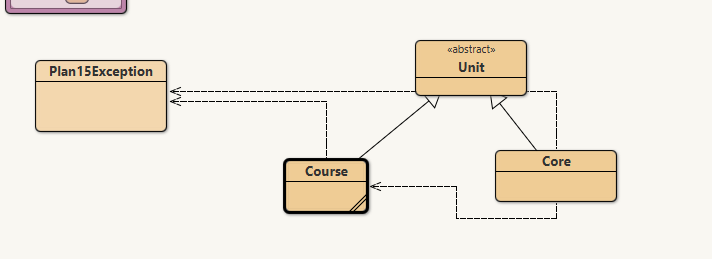
PRACTICANDO MDD y BDD con EXCEPCIONES

En este punto vamos a aprender a diseñar, codificar y probar usando excepciones. Para esto se van a trabajar algunos métodos de la clase Core

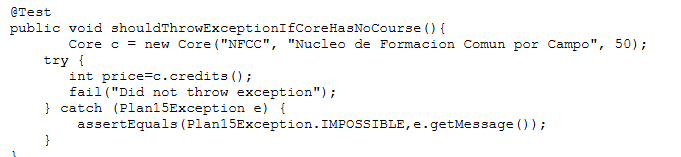
1. En su directorio descarguen los archivos contenidos en units.zip revisen el contenido y estudien el diseño estructural de la aplicación (únicamente la zona en azul).

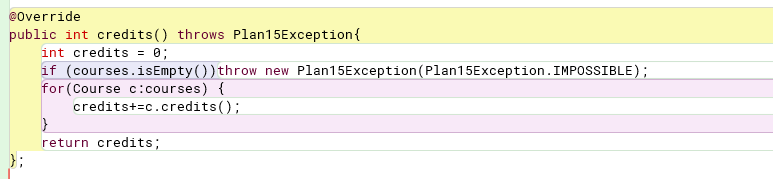
En el directorio se encuentra el paquete de dominio y doc. En el dominio encontramos las clases Core, Course y Unit

1. Expliquen por qué el proyecto no compila. Realicen las adiciones necesarias para lograrlo.

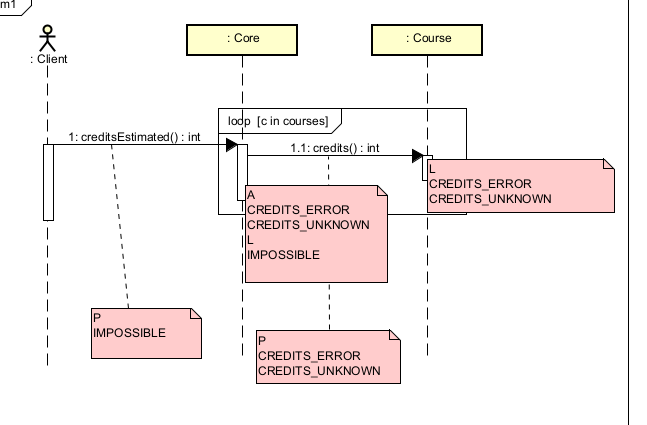
Falla porque falta la clase Plan15Exception, la cual es la encargada de nuestras excepciones. Por lo que tambien definimos como staticas el nombre de las excepciones.

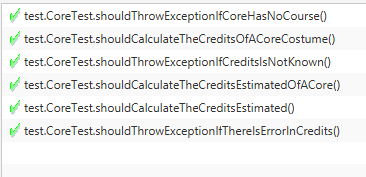
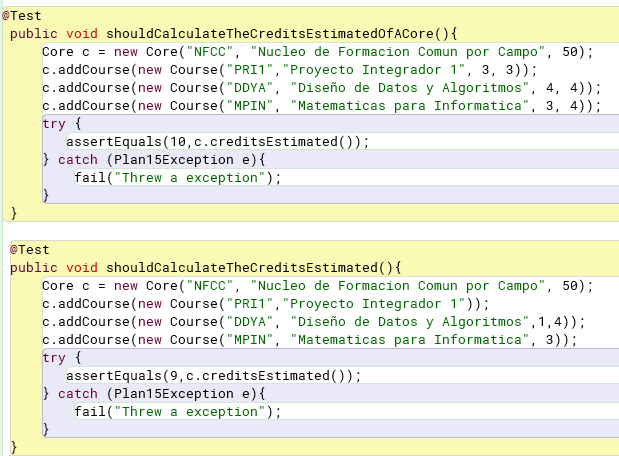
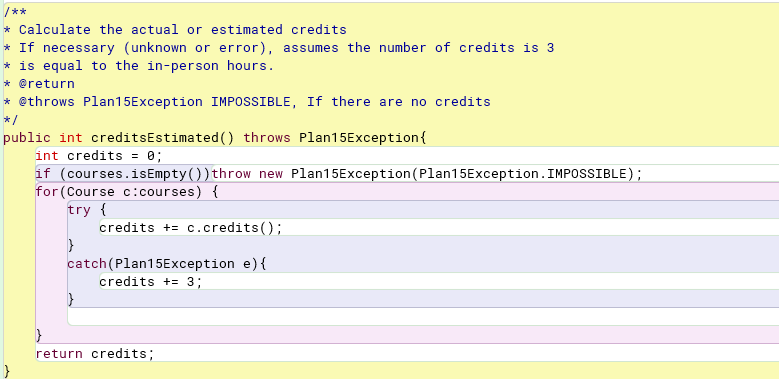
1. Dado el diseño y las pruebas documenten y codifiquen el método credits().

Por las pruebas vemos que, si un Core no tiene clases, debería salir la excepción IMPOSSIBLE

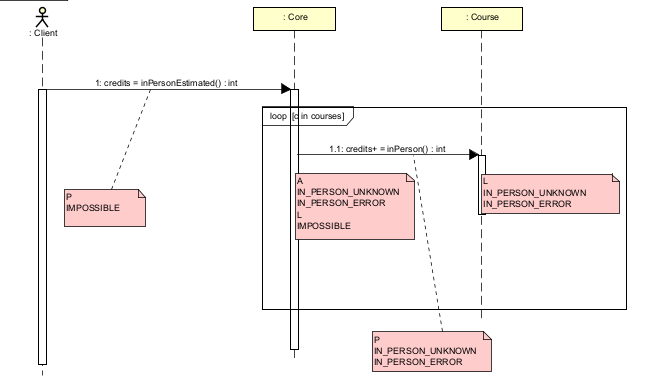
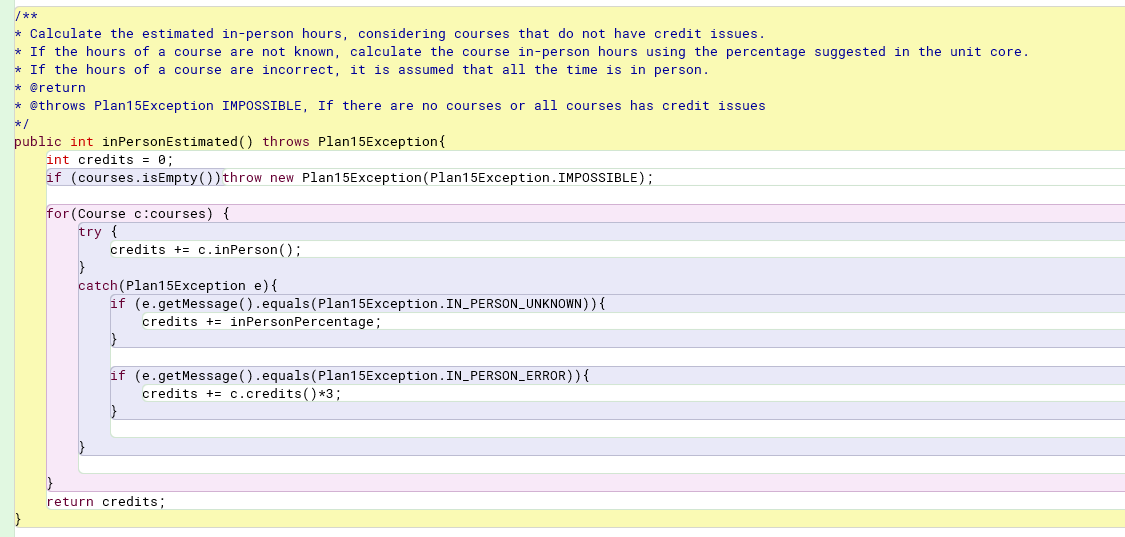


1. Dada la documentación y el diseño, codifiquen y prueben el método creditsEstimated().





1. Documenten, diseñen, codifiquen y prueben el método inPersonEstimated().

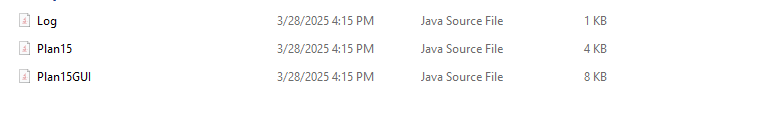


# En consola

## Conociendo el proyecto Plan15

No olviden respetar los directorios bin docs src

1. En su directorio descarguen los archivos contenidos en plan15.zip, revisen el contenido ¿Cuántos archivos se tienen? ¿Cómo están organizados? ¿Cómo deberían estar organizados?

Al descargar los archivos contenidos en plan15.zip encontramos tres archivos .java, no se encuentran organizados ya que Plan15GUI es un archivo que debería estar en el paquete de presentación, mientras que Log y Plan15 deberían estar en Dominio.

1. Estudien el diseño del programa: diagramas de paquetes y de clases. ¿cuántos paquetes tenemos? ¿cuántas clases tiene el sistema? ¿cómo están organizadas? ¿cuál es la clase Ejecutiva?

La cantidad de paquetes que tenemos son dos (Presentation – Domain)

El sistema tiene cinco clases asociadas.

En el paquete de dominio se encuentran clases como Core-Course-Plan15-Unit, mientras que en javac -d bin -sourcepath src src/domain/\*.java presentación esta Plan15GUI.

La clase ejecutiva es Plan15GUI que se encuentra en el paquete de presentación.

1. Prepare los directorios necesarios para ejecutar el proyecto. ¿qué estructura debe tener? ¿qué clases deben tener? ¿dónde están esas clases? ¿qué instrucciones debe dar para ejecutarlo?

El nombre del directorio general va a ser Plan15 este va a contener los directorios src, bin y docs. Las clases que debe tener src son las que estan asociadas a los paquetes de presentation,domain y test

Las primeras instrucciones antes de ejecutarlo sería utilizar javac para compilar los respectivos archivos y generar todos los .class

javac -d bin -sourcepath src src/presentation/\*.java

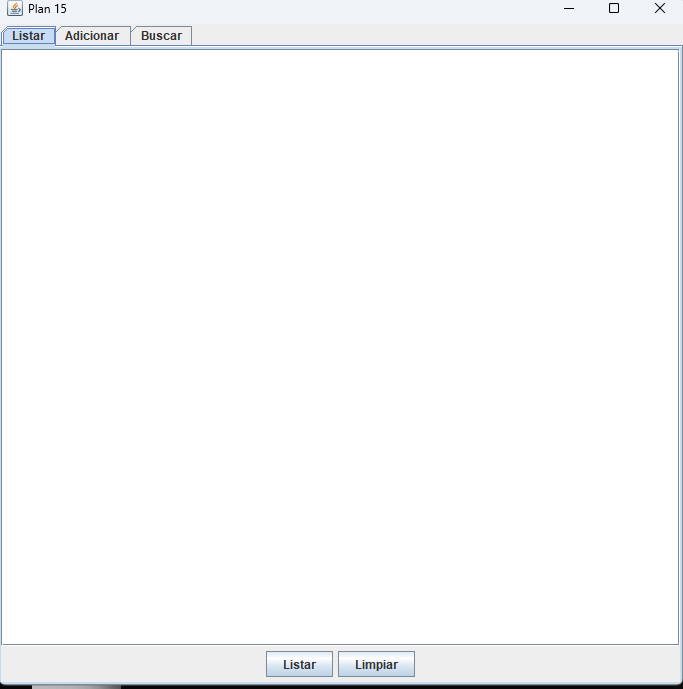
javac -d bin -sourcepath src src/domain/\*.java

java –cp bin presentation.Plan15GUI - Esta última linea de comando ejecutara el programa.

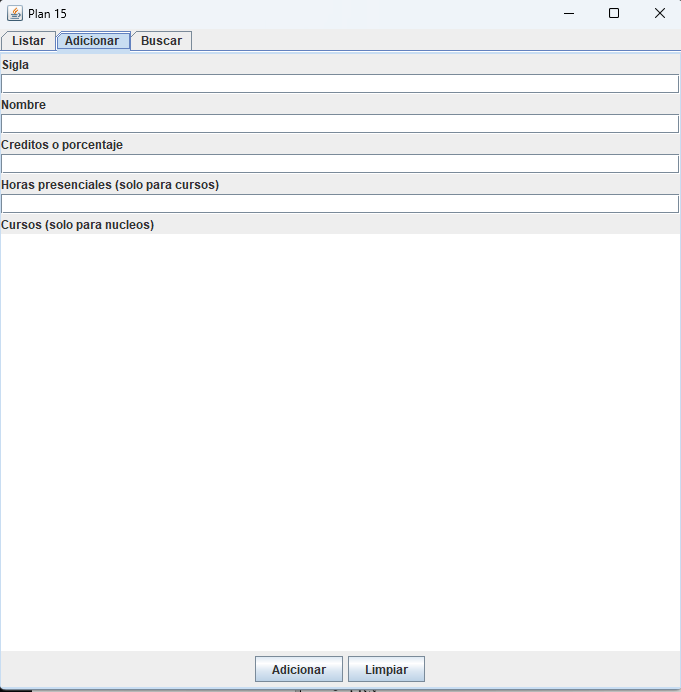
Y para generar la documentacion es:

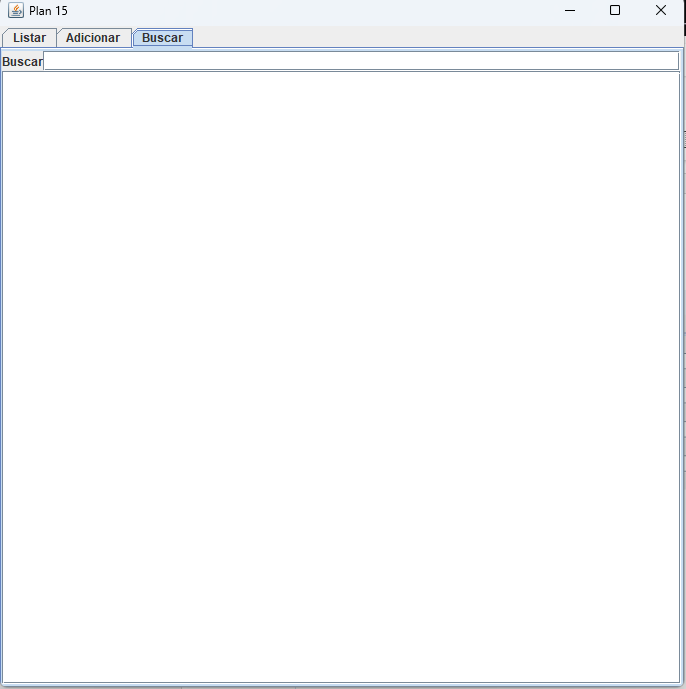
* javadoc -d docs -cp src -subpackages presentation domain

1. Ejecute el proyecto, ¿qué funcionalidades ofrece? ¿cuáles funcionan?



Tienen dos botenes que son Listar y Limpiar, también puede adicionar en otra pestaña donde tiene para ingresar datos como sigla, nombre, créditos o porcentaje, horas presenciales (solo para cursos) cursos (solo para nucleos) y otra pestaña que es buscar que permitirá hacer eso.

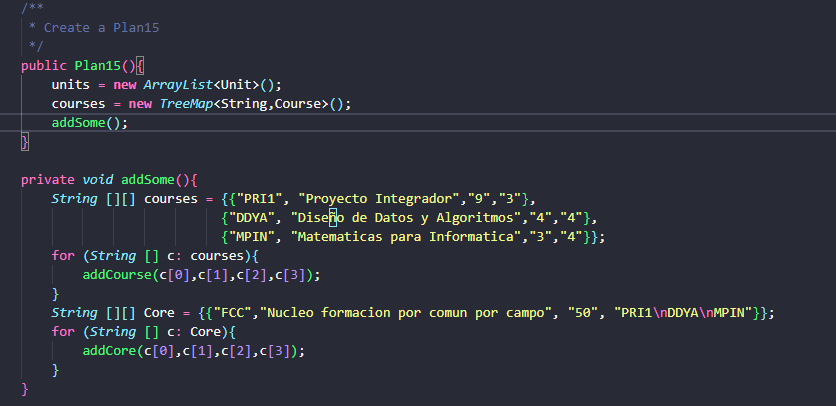




1. Revisen el código y la documentación del proyecto. ¿De dónde salen las unidades iniciales?¿Qué clase pide que se adicionen? ¿Qué clase los adiciona?

Las unidades iniciales salen del método addSome() en esta se encuentran varios courses y cores.

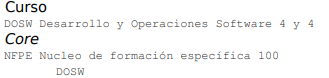
La clase que pide que se adicionen es Plan15 y la clase que los adiciona es la mismo desde el constructor.



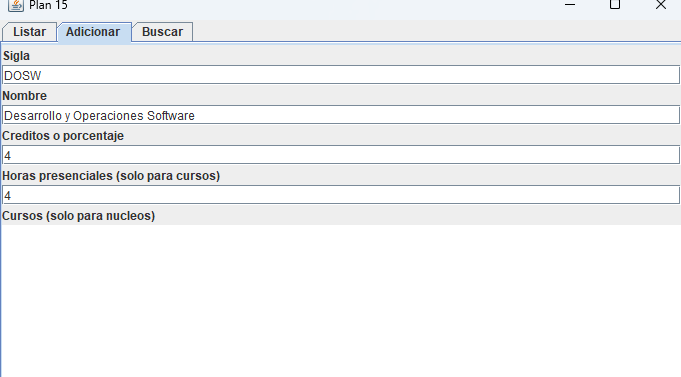
## Adicionar y listar. Todo OK

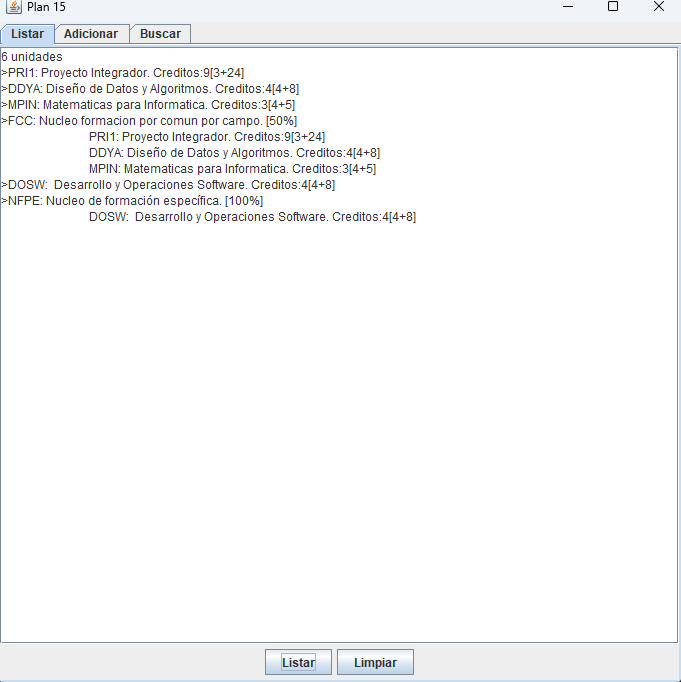
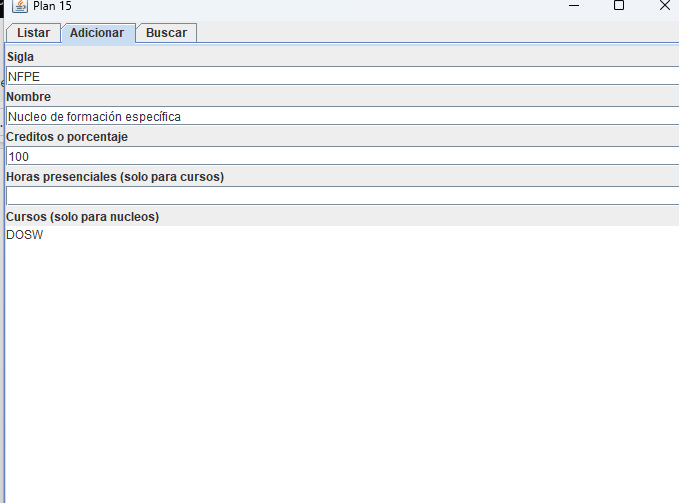
(NO OLVIDEN BDD - MDD)

El objetivo es realizar ingeniería reversa a las funciones de adicionar y listar.

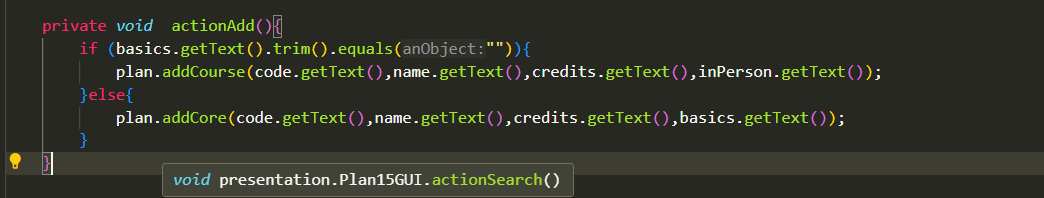
1. Adicionen un nuevo curso y un nuevo nucleo

¿Qué ocurre? ¿Cómo lo comprueban? Capturen la pantalla. ¿Es adecuado este comportamiento?

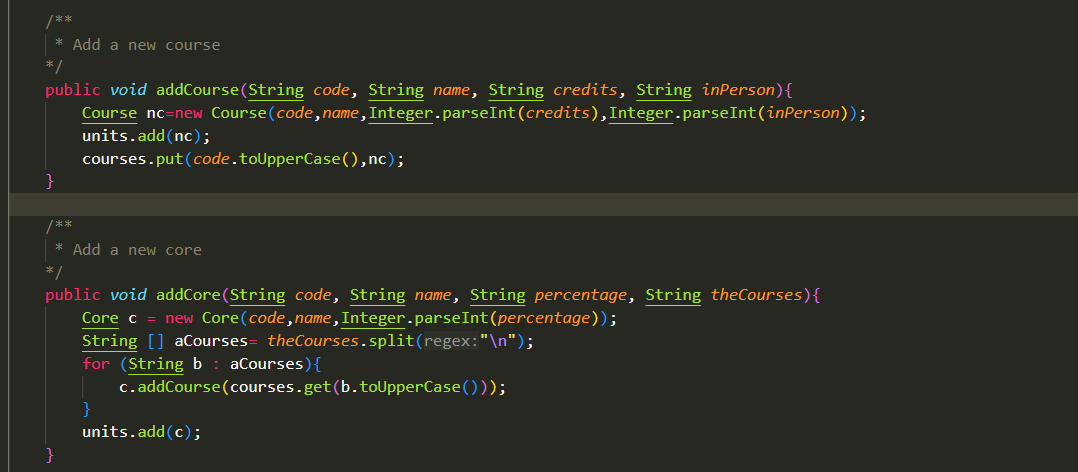


En la pestaña de listar se pueden observar que se añadieron las correspondientes courses y cores esto se comprueba en esta pestaña, pero el comportamiento en este momento no es adecuado ya que en caso de que por error adicionemos dos veces un curso lo va a permitir, pero esto es incorrecto.

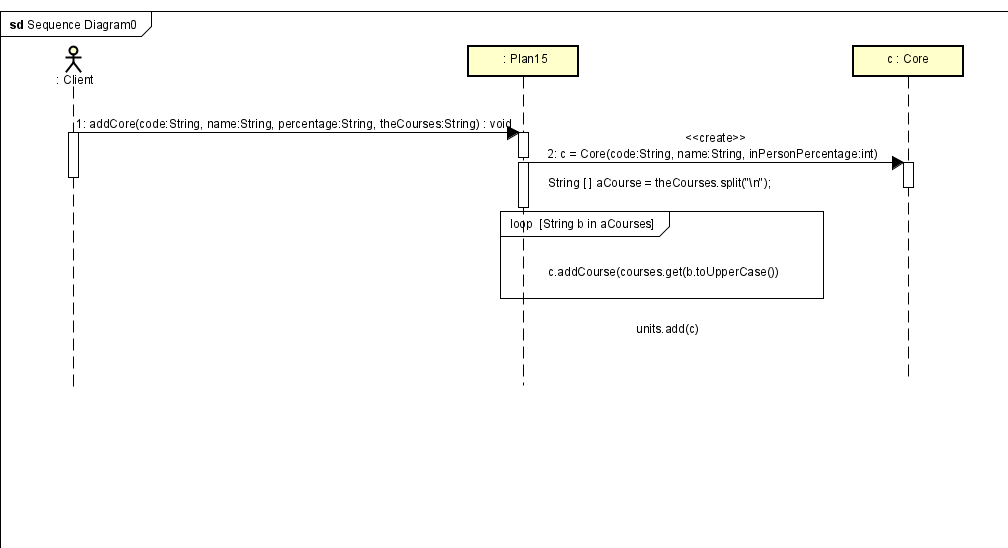
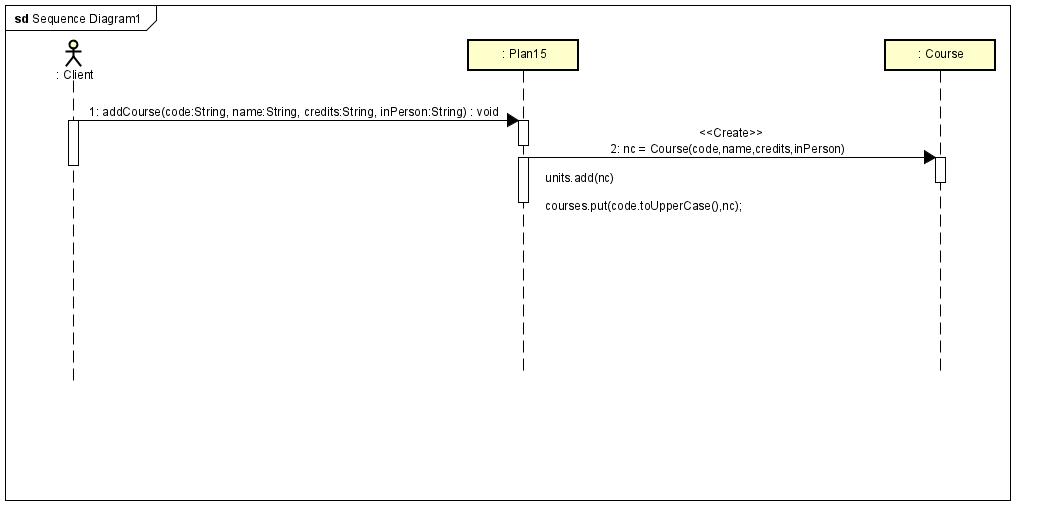
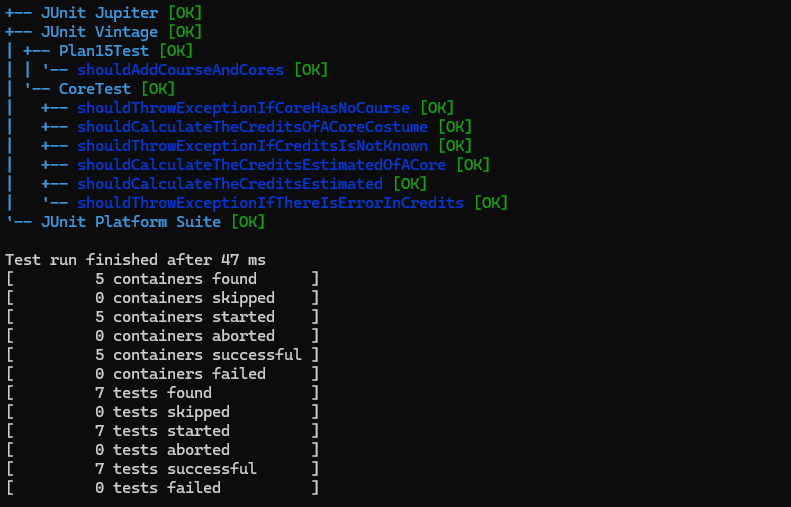
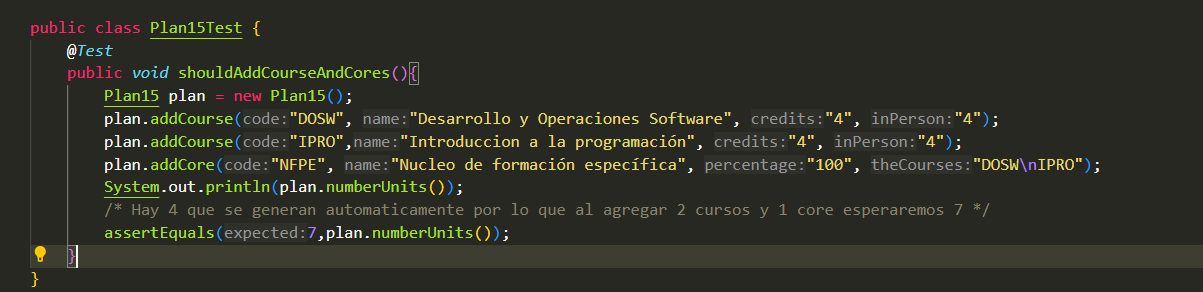
1. Revisen el código asociado a adicionar en la capa de presentación y la capa de dominio. ¿Qué método es responsable en la capa de presentación? ¿Qué método en la capa de dominio?

El método action add es el responsable en la capa de presentación.

Por otra parte, los métodos addCourse, addCore son los encargados en la capa de dominio

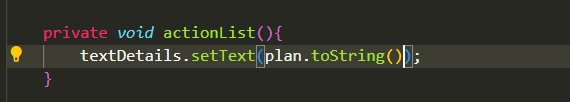


1. Realicen ingeniería reversa para la capa de dominio para adicionar. Capturen los resultados de las pruebas de unidad.

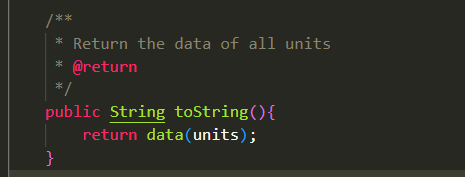


1. Revisen el código asociado a listar en la capa de presentación y la capa de dominio. ¿Qué método es responsable en la capa de presentación? ¿Qué método en la capa de dominio?

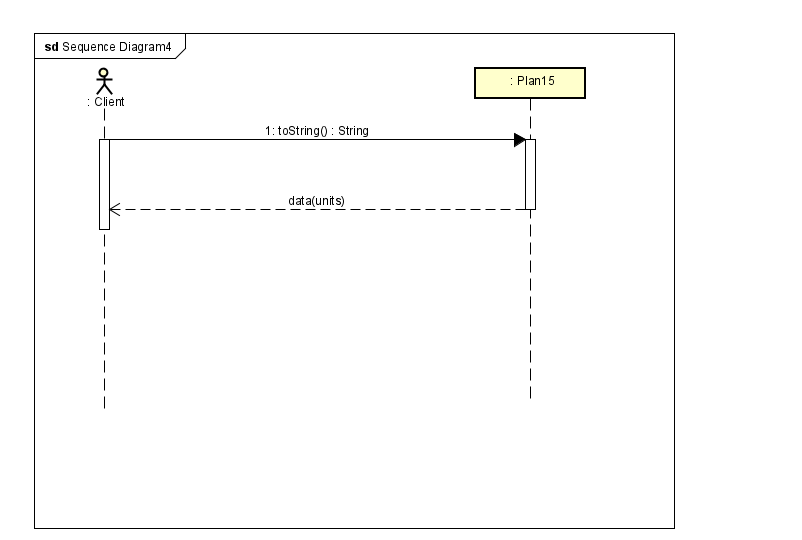
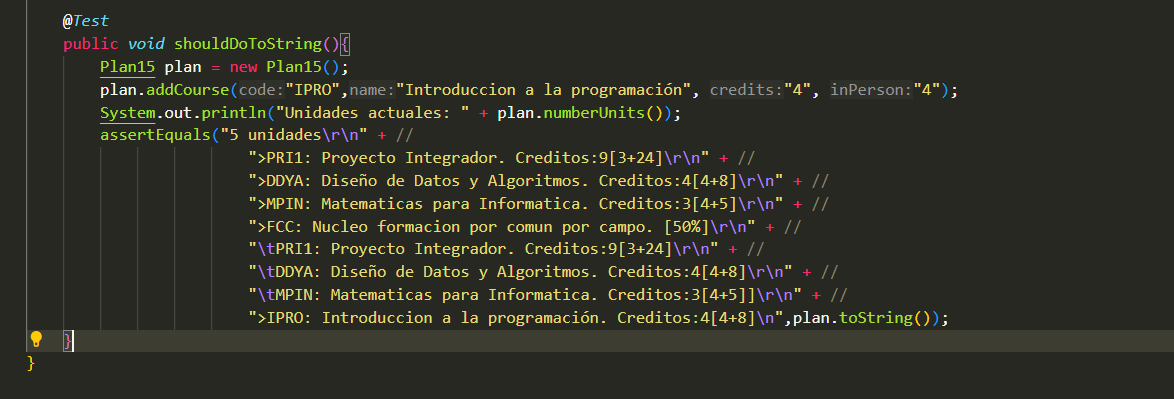
En la capa de presentación encontramos el método action list



En la capa de dominio encontramos el método toString()



1. Realicen ingeniería reversa para la capa de dominio para listar. Capturen los resultados de las pruebas de unidad.



1. Propongan y ejecuten una prueba de aceptación.



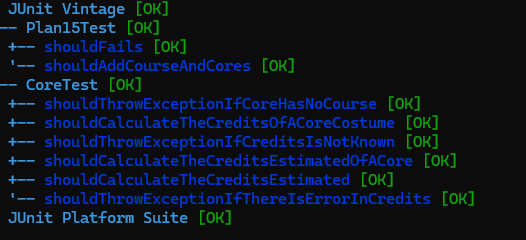
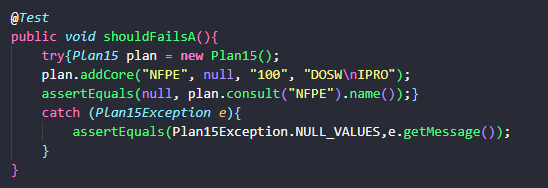
## Adicionar una unidad. Funcionalidad robusta

El objetivo es perfeccionar la funcionalidad de adicionar un curso para hacerla más robusta.

Para cada uno de los siguientes casos realice los pasos del 1 al 4.

a. ¿Y si el nombre de la unidad no existe?

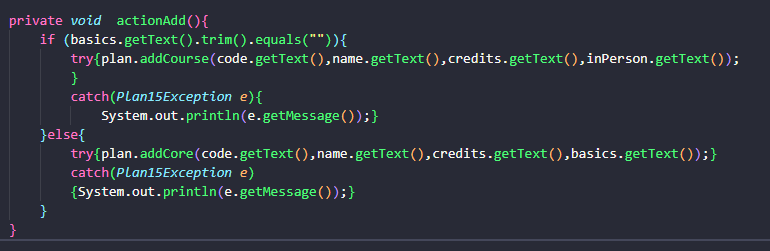
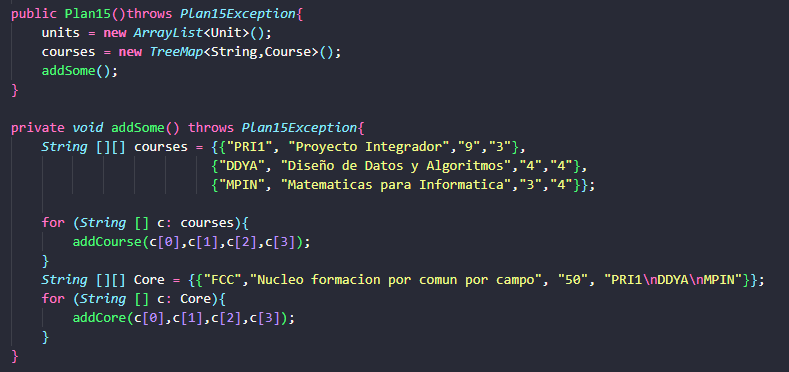
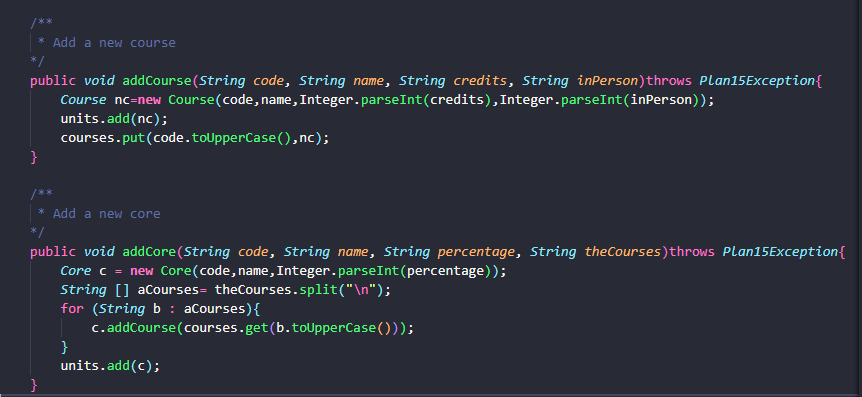
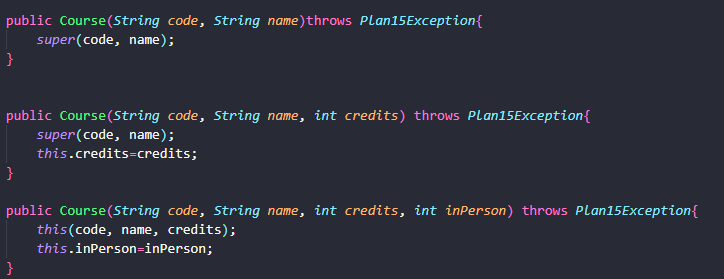
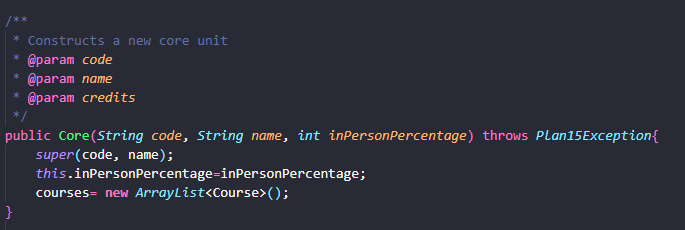
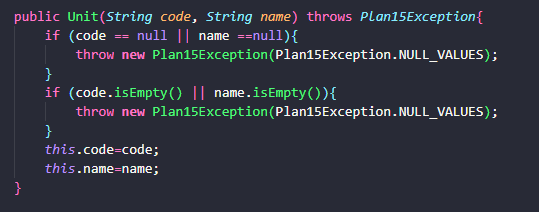
1. Propongan una prueba de aceptación que genere el fallo.

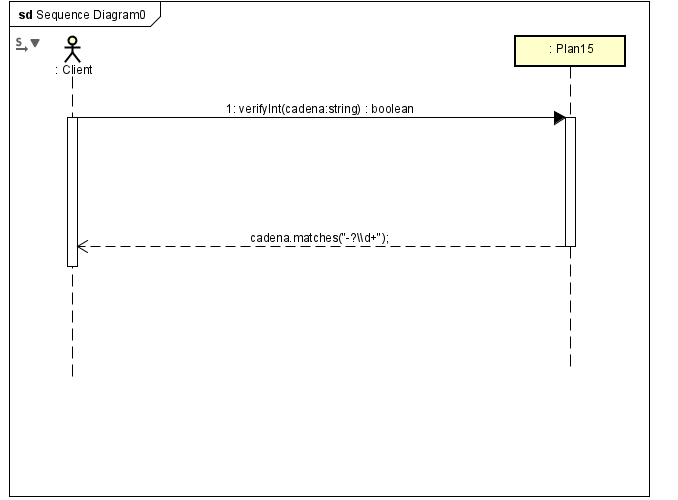
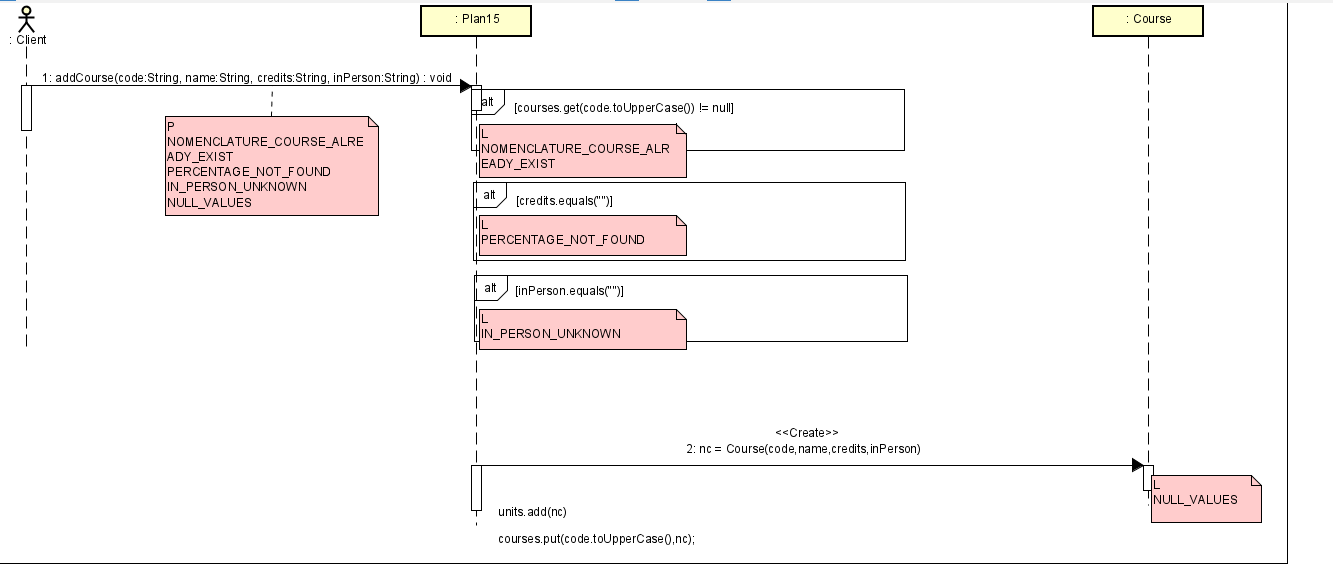
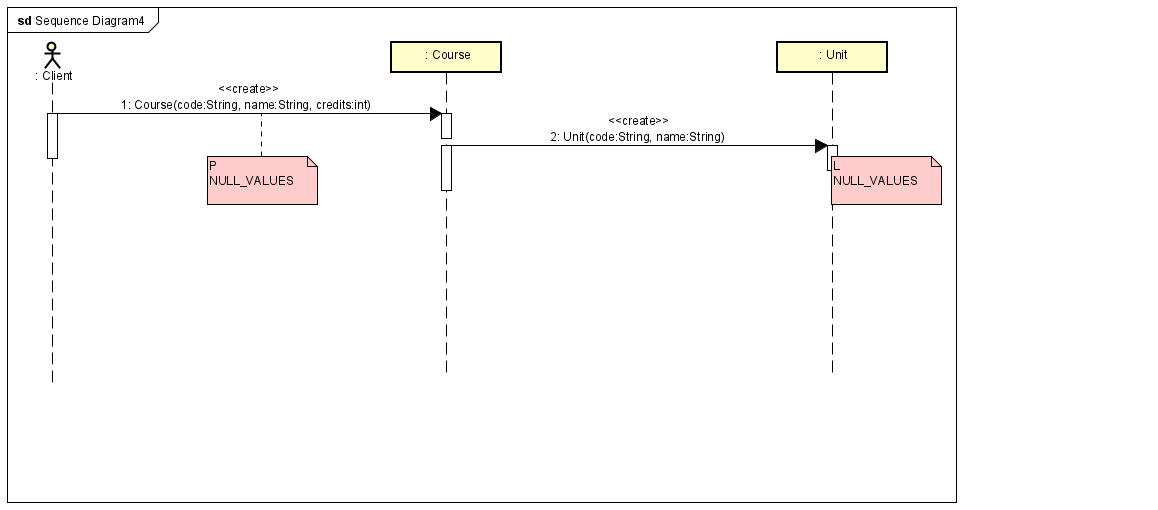
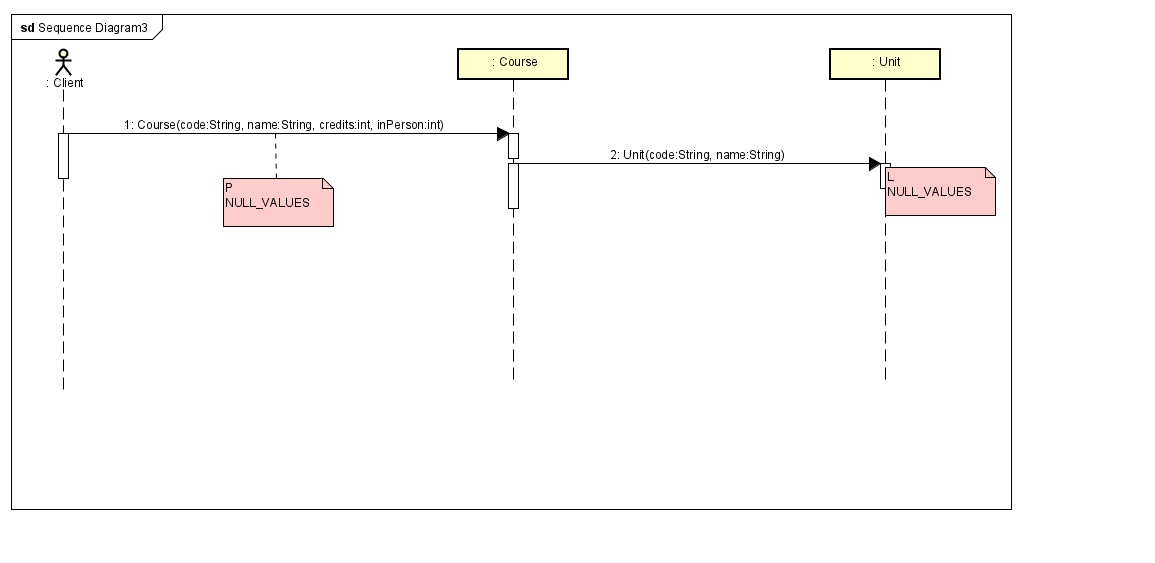
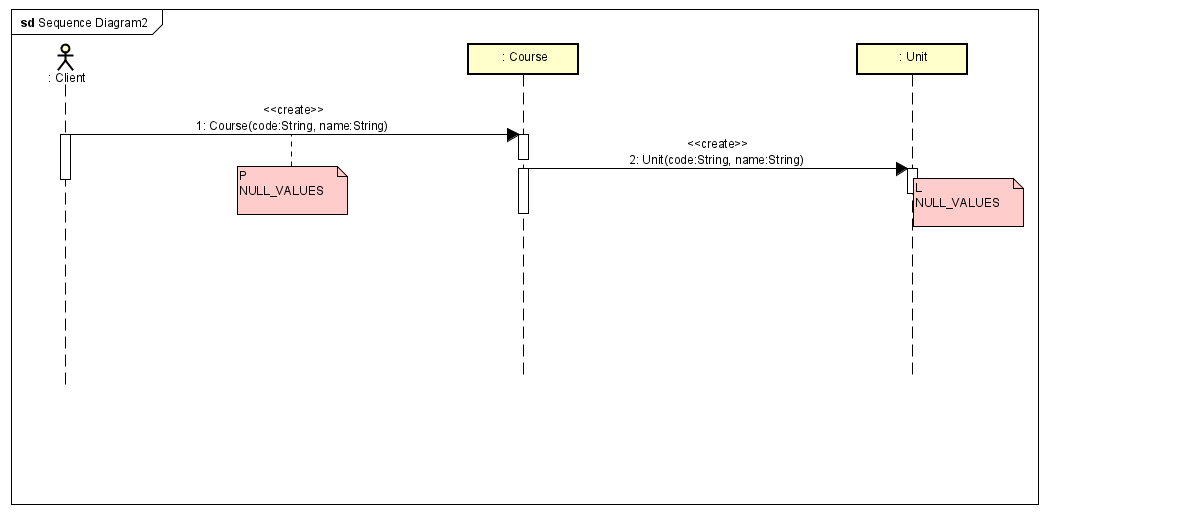


2. Analicen el diseño realizado. Para hacer el software robusto: ¿Qué método debería lanzar la excepción? ¿Qué métodos deberían propagarla? ¿Qué método debería atenderla? Explique Claramente.

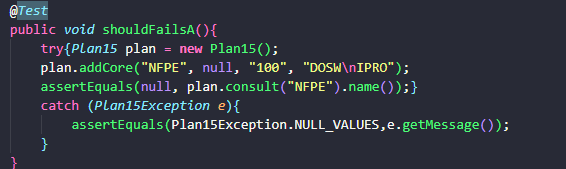
Los constructores son los que deberían lanzar la excepción. Se propagará desde unit hasta el plan15GUI y como es la superclase entonces sus respectivos constructores van a propagar esta excepción, al propagar estas todo lo que hace cosas como addCourse y addCore se debería propagar y Plan15 también la propagaría. Entonces, se va a atender en la interfaz para indicarle al usuario cual es el error.

3. Construya la solución propuesta. Capture los resultados de las pruebas de unidad.

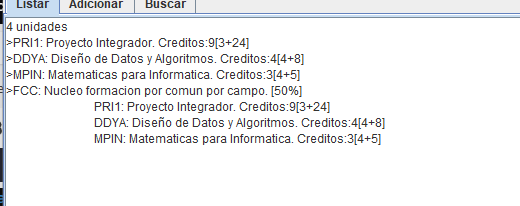
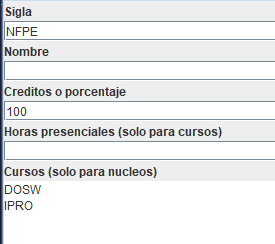


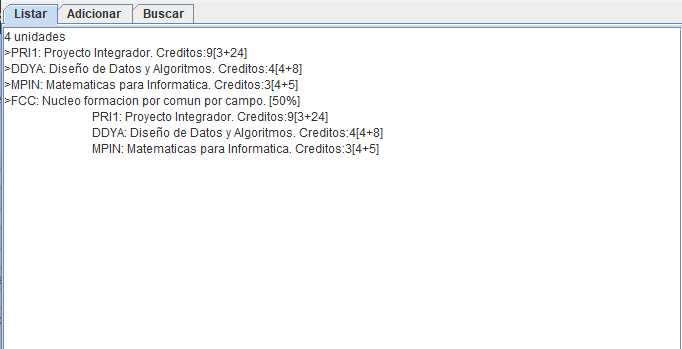


Y como estamos propagando hasta la GUI la prueba de unidad aun va a seguir fallando:



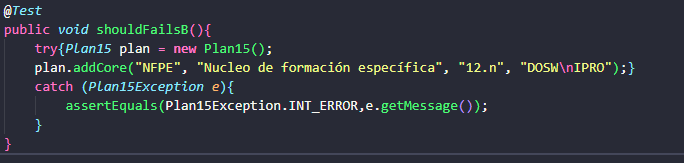
4. Ejecuten nuevamente la aplicación con el caso de aceptación propuesto en 1. ¿Qué sucede ahora? Capture la pantalla. Ahora nos lanza la excepción y no permite añadirlo si no se cumple esta condición.

Antes:  

Despues: 

b. ¿Y si los valores enteros no son enteros?

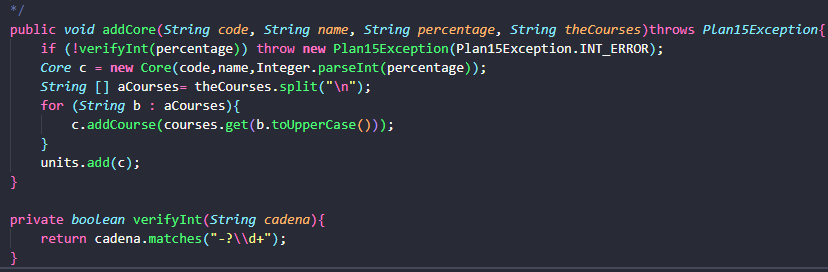
1. Propongan una prueba de aceptación que genere el fallo.

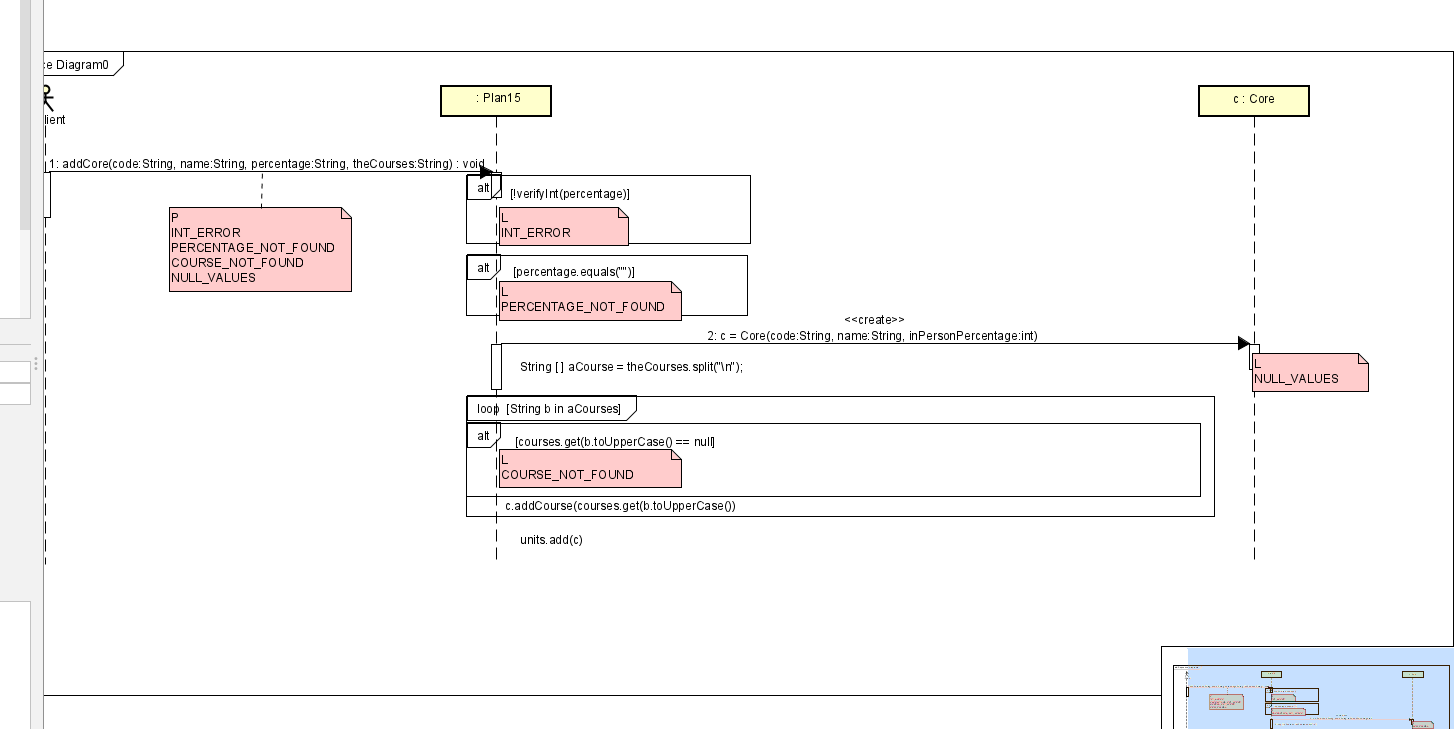


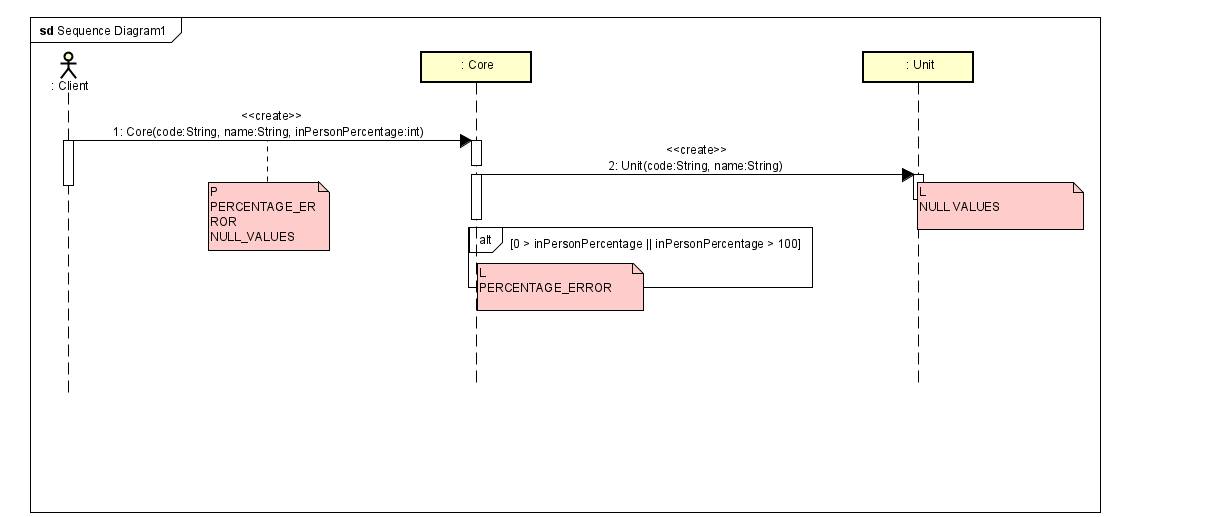
1. Analicen el diseño realizado. Para hacer el software robusto: ¿Qué método debería lanzar la excepción? ¿Qué métodos deberían propagarla? ¿Qué método debería atenderla? Explique claramente.

El método que la propaga es addCore y la lanza este mismo ya que vamos a verificar que mediante la funcion matches() de string si es un valor entero, en caso contrario lanzamos esta.

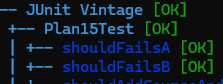
1. Construya la solución propuesta. Capture los resultados de las pruebas de unidad.



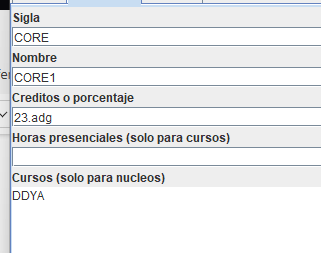
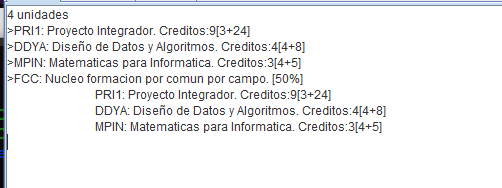
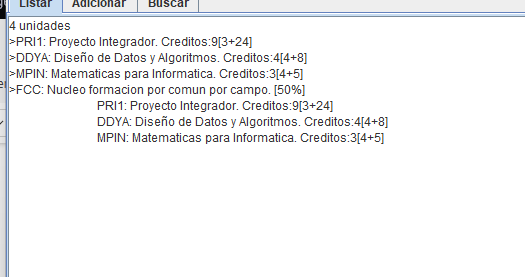




Buscamos una función que nos ayuda a verificar si es un entero (link 1)

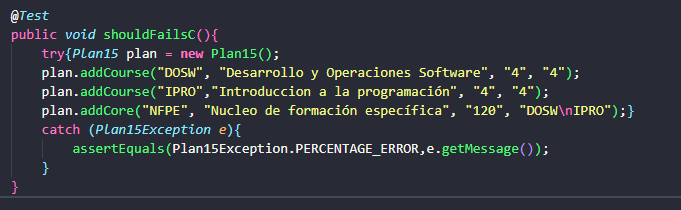


1. Ejecuten nuevamente la aplicación con el caso de aceptación propuesto en 1. ¿Qué sucede ahora? Capture la pantalla.

Antes: Despues: 

c. ¿Y si el portage no está entre 0 y 100?

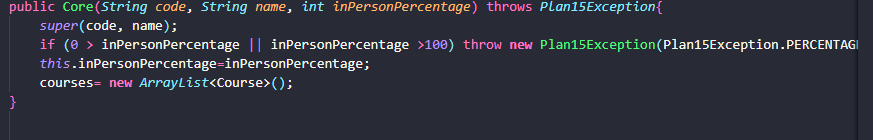
1. Propongan una prueba de aceptación que genere el fallo.

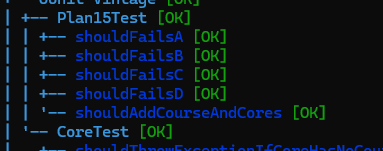
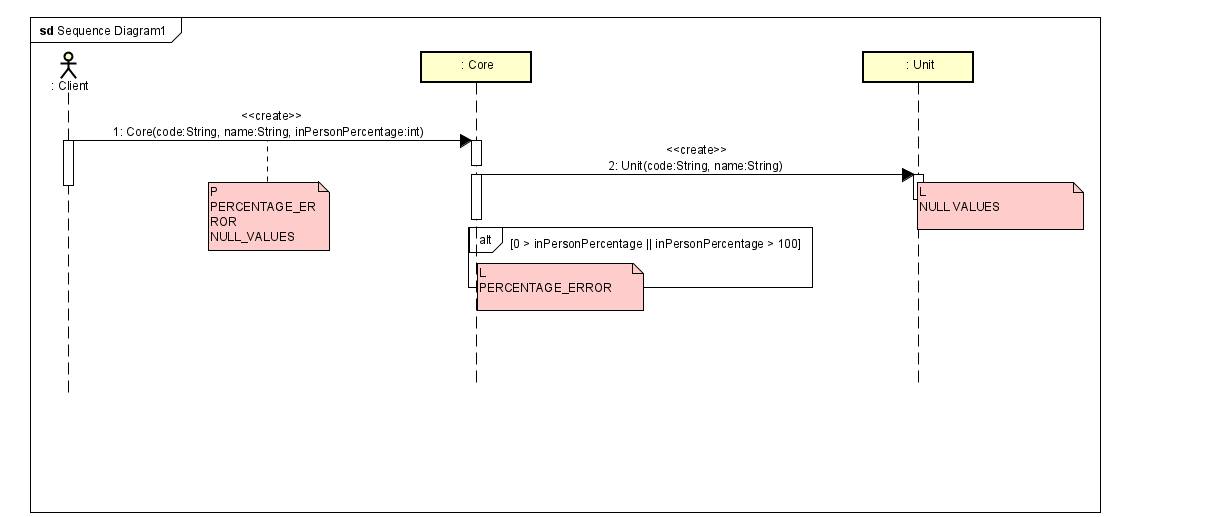


1. Analicen el diseño realizado. Para hacer el software robusto: ¿Qué método debería lanzar la excepción? ¿Qué métodos deberían propagarla? ¿Qué método debería atenderla? Explique claramente.

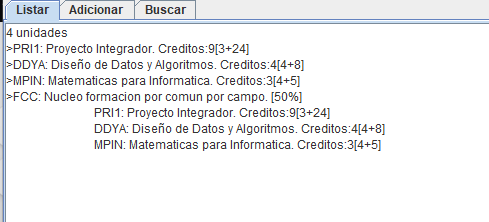
Nuevamente el método en el que se va a lanzar la excepción es addCore, buscamos verificar que el porcentaje estre entre 0 y 100.

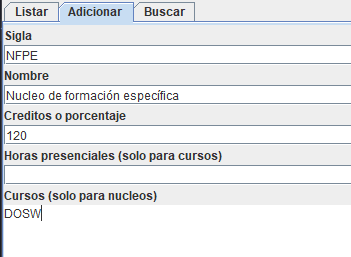
1. Construya la solución propuesta. Capture los resultados de las pruebas de unidad.

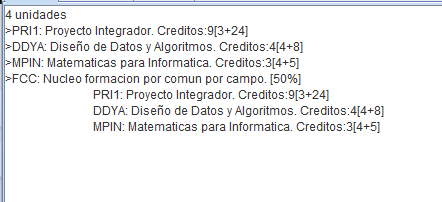




1. Ejecuten nuevamente la aplicación con el caso de aceptación propuesto en 1. ¿Qué sucede ahora? Capture la pantalla.

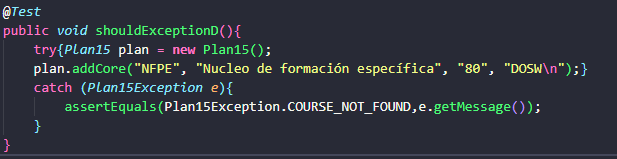
Antes: 



Despues:

d. Proponga una nueva condición: El código tampoco puede ser nulo. No puedes crear un Core si no existe el curso.

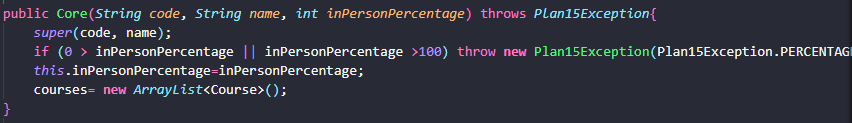
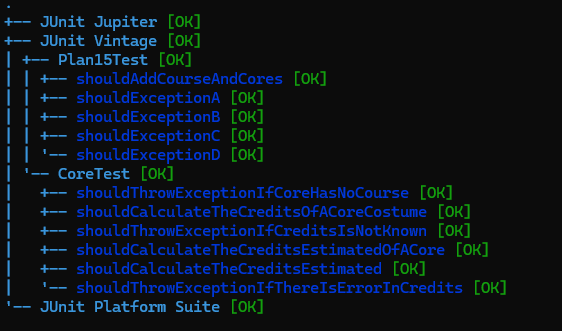
1. Propongan una prueba de aceptación que genere el fallo.



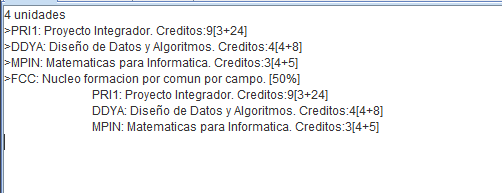
1. Analicen el diseño realizado. Para hacer el software robusto: ¿Qué método debería lanzar la excepción? ¿Qué métodos deberían propagarla? ¿Qué método debería atenderla? Explique claramente.

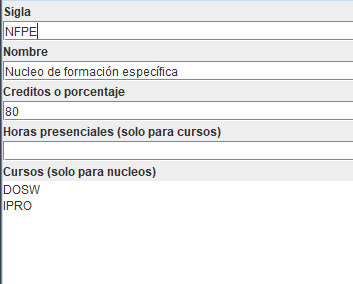
El constructor de core debería de lanzar esta excepción, la propagarían hasta Plan15 y se atendería en actionAdd en Plan15GUI.

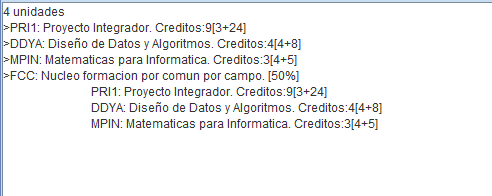
1. Construya la solución propuesta. Capture los resultados de las pruebas de unidad.

Les cambiamos los nombres:  


1. Ejecuten nuevamente la aplicación con el caso de aceptación propuesto en 1. ¿Qué sucede ahora? Capture la pantalla.







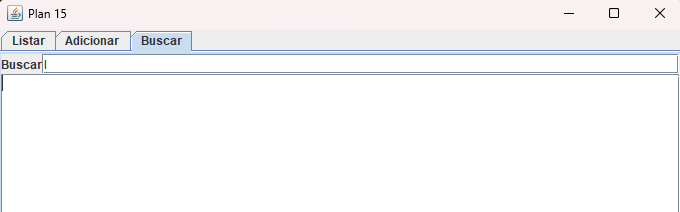


## Consultando por patrones. ¡No funciona y queda sin funcionar!

(NO OLVIDEN BDD - MDD)

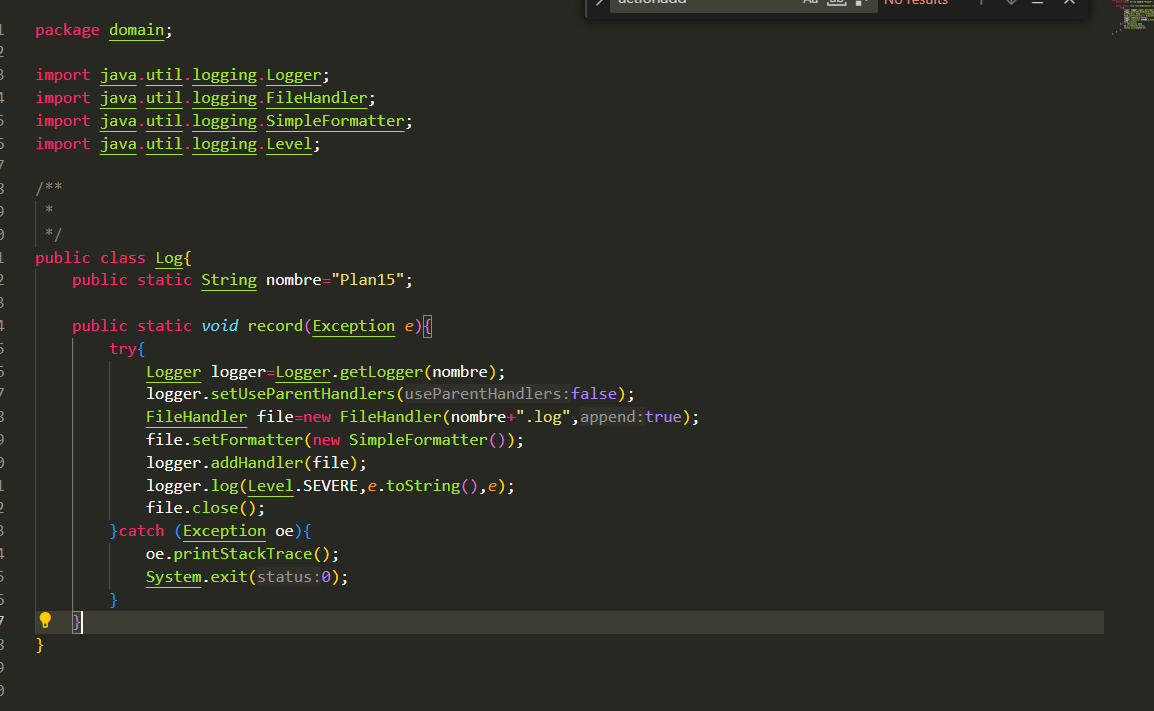
1. Consulten una unidad que inicie con I. ¿Qué sucede? ¿Qué creen que pasó? Capturen el resultado. ¿Quién debe conocer y quien NO debe conocer esta información?

Al consultar una unidad que inicia con I, en la consola se lanza varios errores referentes a que una lista se salio de rango, esta información la debe conocer los desarrolladores que son quienes van a manejarlos mientras que los usuarios no deberían observa esta información. Se debe presentar un mensaje indicando que hubo un error y preguntándole al usuario si lo quiere enviar o no para guardarlo en log.



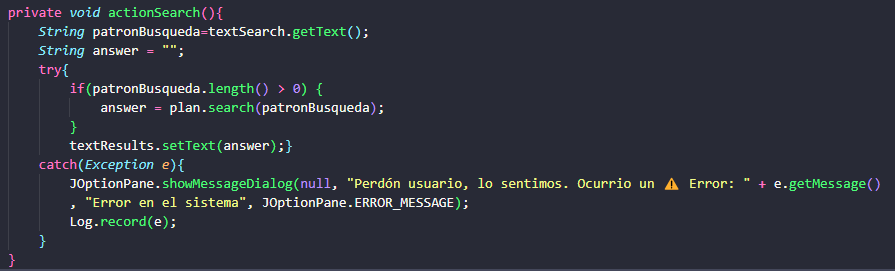
2. Exploren el método record de la clase Log ¿Qué servicio presta?

Este método ofrece el servicio de registrar errores del sistema en un archivo que puede revisarse después llamado Plan15.log en caso de que algo salga mal entraría al catch donde se imprima la pila de rastreo (trayectoría de llamadas que condujeron a la excepción) y luego terminaría el programa (“System.exit(0);”).



3. Analicen el punto adecuado para que EN ESTE CASO se presente un mensaje especial de alerta al usuario, se guarde la información del error en el registro y continúe la ejecución. Expliquen y construyan la solución.

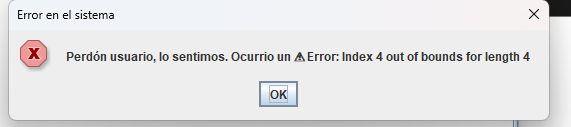
Esto se puede lograr mediante el método de record que adjuntamos en el punto anterior, esto lo podemos utilizar en las pruebas y en el momento en que entre en el catch se registre con un mensaje correspondiente y un aviso al usuario.

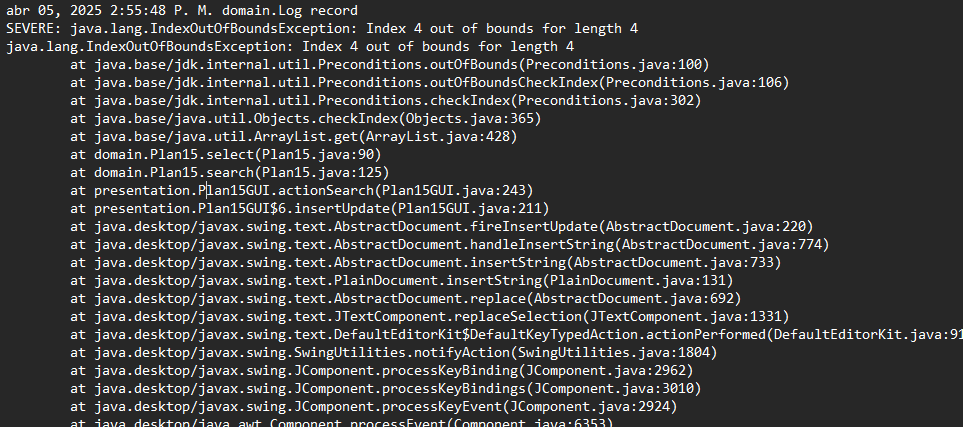
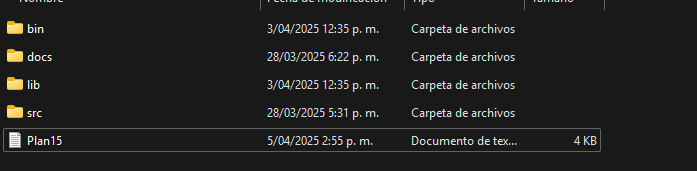


4. Ejecuten nuevamente la aplicación con el caso propuesto en 1. ¿Qué mensaje salió en

pantalla? ¿La aplicación termina? ¿Qué información tiene el archivo de errores?

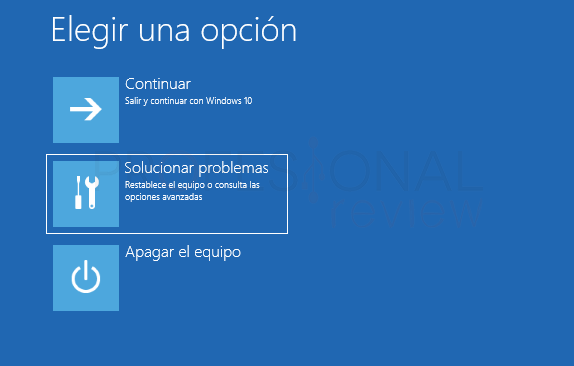
El mensaje que sale en la pantalla es el que determinamos cada vez que se va a registrar en el log, la aplicación no termina ya que no se presenta ningún problema a la hora de registrar el error en el archivo Plan15.log. La información que tiene este archivo son las respectivas excepciones que le salían al usuario y no debería verlas.





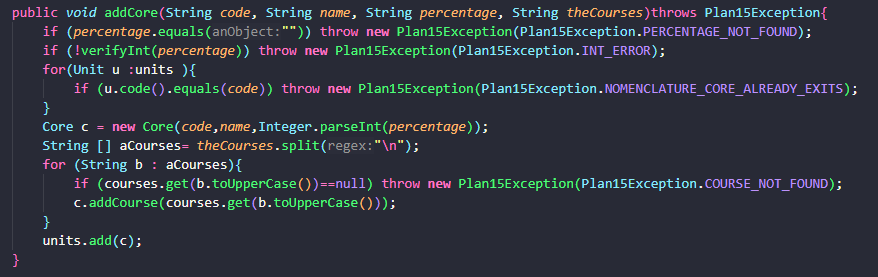
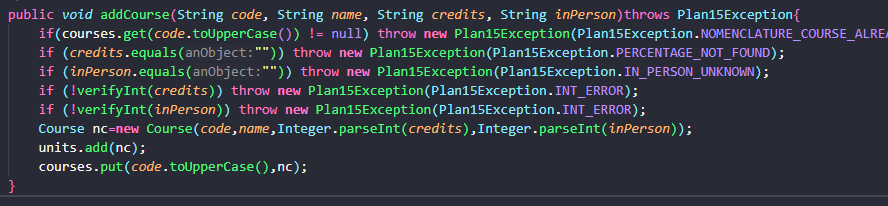
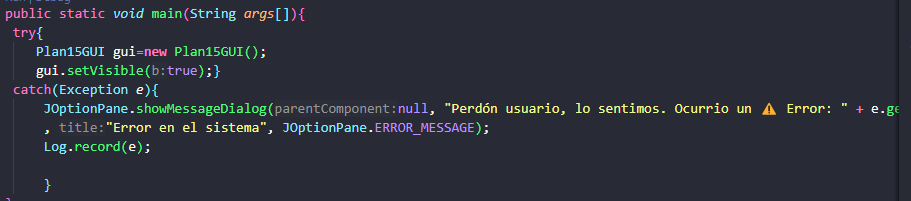
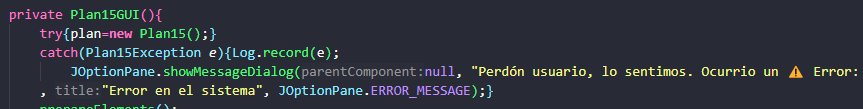
5. ¿Es adecuado que la aplicación continúe su ejecución después de sufrir un incidente como este? ¿de qué dependería continuar o parar?

No es adecuado que la aplicación continue su ejecución tras un incidente como este. Sin embargo, esto dependería ya que se podría presentar la opción al usuario de enviar un reporte o volver a intentar esta acción.   
Por ejemplo:



Todo lo anterior depende de la capacidad de la aplicación de recuperarse frente al tipo de error que se presente. Si daña la experiencia del usuario o si daña datos del sistema es mejor cerrarla.

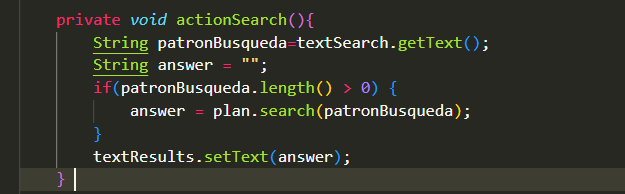
6. Modifiquen la aplicación para garantizar que SIEMPRE que haya un error se maneje de forma adecuada. ¿Cuál fue la solución implementada?

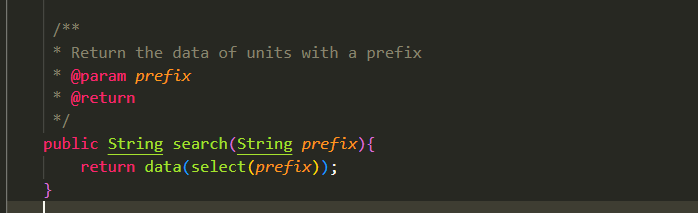
Al momento de hacer varios casos de pruebas añadimos más excepciones para cubrir casos que podrían surgir al momento de añadir un Curso o un Core:  
Entonces si queremos garantizar que siempre se maneje de forma adecuada usamos la clase log la cual nos va a informar sobre errores que nosotros no hemos tenido en cuenta:Asi si el error es grave se cerrara la aplicación y le informara al usuario.

## Consultando por patrones. ¡Ahora si funciona!

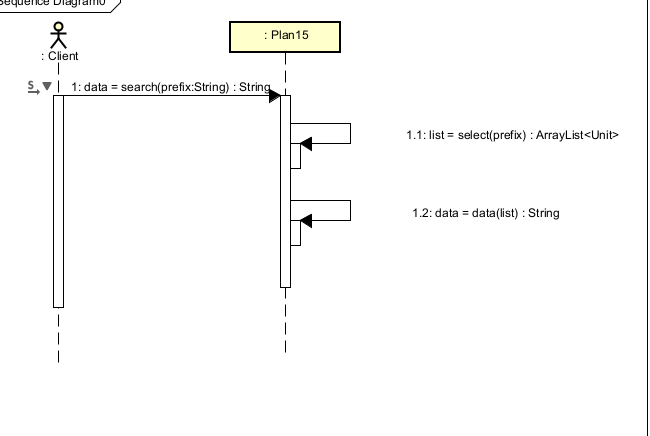
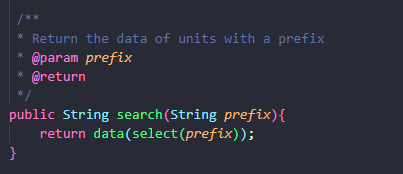
(NO OLVIDEN BDD - MDD)

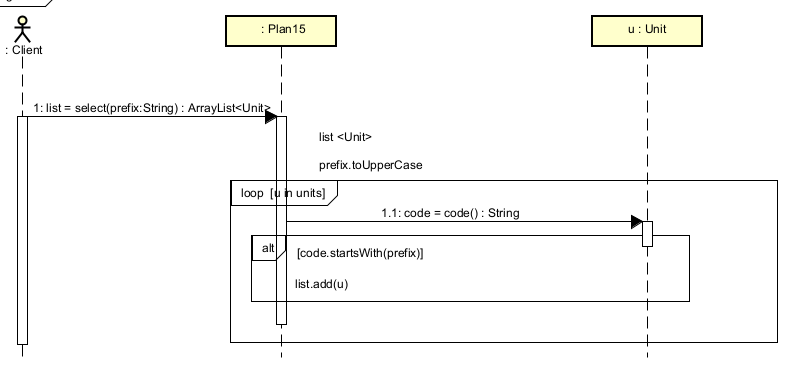
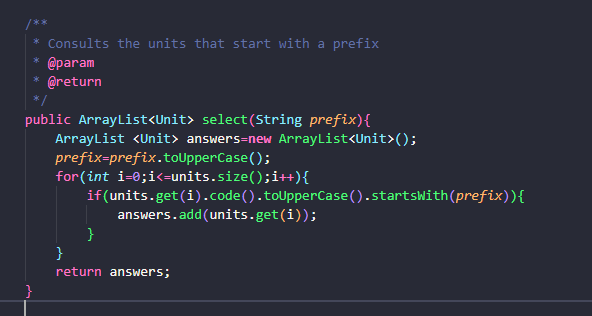
1. Revisen el código asociado a buscar en la capa de presentación y la capa de dominio. ¿Qué método es responsable en la capa de presentación? ¿Qué método es responsable en la capa de dominio?

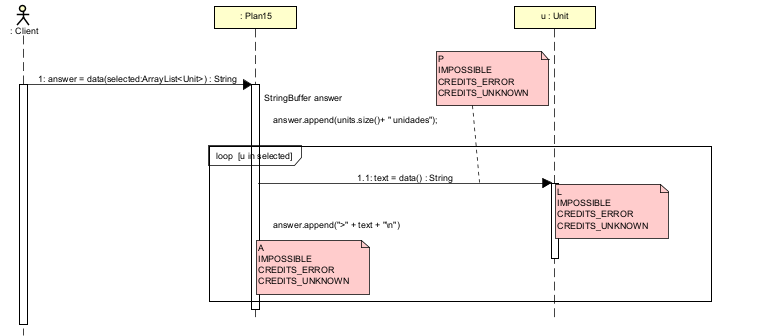
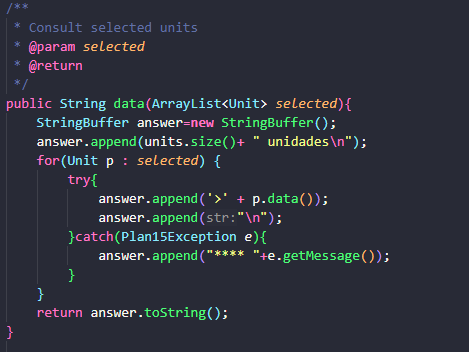
El método que es responsable en la capa de presentación es actionSearch() Mientras que el método responsable en la capa de dominio es search(String prefix) que a su vez usa la funcion select y data



2. Realicen ingeniería reversa para la capa de dominio para buscar. Capturen los resultados de las pruebas. Deben fallar.



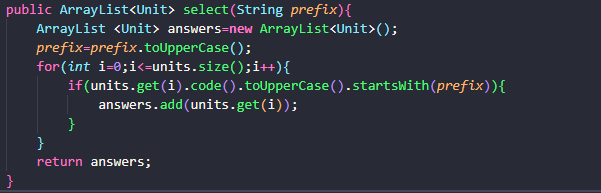




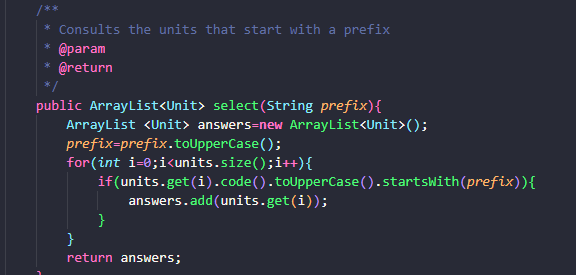


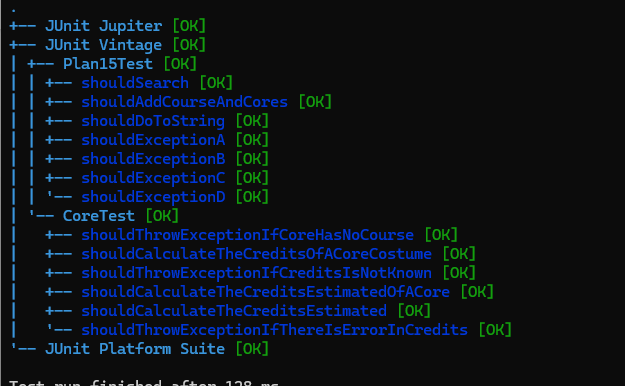
Y haciendo la prueba vemos que falla: 

3. ¿Cuál es el error? Soluciónenlo. Capturen los resultados de las pruebas.

El error es el metodo select(String prefix) de Plan15   
ya que esta mal la condicion del for i <= units.size y por eso es el error de out of bounds.

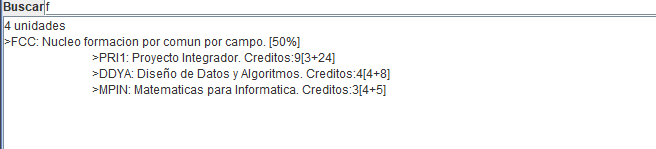
Para arreglarlo seria que sea un menor extricto o podriamos hacer un for each.





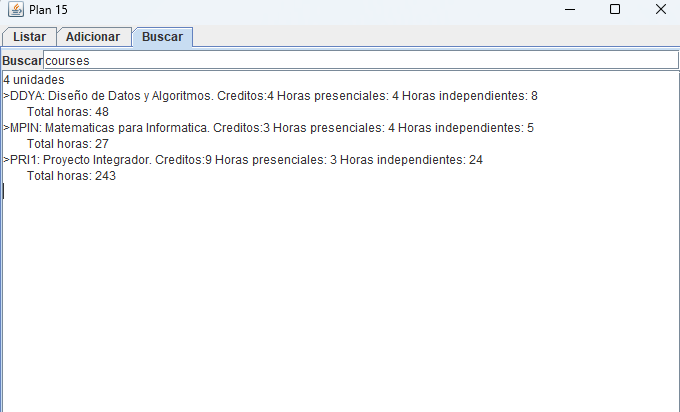
4. Ejecuten la aplicación nuevamente con el caso propuesto. ¿Qué tenemos en pantalla? ¿Qué información tiene el archivo de errores?

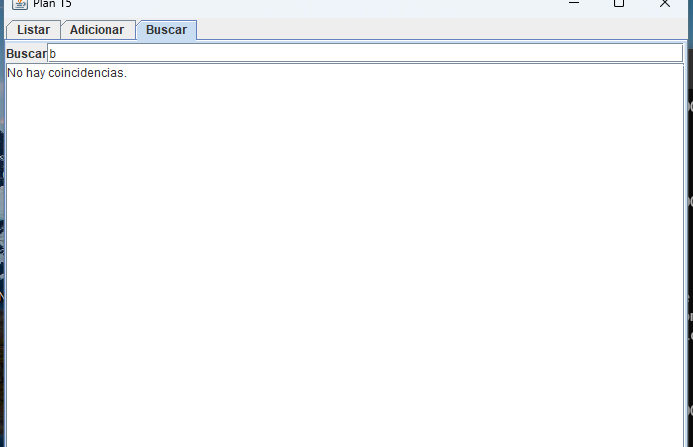
Antes en el archivo encontrábamos los errores de out of bounds y ahora no tendremos nada ya que no hay problema ahora.  
Con cada letra que buscamos aparece la búsqueda tipo sugerencia de Google que con cada letra va buscando.



5. Refactorice la funcionalidad para que sea más amable con el usuario. ¿Cuál es la propuesta? ¿Cómo la implementa?

Para que sea más amigable con el usuario vamos a hacer que si escribe Courses se liste todos los cursos que hay en Plan15 por lo que se verá como si estuviera listando. Así con información más detallada.

Lo implementamos usando la clase abstracta Unit para crear un nuevo tipo de formato para estas salidas sin afectar las que ya teníamos para las pruebas. Tambien añadimos mensajes emergentes cuando se equivocan en la entrada de datos. Ademas que si no encuentra coincidencias diga al usuario que no hay coincidencias.



# Retrospectiva

1. ¿Cuál fue el tiempo total invertido en el laboratorio por cada uno de ustedes? (Horas/

Hombre)

Andrés Cardozo: 20 Horas

Tulio Riaño: 20 Horas

2. ¿Cuál es el estado actual de laboratorio? ¿Por qué? (Para cada método incluya su

estado)

El estado actual del laboratorio es completo, ya que desarrollamos de manera acertada cada uno de los puntos propuestos incluyendo el bono.

3. Considerando las prácticas XP del laboratorio de hoy ¿por qué consideran que son

Importe?

Las prácticas XP que se presentaron son importantes ya que nos permite una mayor comprensión del código y de la manera en que este se escribe (respetando los respectivos estándares). Por otra parte, las pruebas nos permiten ver que lo que estamos desarrollando sea de manera correcta y no llegue a presentarse ningún inconveniente.

4. ¿Cuál consideran fue su mayor logro? ¿Por qué? ¿Cuál consideran que fue su mayor

problema? ¿Qué hicieron para resolverlo?

Nuestro mayor logro fue el manejo correcto de todos los tipos de excepciones que se podrian presentar en Plan15GUI, nuestro mayor problema fue interpretar la manera correcta en la que se estaba haciendo todo lo relacionado a los logs.

5. ¿Qué hicieron bien como equipo? ¿Qué se comprometen a hacer para mejorar los

Resultados?

De nuevo resaltamos la comunicación y disposición para desarrollar el laboratorio, así mismo mantenemos nuestro compromiso por la ayuda mutua que se evidencia en el desempeño.

6. ¿Qué referencias usaron? ¿Cuál fue la más útil?

Incluyan citas con estándares adecuados.

1. [String matches() Method in Java with Examples | GeeksforGeeks](https://www.geeksforgeeks.org/string-matches-method-in-java-with-examples/)
2. <https://learn.microsoft.com/es-es/windows/uwp/launch-resume/run-minimized-with-extended-execution>