UNIVERSIDAD TECMILENIO

Mauricio Estrada De la Garza - AL02976904
Igor Sung Min Kim Juliao - AL02829189
28/09/2023
Programación Orientada a Objetos
Reto 6

Reto 6: Implementación de los conceptos fundamentales de Programación Orientada a Objetos, incluyendo Herencia y excepciones

Objetivo de la actividad:

El alumno desarrollará un programa en el cual se implementa los conceptos:

- **Creación de clases.**
- **Implementación de Clases usando objetos.**
- **Uso de atributos de un objeto y una clase.**
- **Uso de métodos de un objeto y una clase.**
- **Uso e implementación del concepto de encapsulación.**
- **Correcto manejo de sobrecarga.**
- **Uso implementación de ciclos.**
- **Uso de excepciones.**
- **Correcta definición de herencia.**

Requerimientos para la actividad:

Uso Java jdk a través de un IDE, preferentemente Netbeans. ☕

Instrucciones para el alumno:

Creación de una aplicación de gestión de biblioteca con menú de opciones, ciclos y manejo de excepciones

Descripción: Desarrollar una aplicación en Java que permita gestionar una biblioteca de libros. Deben aplicar los conceptos de creación de clases, instanciación de objetos, sobrecarga de constructores y métodos, encapsulación, herencia, ciclos y manejo de excepciones en el diseño y la implementación de la aplicación.

Instrucciones:

- **Creación de clases y Herencia:**
    - **Crear una clase base llamada "Libro" que contenga los atributos comunes a todos los libros, como título, autor y número de páginas. (X)**
    - **Crear clases derivadas para representar diferentes tipos de libros, como "LibroFiccion" y "LibroNoFiccion". Cada clase debe heredar de la clase base "Libro". (X)**
    - **Crear colecciones (Arreglos) por cada tipo de clases, iniciales y heredadas) (X)**

- **Sobrecarga de constructores y métodos:**
    - **En la clase base "Libró", implementar constructores sobrecargados que permitan la creación de objetos con diferentes atributos (por ejemplo, título y autor, título, autor y número de páginas, etc.). (X)**
    - **Sobrecargar métodos en las clases derivadas para realizar acciones específicas de cada tipo de libro, cómo calcular el precio de alquiler. (X)**
- **Encapsulación:**
    - **Utilizar modificadores de acceso (public, private, protected) para encapsular apropiadamente los atributos y métodos en las clases.**
    - **Proporcionar métodos públicos para acceder y modificar los atributos de manera controlada.**
- **Herencia y SobreEscritura de Métodos:**
    - **Crear un método en la clase base "Libro" llamado "mostrar Información" que muestre información general sobre el libro, como título, autor y número de páginas.**
    - **Sobrescribir este método en las clases derivadas para mostrar información específica de cada tipo de libro.**
- **Interfaz de usuario con menú de opciones y ciclos:**
    - **Crear un menú de opciones que permita al usuario realizar las siguientes acciones de forma repetida hasta que decida salir: (X)**
    - **Agregar un libro a la biblioteca. (X)**
    - **Calcular el precio de alquiler de un libro. (X)**
    - **Mostrar información de un libro en particular. (X)**
    - **Salir de la aplicación. (X)**

- **Implementación de las opciones del menú y manejo de excepciones:**
    - **Implementar las funciones correspondientes a cada opción del menú. (X)**
    - **Utilizar ciclos para que el menú se repita hasta que el usuario decida salir. (X)**
    - **Implementar el manejo de excepciones para controlar situaciones inesperadas, como entrada de datos incorrectos.**
- **Pruebas*:**
- **Realizar pruebas que contengan los siguientes casos:**
    - **Se ejecutan las cuatro opciones principales del menú, al menos para un caso en específico.**
    - **Forzar las excepciones implementadas.**
    - **Funcionamiento del ciclo, utilizando todas las opciones del menú y probando diferentes escenarios que puedan generar excepciones.**

***Estas pruebas deberán mostrarse en el reporte a entregar.**

Conceptos a calificar:

- **Implementación de los conceptos definidos al principio de la actividad.**
- **Funcionalidad.**
- **Ejecución de los casos de pruebas.**

🍪🍪 ***Brownie points por correcta definición de los casos de prueba***🍪🍪😃😃

1. **Reflexión: Discutir y reflexionar con diversos ejemplos de uso de objetos y posibles implementaciones usando herencia y poliformismo.**

Entregables:

- **Código Fuente, archivos con extensión .java (Recuerda, cualquier otro tipo de archivo no es permitido, i.e. .class, .zip, .jar, etc)**
- **Compilación sin errores.**
- **Cumplimiento de Especificaciones.**
- **Reporte.**

- **¡¡¡¡¡ Reflexión individual por integrante del equipo. !!!!!**

Importante: Solo se revisa la última entrega

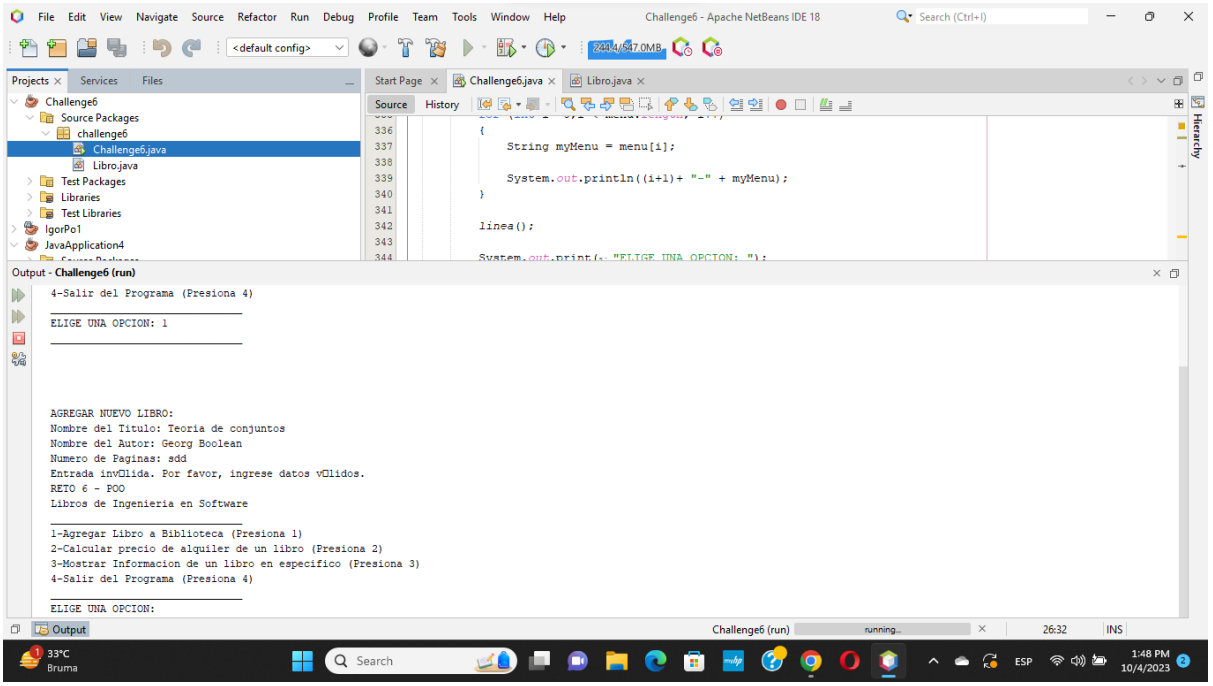
*** Sin reporte y/o código fuente, ningún porcentaje es considerado ⇒ Calificación NE**

Restricción: Equipos de no más de tres personas, no menos de dos. 😀

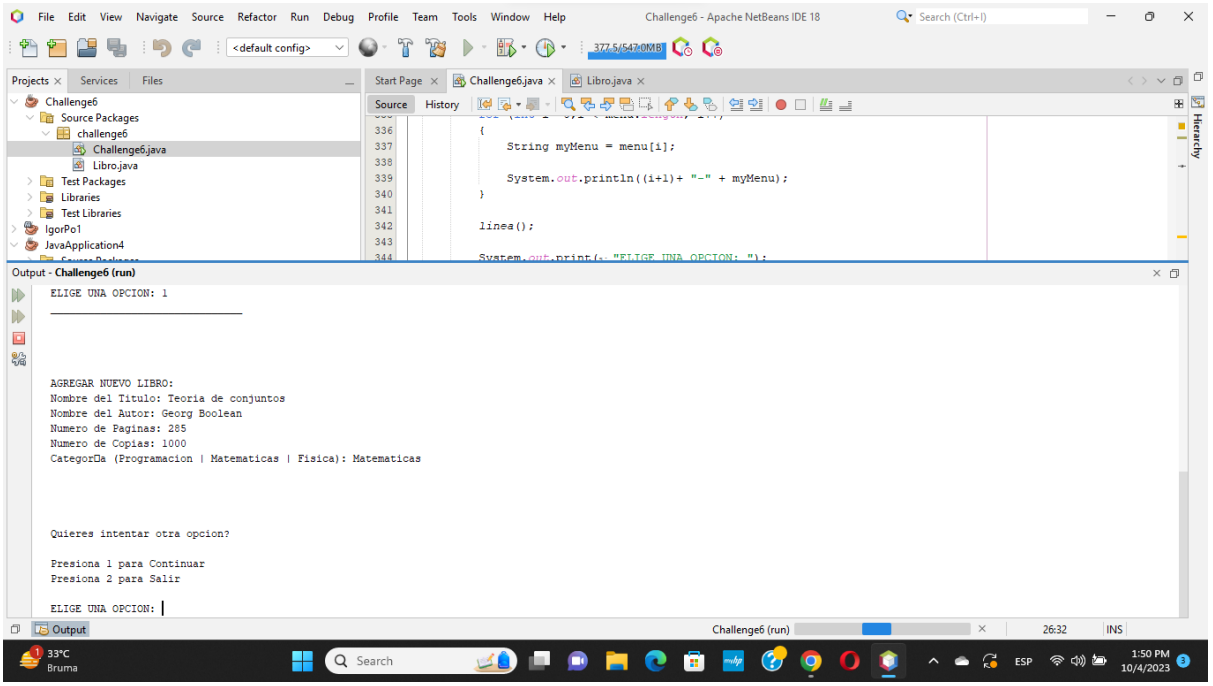
Reporte:

Hay dos classes en este programa la primera siendo Challenge6.java y la segunda siendo Libro.Java, cada una en un archivo distinto En ambas clases hay las mismas 3 sub classes que se extienden de la classe Libro siendo estas Programacion, Matematicas y Fisica, estas clases son las categorias donde se van a guardar los objetos en otras palabras cada libro, cada una de las clases tiene una función arraylist para que se guarden los arreglos dentro de cada una, la cual es recopilada por medio de un switch que usa un ciclo while con ifs anidados para mostrar un menú y recopilar los datos, los datos recopilados se guardan en los objetos de la clase libro que fueron hechos dentro de esa misma clase y se guardan en los objetos a través de getters y setters, los cuales pueden ser utilizados dentro de la clase Challenge 6 gracias a un extend. Dentro de los switch, hay opciones de if anidados que se utilizan para conseguir los datos estos switch están dentro de l¿unas funciones try para que todo dato que no se coloque dentro del programa correctamente sea mandado directamente a la función catch que va a ejecutar una función de InputMismatchException en caso de que coloque un dato que no sea del tipo correcto por ejemplo colocar algún dato de tipo string dentro cuando se estén ejecutando las funciones de get de datos int. Este mismo switch es él que muestra él menú de opciones para cuando vas a realizar cualquiera de las opciones ya que aquí se usaron ifs y switchs anidados en una jerarquía de niveles siendo él primer nivel para él menú de operaciones y él segundo para él ingreso de datos.

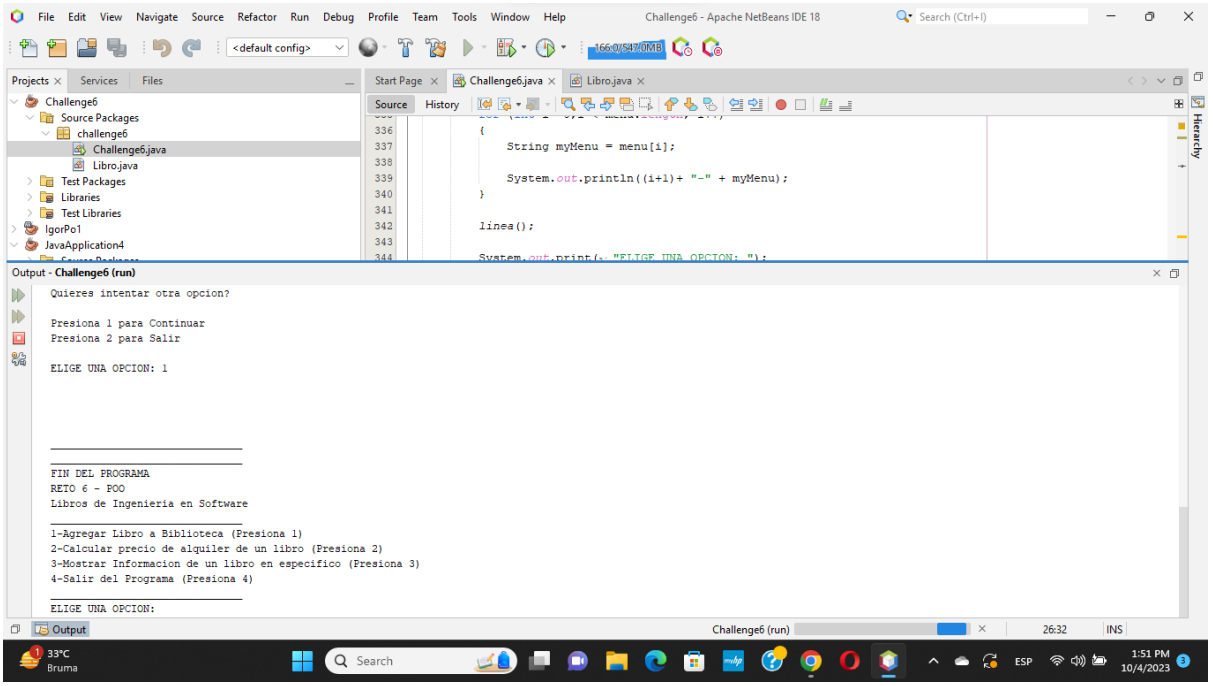
PRUEBAS:
Prueba 1 Excepción(InputMismatchException)



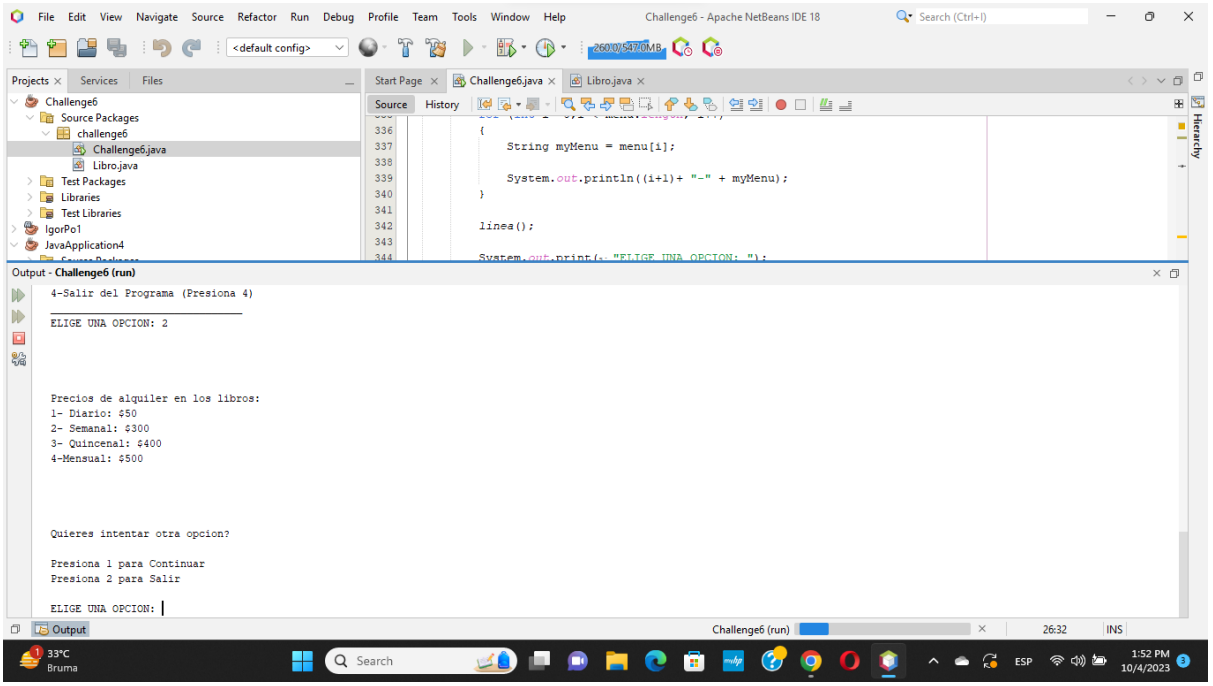
Prueba 2 Agregar un libro



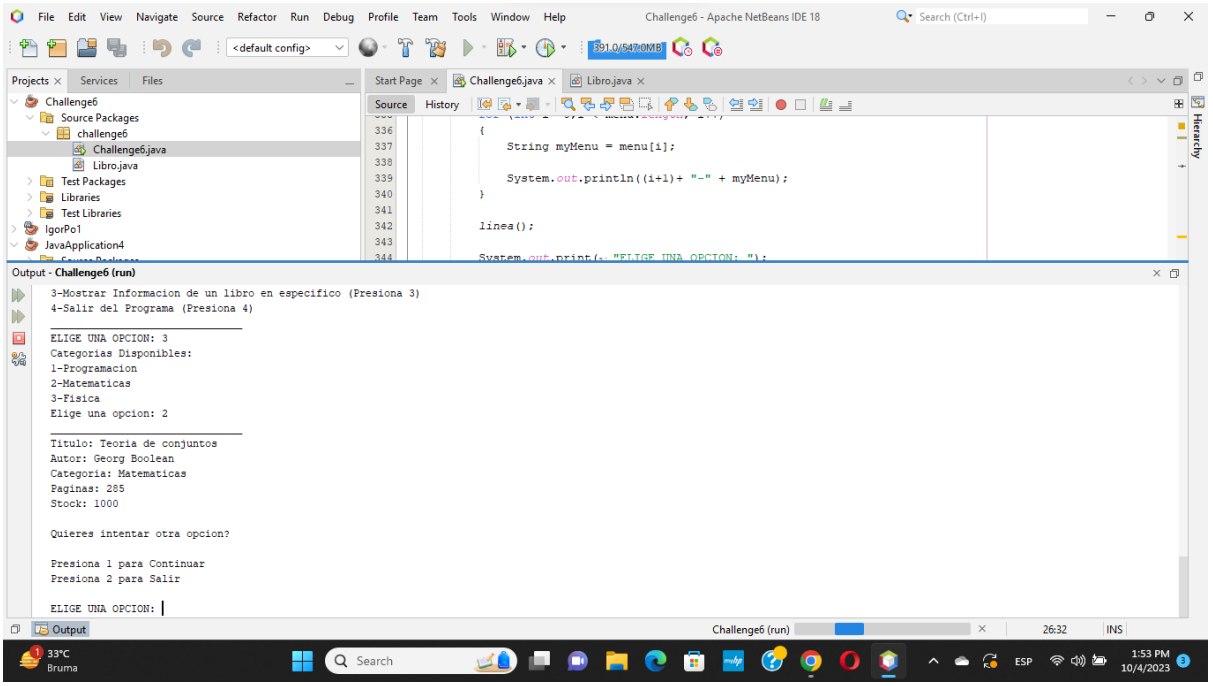
Prueba 3 Continuar



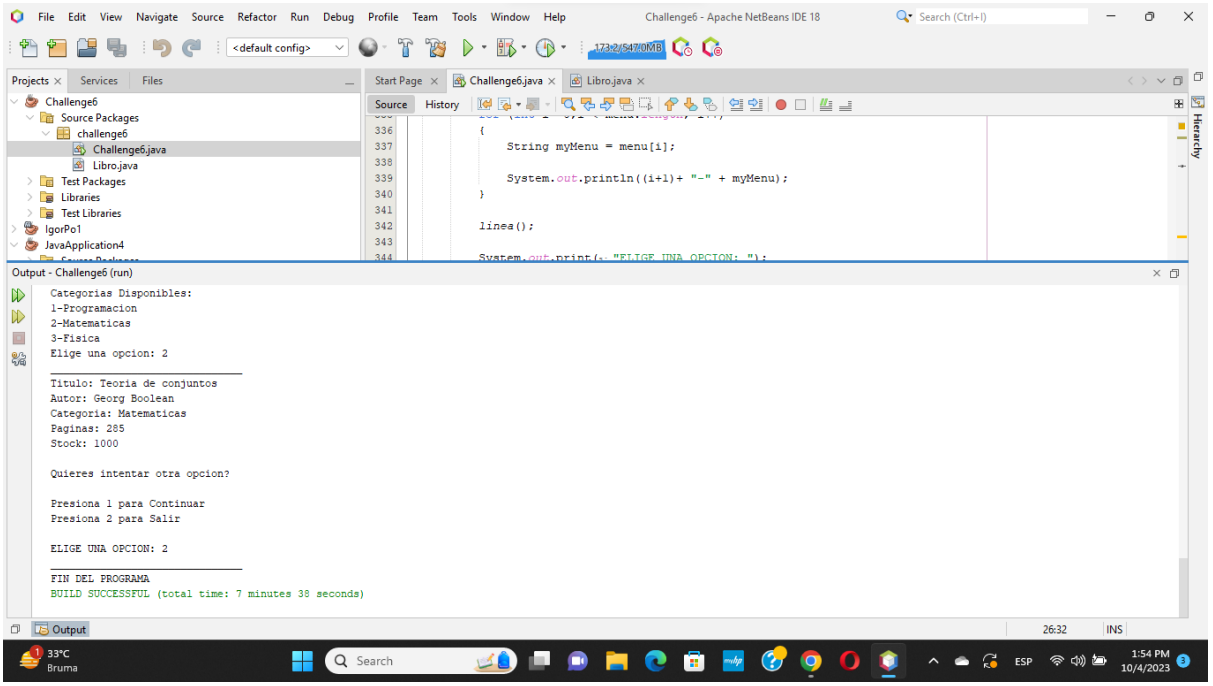
Prueba 4 Calcular Alquiler



Prueba 5 Mostrar Información de un libro específico



Prueba Final Salir del programa



CONCLUSIÓN:

Igor:

Fue bastante interesante trabajar con la herencia de clases(extend) de esta manera ya que te deja ver más posibilidades a la hora de programar, me di cuenta en este ejercicio que para lo que más sirve esta funcion es para él manejo de directorios y

hace que sea muy versatil el acomodo de los objetos ya que te permite guardar más clases y utilizarlas en conjunto en él mismo programa, lo que ayuda mucho a la hora de hacer él codigo más corto y tambien da mucha más versatilidad en él manejo de datos, ya que ahora puedes usar constructores para objetos dentro de clases distintas en conjunto sin que te suceda ningun error por él hecho de que no sean de la misma clase, si lo fueramos a decir en un termino más simple, es como cuando mandas a llamar una funcion para poder usarla de nuevo pero en niveles superiores y esta vez en vez de usar solo unas cuantas lineas de codigo dentro del mismo archivo, te permite manejar él directorio de tu computadora, obviamente siempre y cuando las clases esten dentro del mismo paquete, pero él poder ejecutar de manera simultanea un archivo distinto dentro del mismo programa facilita mucho lo que es la accesibilidad del programa haciendo que sea más practico a la hora de uso, y no solo es más practico, tambien queda mucho más rapido y comprimido, porque en vez de correr todas las lineas de codigo del archivo una por una, solo llama lo que necesitas del archivo aparte cuando se ejecute en la maquina, al mismo tiempo que evita él uso excesivo de codigo, haciendo que tengas menos lineas que ejecutar.

Mau:

Para hacer el código, me base un poco en el Reto 5 que era sobre agregar productos en un inventario virtual, básicamente el programa del Reto 6 es prácticamente lo mismo, únicamente la diferencia es que ahora se trata de guardar libros en diferentes categorías, y cada categoría va en un arreglo único. Pudimos hacer miles de arreglos por cada categoría, pero para ahorrarnos más tiempo, decidimos mejor hacer categorías de algo que sea relevante en nuestras vidas como la carrera que estudio, únicamente hicimos la categoria de Programación, Matemáticas y Física para hacer mas simple el trabajo. Ahorita el chiste es ir

practicando y guardar este trabajo para en el futuro que trabaje en proyectos pueda consultar mis tareas de la carrera y ver como puedo programar algún algoritmo en especial. Al principio fue algo complicado para mi programar este codigo, luego lo complete pero tuve varios bugs y al fin investigué en internet como puedo mejorar mi codigo y por fin pude hacer el reto correctamente.