Laboratorio 4 - Programación Orientada a Objetos

Principios y Prácticas de Desarrollo de Software Orientado a Objetos
grupo A1

Estudiante:
Daniel Eduardo Cobos Ayala - 2191954

Presentado a:
Darío Alejandro Riaño Velandia

Bucaramanga
2023

Contenido

Introducción

En el desarrollo de este laboratorio se llevaron a cabo una serie de ejercicios destinados a aplicar conceptos teóricos y desarrollar habilidades prácticas. Estos ejercicios abarcaron tres áreas fundamentales de la programación en Java y se centraron en la implementación de soluciones prácticas.

El primer ejercicio se centrará en la creación de una máquina de café simulada en Java. Esta máquina tendrá la capacidad de aceptar monedas como pago y dispensar café. Lo destacado de este ejercicio será la implementación del manejo de excepciones. Se introducirá la excepción estándar `InputMismatchException` para controlar errores cuando los usuarios ingresen datos no válidos, y se diseñará una excepción personalizada llamada `InsufficientFundsException` para abordar situaciones en las que los usuarios no proporcionen suficiente dinero para comprar el café.

El segundo ejercicio se enfocará en la creación de un sistema para calcular el salario neto de los empleados de una empresa ficticia llamada "Soluciones Clic Derecho". Para lograrlo, se emplearán clases abstractas, específicamente la clase "Empleado" que servirá como plantilla base. Las clases "EmpleadoPlanta" y "EmpleadoTemporal" heredarán de esta clase abstracta y proporcionarán implementaciones específicas para calcular el salario neto. Además, se utilizará una interfaz llamada "AfiliaciónSindical" para definir un contrato que las clases de empleados deberán implementar, lo que permitirá gestionar los detalles relacionados con la afiliación sindical. La nómina de la empresa se manejará utilizando una estructura de datos adecuada.

El tercer ejercicio se centrará en la implementación de un sistema de registro de cuotas de manejo para diferentes tipos de tarjetas en el banco "Ahorra cuando puedes". Este ejercicio involucrará conceptos avanzados de programación orientada a objetos, como herencia, clases abstractas, interfaces, enumeradores y patrones contenedores. Se crearán clases derivadas, como "Joven", "Nomina" y "Visa", que heredarán de la clase abstracta "Tarjeta". Se utilizará una interfaz llamada "PatronCartera" para garantizar la implementación de métodos específicos, y se emplearán enumeradores para gestionar los tipos de descuento. La estructura de datos adecuada se implementará en la clase "Cartera" para administrar múltiples objetos de tarjeta.

Estos ejercicios proporcionarán una oportunidad para poner en práctica los conceptos aprendidos en el curso y fortalecer las habilidades de programación orientada a objetos, el manejo de excepciones y la implementación de soluciones de software efectivas.

Ejercicios

- **Máquina de café con excepciones.**

  En el código relacionado a este ejercicio, se implementaron dos tipos de excepciones: InputMismatchException y una excepción personalizada llamada InsufficientFundsException.

  InputMismatchException: Esta es una excepción estándar en Java que ocurre cuando se produce un error al intentar leer un valor de un Scanner que no coincide con el tipo de datos esperado. En el código, se utiliza para capturar errores cuando el usuario ingresa datos que no son números enteros válidos. Por ejemplo, si el usuario ingresa letras en lugar de un número al ingresar el dinero, se lanza esta excepción.

  InsufficientFundsException (excepción personalizada): Esta es una excepción personalizada que se ha creado específicamente para manejar la situación en la que un usuario no proporciona suficiente dinero para comprar café. En lugar de utilizar una excepción estándar, se ha definido una clase personalizada para este propósito.

- **Ejercicio clases abstractas.**

  En el código relacionado a este ejercicio, se realiza un software para calcular el salario neto a pagar a los empleados de la empresa "Soluciones Clic Derecho". Para esto, se utilizan los siguientes conceptos y componentes de Java: Clases abstractas: Se utilizan clases abstractas en la forma de la clase "Empleado". Las clases "EmpleadoPlanta" y "EmpleadoTemporal" heredarán de esta clase abstracta y proporcionarán implementaciones específicas de métodos relacionados con el cálculo del salario neto. Clases tipo interface: Se utiliza una interfaz llamada "AfiliacionSindical" para definir un contrato que las clases de empleados deben implementar. La interfaz especifica dos métodos: "getBonoAntiguedad" y "getAporteASindicato". La clase "EmpleadoPlanta" deberá implementar estos métodos para proporcionar valores específicos relacionados con la afiliación sindical. Uso de colecciones en la aplicación de patrón contenedor: La clase "Nomina" se utiliza como un patrón contenedor para mantener una lista de objetos de tipo "Empleado". Esto permite organizar y gestionar de manera eficiente los detalles de los empleados en la nómina de la empresa "Soluciones Clic Derecho". La lista se inicializa en el constructor, se pueden agregar empleados con el método "addEmpleado" y se puede imprimir la nómina con "imprimirNomina".

- **Ejercicio tarjetas de crédito.**

  El ejercicio se centra en la implementación de un sistema de registro de cuotas de manejo para diferentes tipos de tarjetas en el banco "Ahorra cuando puedes" en Java. Se deben utilizar varios conceptos y características de programación orientada a objetos, se mencionan a continuación:

Herencia: Debe comprender cómo funcionan las clases base y derivadas, ya que se utilizan para crear las clases Joven, Nomina y Visa, que heredan de la clase Tarjeta.

Clases abstractas: Debe entender cómo definir y utilizar clases abstractas, ya que la clase Tarjeta es una clase abstracta que sirve como plantilla para las clases derivadas.

Interfaces: Se usa la interfaz *PatronCartera* para garantizar que las clases cumplan con un conjunto específico de métodos. Debe saber cómo implementar interfaces en las clases.

Enumeradores: Debe estar familiarizado con los enumeradores en Java para crear un enumerador llamado Descuento y usarlo para gestionar los tipos de descuento.

Patrón contenedor: Debe comprender cómo implementar un patrón contenedor, que implica almacenar y gestionar múltiples objetos (en este caso, las tarjetas) en una estructura de datos adecuada dentro de la clase Cartera.

Evidencia

- **Máquina de café con excepciones.**
  https://github.com/danny2768/PrincipiosPracticasPOO/tree/master/Lab4/CoffeeMachineWithExceptions
- **Ejercicio clases abstractas.**
  https://github.com/danny2768/PrincipiosPracticasPOO/tree/master/Lab4/AbstractClasses
- **Ejercicio tarjetas de crédito.**
  https://github.com/danny2768/PrincipiosPracticasPOO/tree/master/Lab4/BancoAhorra

Lecciones aprendidas

En el desarrollo de este laboratorio, se han obtenido valiosas lecciones y conocimientos en relación a la programación en Java. Dos aspectos destacados que se aprendieron son:

Enums en Java: Durante el ejercicio de tarjetas de crédito, se utilizó un enumerador llamado *"EDescuento"* para gestionar diferentes tipos de descuentos. Los enumeradores en Java son una forma eficaz de definir un conjunto fijo de constantes relacionadas. Se aprendió cómo crear, utilizar y aprovechar al máximo los enumeradores para mantener un código más limpio y legible.

Manejo de Excepciones en Java: En el ejercicio de la máquina de café, se implementaron excepciones, incluyendo la excepción personalizada *"InsufficientFundsException"*. Además, se utilizó la excepción estándar *"InputMismatchException"* para gestionar errores de entrada del usuario. A lo largo de este ejercicio, se adquirió una comprensión sólida sobre cómo manejar excepciones de manera efectiva en Java, lo que es esencial para desarrollar aplicaciones robustas y resistentes a fallos.

Estas lecciones aprendidas no solo son aplicables a los ejercicios mencionados, sino que también son fundamentales en la programación en Java en general. La comprensión de los enumeradores y el manejo de excepciones son habilidades esenciales que los desarrolladores pueden aplicar en una amplia variedad de proyectos, lo que demuestra la importancia de la práctica y la experimentación en el aprendizaje de la programación.

Referencias

[1] Baeldung, & Baeldung. (2023). A guide to Java Enums | Baeldung. Baeldung. https://www.baeldung.com/a-guide-to-java-enums

[2] Enum Types (The JavaTM Tutorials > Learning the Java Language > Classes and Objects). (n.d.). https://docs.oracle.com/javase/tutorial/java/javaOO/enum.html

[3] Abstract Methods and Classes (The JavaTM Tutorials >Learning the Java Language > Interfaces and Inheritance). (n.d.). https://docs.oracle.com/javase/tutorial/java/IandI/abstract.html