



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Ingeniería en computación

Laboratorios de computación salas A y B

Profesor: _____ Ing. Rene Adrian Davila Perez

Asignatura: _____ Programación Orientada a Objetos

Grupo: _____ 1

No. de práctica: _____ 5 y 6

Integrante: _____ 424121462

Semestre: _____ 2025-1

Fecha de entrega: _____ 27 de septiembre de 2024

Observaciones: _____

CALIFICACIÓN: _____

Índice

1. Introducción (2)
2. Marco Teórico (3)
3. Desarrollo (3-5)
4. Resultados (5-8)
5. Conclusiones (9)
6. Referencias (9)

Introducción:

La práctica se dividirá en varios ejercicios, cuyos planteamientos de problemas se presentan en el siguiente orden ascendente:

Ejercicio 1:

- **Paso 1.** Ajustar el programa del Proyecto 1 agregando capacidades de encapsulamiento.
- **Paso 2.** Definir un paquete con base en las convenciones vistas en la parte teórica vista en la primera hora.
- **Paso 3.** Crear el paquete al momento de compilar el ejercicio.
- **Paso 4.** Con el paquete creado, comprimirlo en archivos jar (Java Archive).
- **Paso 5.** Crear Javadoc de las clases creadas.

Ejercicios de tarea:

- Rehacer las prácticas 1 a 4 con los conceptos de POO hasta encapsulamiento y paquetes.
- Definir una convención de paquete similar a lo visto en la sesión práctica.
- Cada práctica ajustada va en su propio paquete.
- Se debe crear un jar por cada práctica.
- Se debe crear Javadoc de cada clase.
- En el reporte se debe definir el diagrama de cada clase creada.

Estos problemas se resolverán utilizando los métodos vistos en clase y/o basándose en lo aprendido hasta la fecha. Los ejercicios propuestos tienen como finalidad:

Ejercicio 1: El ejercicio realizado tendrá como finalidad el elaborar y ejecutar de manera correcta los pasos para realizar de manera correcta el empaquetado y encapsulamiento de clases, a su vez, se tiene como objetivo comprender la manera de acceso de estas clases en distintas ubicaciones de la aplicación.

Ejercicios de tarea: Los ejercicios de tarea tienen como finalidad el lograr colocar de mejor manera las bases de esta nueva herramienta, la cual se utiliza en diversas áreas de la programación, esta herramienta es muy común en proyectos de escalas más grandes debido a la utilidad de los paquetes, pudiendo tener mayor control y estabilidad a futuro, lo cual da escalabilidad a los proyectos. Por otra parte, el realizar documentación de los códigos proporciona una vista más profesional y de identidad a los proyectos realizados.

Además, se busca realizar esta práctica con las mejores herramientas y convenciones posibles, lo cual se ve altamente beneficiado en el ámbito laboral, ya que el realizar comentarios y/o tener una buena documentación facilita el trabajo en equipo, lo cual permite la resolución del proyecto de manera más eficiente

Marco Teórico:

El encapsulamiento en Java consiste en ocultar los detalles internos de cómo funciona una clase y exponer sólo los métodos que permiten que interactúe dicha clase. Esto se logra definiendo las variables de la clase como privadas y proporcionando métodos públicos para establecer setters y obtener getters. El encapsulamiento en Java en otras palabras ayuda a mantener la integridad de los datos y a controlar cómo se accede y se modifica la información dentro de los objetos de una clase. [1]

Un archivo JAR es un formato que combina múltiples archivos en uno solo. El compilador de Java convierte el código fuente en instrucciones para la máquina virtual, que se almacenan en archivos de clase. Los archivos JAR pueden contener estos archivos de clase y se usan tanto para archivar como para distribuir programas Java. [2]

El comando **javadoc** analiza archivos fuente de Java para generar documentación en HTML de elementos públicos y protegidos como clases, métodos, interfaces y campos. Se puede utilizar para crear documentación de la API o de la implementación. Se puede aplicar a paquetes completos o a archivos específicos.[3]

Desarrollo:

Para la clase **Punto** se tiene:

- latitud: double
- longitud: double
- Métodos set y get de cada una de las variables
- Método constructor

Para realizar las pruebas de la Práctica 1, se utilizaron los valores de latitud y longitud del punto 1 como 80, para el punto dos se usaron de 90; se espera como respuesta la correcta ejecución del programa, mostrando la distancia aproximada de 1111 [km].

Para la clase **TrianguloPascal** se tiene:

- filas: int
- Método constructor
- void mostrarTriangulo(): muestra en pantalla el triángulo de Pascal
- private int factorial(@x): retorna el factorial de un número entero dado

Para realizar las pruebas de Práctica 2 se realizó con n = 5, para así no tomar valores tan grandes y/o alejados, obteniendo como resultado el triángulo de Pascal de solo 5 filas.

Para la clase **Caracter** se tiene:

- cadena: String
- Métodos set y get la variable
- Método constructor

- String reemplazarCaracter(@c1, @c2): retorna una cadena actualizada dado una cadena original (c1) y el carácter que se desea reemplazar (c2) en la cadena original

Para realizar las pruebas de Práctica 31 se utilizará la cadena: “HOLA PROFE”, además se querrá reemplazar el carácter “O” con el carácter “o”, esperando como resultado la nueva cadena: “HoLA PRoFE”.

Para la clase **ArrayColores** se tiene:

- colores: ArrayList<String>
- Método constructor
- cambiarColor(@indice, @nuevoColor): cambia un color en el ArrayList dados un índice y un color
- mostrarLista(): muestra en pantalla la lista
- String obtenerColor(@indice): retorna la cadena que le pertenece a un índice dado

Para realizar las pruebas de Práctica 32 se utilizó la lista integrada en el constructor de la clase correspondiente, a su vez se le ingresó el color rosa como el reemplazo al color verde, este perteneciente al método.

Para la clase **MapaColores** se tiene:

- mapa: HashMap<Integer, String>
- Método constructor
- String obtenerContenido(@llave): retorna un contenido dada una llave
- mostrarMapa(): muestra en pantalla el mapa

Para realizar las pruebas de Práctica 33 se utilizó el mapa integrado en el constructor de la clase correspondiente, a su vez se le ingresó la llave 5 para así poder acceder a su contenido.

Para la clase **CuentaBanco** se tiene:

- nombre: String
- cuenta: String
- saldo: double
- Métodos set y get de cada una de las variables
- Método constructor
- depositar(@monto): incrementa el saldo dado un monto
- retirar(@monto): decrementa el saldo dado un monto
- mostrarInfoCuenta(): muestra en pantalla la información de la cuenta

Para realizar las pruebas de Práctica 4 se utilizó se utilizarán los valores de “Axel Cote”, “424121462” y 100 como los valores de nombre, cuenta y saldo respectivamente, además se utilizarán los valores de 55 y 100 como los valores de monto a depositar y monto retirar respectivamente. Se espera como resultado el funcionamiento correcto del programa y que se muestre en pantalla de manera correcta.

Para la clase **Libro** se tiene:

- título: String
- autor: String
- Métodos set y get de cada una de las variables
- Método constructor

Para la clase **Libreria** se tiene:

- libros: ArrayList<Libro>
- Método constructor
- agregarLibro(@libro): agrega un libro dado
- quitarLibro(@libro): quita un libro dado
- ArrayList<Libro> getLibros(): retorna libros

Para realizar las pruebas del Proyecto 1, se ejecutó el programa sin ingresar o modificar valores debido a que el programa así fue diseñado, por lo que se esperaba que se ejecutaría de manera cotidiana.

Resultados:

```
File Edit Selection View Go Run ... ← → 🔍 Práctica56
PROBLEMS 12 OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + - 🌐 ...
PS C:\Users\axeld\Desktop\eda2\Práctica56> cd práctica1
PS C:\Users\axeld\Desktop\eda2\Práctica56\práctica1> javac -d . Punto.java
PS C:\Users\axeld\Desktop\eda2\Práctica56\práctica1> javac -d . Practical.java
PS C:\Users\axeld\Desktop\eda2\Práctica56\práctica1> java mx.unam.fi.poo.g1.p1.Practical1

Ingresa las coordenadas decimales de la latitud (x1) del punto 1: 80
Ingresa las coordenadas decimales de la longitud (y1) del punto 1: 80

Ingresa las coordenadas decimales de la latitud (x2) del punto 2: 90
Ingresa las coordenadas decimales de la longitud (y2) del punto 2: 90

La distancia entre los dos puntos es: 1111.9510117748407 [km]
PS C:\Users\axeld\Desktop\eda2\Práctica56\práctica1> jar -cvfe Practical1.jar mx.unam.fi.poo.g1.p1.Practical1 mx/unam/fi/poo/g1/p1/
added manifest
adding: mx/unam/fi/poo/g1/p1/(in = 0) (out= 0)(stored 0%)
adding: mx/unam/fi/poo/g1/p1/Practical1.class(in = 1922) (out= 1018)(deflated 47%)
adding: mx/unam/fi/poo/g1/p1/Punto.class(in = 565) (out= 349)(deflated 38%)
PS C:\Users\axeld\Desktop\eda2\Práctica56\práctica1> jar tf Practical1.jar
META-INF/
META-INF/MANIFEST.MF
mx/unam/fi/poo/g1/p1/
mx/unam/fi/poo/g1/p1/Practical1.class
mx/unam/fi/poo/g1/p1/Punto.class
PS C:\Users\axeld\Desktop\eda2\Práctica56\práctica1> java -jar Practical1.jar

Ingresa las coordenadas decimales de la latitud (x1) del punto 1: 80
Ingresa las coordenadas decimales de la longitud (y1) del punto 1: 80

Ingresa las coordenadas decimales de la latitud (x2) del punto 2: 90
Ingresa las coordenadas decimales de la longitud (y2) del punto 2: 90

La distancia entre los dos puntos es: 1111.9510117748407 [km]
PS C:\Users\axeld\Desktop\eda2\Práctica56\práctica1> javadoc -d punto Punto.java
Loading source file Punto.java...
Constructing Javadoc information...
Creating destination directory: "punto\"
```

Se muestra la ejecución de la Práctica 1 en la cual se muestran los resultados esperados descritos en el desarrollo junto con la creación del archivo.jar y el javadoc de la clase.

```
PS C:\Users\axeld\Desktop\eda2\Practica56> cd practica2
PS C:\Users\axeld\Desktop\eda2\Practica56\practica2> javac -d . TrianguloPascal.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica2> javac -d . Practica2.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica2> java mx.unam.fi.poo.g1.p2.Practica2
Escribe el numero de filas: 5
    1
    1 1
    1 2 1
    1 3 3 1
    1 4 6 4 1
PS C:\Users\axeld\Desktop\eda2\Practica56\practica2> jar -cvfe Practica2.jar mx.unam.fi.poo.g1.p2.Practica2 mx/unam/fi/poo/g1/p2/
added manifest
adding: mx/unam/fi/poo/g1/p2/(in = 0) (out= 0)(stored 0%)
adding: mx/unam/fi/poo/g1/p2/Practica2.class(in = 743) (out= 489)(deflated 34%)
adding: mx/unam/fi/poo/g1/p2/TrianguloPascal.class(in = 961) (out= 684)(deflated 28%)
PS C:\Users\axeld\Desktop\eda2\Practica56\practica2> jar tf Practica2.jar
META-INF/
META-INF/MANIFEST.MF
mx/unam/fi/poo/g1/p2/
mx/unam/fi/poo/g1/p2/Practica2.class
mx/unam/fi/poo/g1/p2/TrianguloPascal.class
PS C:\Users\axeld\Desktop\eda2\Practica56\practica2> java -jar Practica2.jar
Escribe el numero de filas: 5
    1
    1 1
    1 2 1
    1 3 3 1
    1 4 6 4 1
PS C:\Users\axeld\Desktop\eda2\Practica56\practica2> javadoc -d triangulo TrianguloPascal.java
Loading source file TrianguloPascal.java...
Constructing Javadoc information...
Creating destination directory: "triangulo\"...
Building index for all the packages and classes...
Standard Doclet version 19.0.1+10-21
Building tree for all the packages and classes...
Generating triangulo\mx\unam\fi\poo\g1\p2\TrianguloPascal.html...
```

Se muestra la ejecución de la Práctica 2 en la cual se muestra el triángulo de Pascal de 5 filas junto con la creación del archivo.jar y el javadoc de la clase.

```
PS C:\Users\axeld\Desktop\eda2\Practica56> cd practica31
PS C:\Users\axeld\Desktop\eda2\Practica56\practica31> javac -d . Caracter.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica31> javac -d . Practica31.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica31> java mx.unam.fi.poo.g1.p3.Practica31
Escribe una frase: HOLA PROFE
Escribe el carácter que quieras reemplazar: O
Escribe el carácter por el que lo que quieras reemplazar: o
Frase modificada: Hola pRoFE
PS C:\Users\axeld\Desktop\eda2\Practica56\practica31> jar -cvfe Practica31.jar mx.unam.fi.poo.g1.p3.Practica31 mx/unam/fi/poo/g1/p3/
added manifest
adding: mx/unam/fi/poo/g1/p3/(in = 0) (out= 0)(stored 0%)
adding: mx/unam/fi/poo/g1/p3/Caracter.class(in = 695) (out= 383)(deflated 44%)
adding: mx/unam/fi/poo/g1/p3/Practica31.class(in = 1467) (out= 779)(deflated 46%)
PS C:\Users\axeld\Desktop\eda2\Practica56\practica31> jar tf Practica31.jar
META-INF/
META-INF/MANIFEST.MF
mx/unam/fi/poo/g1/p3/
mx/unam/fi/poo/g1/p3/Caracter.class
mx/unam/fi/poo/g1/p3/Practica31.class
PS C:\Users\axeld\Desktop\eda2\Practica56\practica31> java -jar Practica31.jar
Escribe una frase: hola profe
Escribe el carácter que quieras reemplazar: o
Escribe el carácter por el que lo que quieras reemplazar: O
Frase modificada: Hola prOfe
PS C:\Users\axeld\Desktop\eda2\Practica56\practica31> javadoc -d caracter Caracter.java
Loading source file Caracter.java...
Constructing Javadoc information...
Creating destination directory: "caracter\"...
Building index for all the packages and classes...
Standard Doclet version 19.0.1+10-21
Building tree for all the packages and classes...
Generating caracter\mx\unam\fi\poo\g1\p3\Caracter.html...
Generating caracter\mx\unam\fi\poo\g1\p3\package-summary.html...
Generating caracter\mx\unam\fi\poo\g1\p3\package-tree.html...
Generating caracter\overview-tree.html...
Building index for all classes...
```

Se muestra la ejecución de la Práctica 31 en la cual se muestra la cadena original y actualizada en base a lo descrito en el desarrollo junto con la creación del archivo.jar y el javadoc de la clase.

```

File Edit Selection View Go Run ... ← → 🔍 Practica56
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + ⌂ ... ×
PS C:\Users\axeld\Desktop\eda2\Practica56> cd practica32
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> javac -d . ArrayColores.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> javac -d . Practica32.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> java mx.unam.fi.poo.g1.p3.Practica32
La lista es: [rojo, verde, azul, blanco, negro, morado]
Escriba un nuevo color a cambiar por "verde": rosa
La lista es: [rojo, rosa, azul, blanco, negro, morado]
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> jar -cvfe Practica32.jar mx.unam.fi.poo.g1.p3.Practica32 mx/unam/fi/poo/g1/p3/
added manifest
adding: mx/unam/fi/poo/g1/p3/(in = 0) (out= 0)(stored 0%)
adding: mx/unam/fi/poo/g1/p3/ArrayColores.class(in = 1722) (out= 921)(deflated 46%)
adding: mx/unam/fi/poo/g1/p3/Practica32.class(in = 1322) (out= 728)(deflated 44%)
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> jar tf Practica32.jar
META-INF/
META-INF/MANIFEST.MF
mx/unam/fi/poo/g1/p3/
mx/unam/fi/poo/g1/p3/ArrayColores.class
mx/unam/fi/poo/g1/p3/Practica32.class
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> jar tf Practica32.jar
META-INF/
META-INF/MANIFEST.MF
mx/unam/fi/poo/g1/p3/
mx/unam/fi/poo/g1/p3/ArrayColores.class
mx/unam/fi/poo/g1/p3/Practica32.class
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> java -jar Practica32.jar
La lista es: [rojo, verde, azul, blanco, negro, morado]
Escriba un nuevo color a cambiar por "verde": rosa
La lista es: [rojo, rosa, azul, blanco, negro, morado]
PS C:\Users\axeld\Desktop\eda2\Practica56\practica32> javadoc -d array ArrayColores.java
Loading source file ArrayColores.java...
Constructing Javadoc information...
Creating destination directory: "array\"...
Building index for all the packages and classes...
Standard Doclet version 19.0.1+10-21
Building tree for all the packages and classes...
Generating array\mx\unam\fipoo\g1\p3\ArrayColores.html...

```

Se muestra la ejecución de la Práctica 32 en la cual se muestra el ArrayList original y actualizado en base a lo descrito en el desarrollo junto con la creación del archivo.jar y el javadoc de la clase.

```

File Edit Selection View Go Run ... ← → 🔍 Practica56
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS powershell + ⌂ ... ×
PS C:\Users\axeld\Desktop\eda2\Practica56> cd practica33
PS C:\Users\axeld\Desktop\eda2\Practica56\practica33> javac -d . MapaColores.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica33> javac -d . Practica33.java
PS C:\Users\axeld\Desktop\eda2\Practica56\practica33> java mx.unam.fi.poo.g1.p3.Practica33
Mapa original: {1=rojo, 2=verde, 3=negro, 4=blanco, 5=azul}
Escriba el valor de la llave para acceder a su contenido: 5
Contenido: azul
PS C:\Users\axeld\Desktop\eda2\Practica56\practica33> jar -cvfe Practica33.jar mx.unam.fi.poo.g1.p3.Practica33 mx/unam/fi/poo/g1/p3/
added manifest
adding: mx/unam/fi/poo/g1/p3/(in = 0) (out= 0)(stored 0%)
adding: mx/unam/fi/poo/g1/p3/MapaColores.class(in = 1579) (out= 800)(deflated 49%)
adding: mx/unam/fi/poo/g1/p3/Practica33.class(in = 1312) (out= 736)(deflated 43%)
PS C:\Users\axeld\Desktop\eda2\Practica56\practica33> jar tf Practica33.jar
META-INF/
META-INF/MANIFEST.MF
mx/unam/fi/poo/g1/p3/
mx/unam/fi/poo/g1/p3/MapaColores.class
mx/unam/fi/poo/g1/p3/Practica33.class
PS C:\Users\axeld\Desktop\eda2\Practica56\practica33> java -jar Practica33.jar
Mapa original: {1=rojo, 2=verde, 3=negro, 4=blanco, 5=azul}
Escriba el valor de la llave para acceder a su contenido: 5
Contenido: azul
PS C:\Users\axeld\Desktop\eda2\Practica56\practica33> javadoc -d mapa MapaColores.java
Loading source file MapaColores.java...
Constructing Javadoc information...
Creating destination directory: "mapa\"...
Building index for all the packages and classes...
Standard Doclet version 19.0.1+10-21
Building tree for all the packages and classes...
Generating mapa\mx\unam\fipoo\g1\p3\MapaColores.html...
Generating mapa\mx\unam\fipoo\g1\p3\package-summary.html...
Generating mapa\mx\unam\fipoo\g1\p3\package-tree.html...
Generating mapa\overview-tree.html...
Building index for all classes...
Generating mapa\allclasses-index.html...
Generating mapa\allpackages-index.html...

```

Se muestra la ejecución de la Práctica 33 en la cual se muestra el mapa original y el valor del contenido obtenido en base al índice 5 junto con la creación del archivo.jar y el javadoc de la clase.

```

File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell + - x
PS C:\Users\axeld\Desktop\eda2\Practica56\practica4> java mx.unam.fi.poo.g1.p4.Practica4
Nombre del beneficiario: Axel Cote
No. de cuenta del beneficiario: 424121462
Deposito inicial: 100
Beneficiario: Axel Cote
No. de cuenta: 424121462
Saldo actual: 100.0

¿Cuanto desea depositar?: 55
Beneficiario: Axel Cote
No. de cuenta: 424121462
Saldo actual: 155.0

¿Cuanto desea retirar?: 100
Beneficiario: Axel Cote
No. de cuenta: 424121462
Saldo actual: 55.0
PS C:\Users\axeld\Desktop\eda2\Practica56\practica4> jar -cvfe Practica4.jar mx.unam.fi.poo.g1.p4.Practica4 mx/unam/fi/poo/g1/p4/
added manifest
adding: mx/unam/fi/poo/g1/p4/ (in = 0) (out= 0)(stored 0%)
adding: mx/unam/fi/poo/g1/p4/CuentaBanco.class(in = 1602) (out= 830)(deflated 48%)
adding: mx/unam/fi/poo/g1/p4/Practica4.class(in = 1129) (out= 682)(deflated 39%)
PS C:\Users\axeld\Desktop\eda2\Practica56\practica4> jar tf Practica4.jar
META-INF/
META-INF/MANIFEST.MF
mx/unam/fi/poo/g1/p4/
mx/unam/fi/poo/g1/p4/CuentaBanco.class
mx/unam/fi/poo/g1/p4/Practica4.class
PS C:\Users\axeld\Desktop\eda2\Practica56\practica4> java -jar Practica4.jar
Nombre del beneficiario: Axel Cote
No. de cuenta del beneficiario: 424121462
Deposito inicial: 100
Beneficiario: Axel Cote
No. de cuenta: 424121462
Saldo actual: 100.0

¿Cuanto desea depositar?: 55
Beneficiario: Axel Cote
No. de cuenta: 424121462
Saldo actual: 155.0

¿Cuanto desea retirar?: 100
Beneficiario: Axel Cote
No. de cuenta: 424121462
Saldo actual: 55.0

```

Se muestra la ejecución de la Práctica 4 en la cual se muestra lo descrito en el desarrollo junto con la creación del archivo.jar y el javadoc de la clase.

```

File Edit Selection View Go Run ... ← → 🔍 Practica56
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell + - x
PS C:\Users\axeld\Desktop\eda2\Practica56\proyecto1> java mx.unam.fi.poo.g1.p56.Proyecto1
Libros disponibles:
La historia interminable por Michael Ende
Crimen y Castigo por Fedor M.Dostoievski
Guerra y paz por Leon Tolstoi
Orgullo y prejuicio por Jane Austen
Libros disponibles despues de quitar Orgullo y prejuicio:
Momo por Michael Ende
La historia interminable por Michael Ende
Crimen y Castigo por Fedor M.Dostoievski
Guerra y paz por Leon Tolstoi
PS C:\Users\axeld\Desktop\eda2\Practica56\proyecto1> jar -cvfe Proyecto1.jar mx.unam.fi.poo.g1.p56.Proyecto1 mx/unam/fi/poo/g1/p56/
added manifest
adding: mx/unam/fi/poo/g1/p56/(in = 0) (out= 0)(stored 0%)
adding: mx/unam/fi/poo/g1/p56/Libreria.class(in = 760) (out= 410)(deflated 46%)
adding: mx/unam/fi/poo/g1/p56/Libro.class(in = 639) (out= 361)(deflated 43%)
adding: mx/unam/fi/poo/g1/p56/Proyecto1.class(in = 2094) (out= 1119)(deflated 46%)
PS C:\Users\axeld\Desktop\eda2\Practica56\proyecto1> jar tf Proyecto1.jar
META-INF/
mx/unam/fi/poo/g1/p56/
mx/unam/fi/poo/g1/p56/Libreria.class
mx/unam/fi/poo/g1/p56/Libro.class
mx/unam/fi/poo/g1/p56/Proyecto1.class
PS C:\Users\axeld\Desktop\eda2\Practica56\proyecto1> java -jar Proyecto1.jar
Libros disponibles:
Momo por Michael Ende
La historia interminable por Michael Ende
Crimen y Castigo por Fedor M.Dostoievski
Guerra y paz por Leon Tolstoi
Orgullo y prejuicio por Jane Austen
Libros disponibles despues de quitar Orgullo y prejuicio:
Momo por Michael Ende
La historia interminable por Michael Ende
Crimen y Castigo por Fedor M.Dostoievski
Guerra y paz por Leon Tolstoi
PS C:\Users\axeld\Desktop\eda2\Practica56\proyecto1> javadoc -d libreria Libreria.java

```

Se muestra la ejecución del Proyecto 1, además se muestra la creación de los archivos.jar y los javadocs de las clases.

Conclusiones:

En conclusión, es fundamental reconocer la relevancia de los Javadocs como una herramienta poderosa que ofrece múltiples formas de uso. A lo largo de esta práctica, hemos generado diferentes empaquetados y documentación para todas las clases trabajadas a lo largo de las tareas de prácticas, además de crear varios archivos JAR de estas mismas, los cuales resultaron de gran ayuda para nuestros objetivos.

El proceso de familiarizarnos con los pasos necesarios para implementar Javadocs no solo facilitó la creación de la documentación requerida, sino que también mejoró nuestra comprensión del código, su estructura y acceso a través de distintas ubicaciones o direcciones. En resumen, el uso adecuado de Javadocs es una inversión valiosa en la calidad y sustentable de cualquier aplicación.

Además conocer el encapsulamiento, los archivos JAR y los javadocs son clave para realizar la práctica. El encapsulamiento protege los datos y asegura un acceso controlado, los archivos JAR permiten empaquetar y distribuir fácilmente el proyecto, y los javadocs generan una documentación clara que facilita su comprensión y mantenimiento. Esto dandonos una aplicación más segura, organizada y accesible para su uso y distribución.

Referencias:

[1] ChatGPT. [En línea]. Disponible: <https://chatgpt.com/>. [Accedido: Sep. 26, 2024]. Pregunta:

“que es encapsulamiento en java?”

[2] IBM Docs, "Archivos JAR y de clase Java." [En línea]. Disponible:

<https://www.ibm.com/docs/es/i/7.5?topic=platform-java-jar-class-files>. [Accedido: Sep. 26, 2024].

[3] S. Shaio, "javadoc," Oracle Help Center. [En línea]. Disponible:

<https://docs.oracle.com/javase/8/docs/technotes/tools/windows/javadoc.html>. [Accedido: Sep. 26, 2024].