Desarrollo Guía 3

Jullians Mauricio Amado Gutierrez

Tomas Alejandro Santiago Reyes

Juan Esteban Cabal Bautista

Universidad Manuela Beltrán

Taller de programación

Olga Lucia Roa Bohórquez

03 de marzo de 2025

Sesión 1

Crear un Programa denominado Fecha que administre y dispare excepciones en caso de: mes, y/o día no  
válido. Debe anexar el código (como un archivo .zip) creado a la entrega de la guía.  
Crear las tres clases: ExceptoFecha, Fecha y Fechas con el Código que se presenta en las Figuras No. 3, 4 y 5 respectivamente.



Preguntas orientadoras  
¿Cuáles fueron los aprendizajes obtenidos al realizar esta guía?, Liste como mínimo 3 aprendizajes y relaciónelos con su futuro que hacer profesional.

Comprensión a la hora de tener un código y saber para que sirve cada elemento que esta dispuesto a desarrollarse en la guía, ya que esto impulsara a la comprensión de códigos mas extensión y de un peso mayor al trabajar con una compañía.

Adaptación con un código nuevo, el no tener que empezar de 0 a desarrollar todas las ideas que se requieren, sino que, empezar a reutilizar uno ya creado para así desarrollar habilidades sobre el manejo de códigos ajenos.

Lectura del código, no solo es adaptarse y comprender, es saber que hace cada variable, cada elemento que hace que el código funcione, saber que es lo que hace la ejecución, y saber sustentar con buenos argumentos el código que nos dejaron y que en un futuro nos dejaran para sustentarlo y explicarlo a las demás personas

¿Dónde presento mayor dificultad resolviendo la guía? y ¿cómo lo resolvieron? ¿cuáles fueron las estrategias de solución?

No tuvimos ninguna complicación a la hora de resolver la guía, desde el inicio hubo un poco de confusión sobre que tocaba hacer exactamente, pero todo después fluyó de una manera en la que pudimos desarrollar esta guía de laboratorio.

Trabajo autónomo

**1. ¿Por qué son importantes las excepciones en la programación?**

Las excepciones son fundamentales porque permiten detectar y manejar errores de manera estructurada, evitando que los programas se bloqueen inesperadamente. Gracias al manejo de excepciones, las aplicaciones pueden continuar funcionando, proporcionar mensajes de error informativos y gestionar situaciones inesperadas de forma controlada.

**2. ¿Cuáles son las excepciones que cree que son más importantes al momento de programar?**

Al programar en Java, algunas de las excepciones más comunes e importantes son:

* El  **NullPointerException** se produce cuando se intenta usar un objeto que no ha sido inicializado.
* El **ArrayIndexOutOfBoundsException**: Ocurre al acceder a un índice inválido en un array.
* El **IOException**: Relacionada con errores en operaciones de entrada/salida, como manejo de archivos.
* El **SQLException**: Se genera cuando hay errores al interactuar con una base de datos.

**3. ¿Qué tipos de excepciones se pueden presentar en una aplicación Java que realice conexión a bases de datos MySQL?**

Cuando una aplicación Java se conecta a una base de datos MySQL, pueden surgir las siguientes excepciones:

* **SQLException**: Es la excepción principal para errores en la base de datos.
* **SQLTimeoutException**: Indica que una operación en la base de datos ha excedido el tiempo límite.
* **SQLSyntaxErrorException**: Se produce cuando hay errores de sintaxis en las sentencias SQL.
* **SQLNonTransientConnectionException**: Indica problemas de conexión con la base de datos que no son transitorios.

Manejar adecuadamente estas excepciones es crucial para que la aplicación responda correctamente a fallos en la base de datos.

**4. ¿Qué tipos de excepciones se pueden presentar en una aplicación Java que trabaje con Threads y Sockets?**

En aplicaciones Java que usan hilos (Threads) y sockets, algunas de las excepciones más comunes incluyen:

* **InterruptedException**: Se lanza cuando un hilo es interrumpido mientras está en espera o dormido.
* **IOException**: Aparece en operaciones de entrada/salida, como la lectura y escritura en sockets.
* **SocketException**: Indica problemas en el socket, como una conexión restablecida.
* **BindException**: Se produce cuando un socket no puede enlazarse a una dirección o puerto porque ya está en uso.

El manejo de estas excepciones es clave para crear aplicaciones de red estables y eficientes.

**5. ¿Qué diferencia hay entre un Error y una Excepción en Java?**

En Java, tanto los **Errores** como las **Excepciones** son subclases de **Throwable**, pero representan problemas distintos:

* **Error**: Son fallos graves que una aplicación no debería manejar. Están relacionados con problemas en la JVM o el entorno del sistema, como OutOfMemoryError o StackOverflowError.
* **Excepción**: Son eventos que pueden ocurrir durante la ejecución del programa y pueden ser gestionados mediante try-catch. Ejemplos incluyen IOException y SQLException.

Video de explicación del código y trabajo autónomo:  
<https://www.youtube.com/watch?v=43qrfZiecD0>

Link del repositorio:

<https://github.com/JCbral04/Taller_prog_Grupo_5.git>