| **Asignatura** | **Programación II** |
| --- | --- |
| **Carrera** | **Ing. en Informática** |
| **Plan** | **Ajuste 2023** |
| **Ciclo** | **2do** |
| **Cuatrimestre** | **1ero** |
| **Tema/Título** | **Practica 5** |
| **Profesor** | **Adrian Tozzi** |

**Grupo de Trabajo**

| **ID/Matrícula** | **APELLIDO, Nombres** | **Correo Electrónico** |
| --- | --- | --- |
| **000-20-2323** | Cannizzaro, Pablo | Pablom.cannizzaro@comunidad.ub.edu.ar |
| **000-20-1394** | Callizaya, Leandro | [leandroa.callizaya@comunidad.ub.edu.ar](mailto:leandro.callizaya@comunidad.ub.edu.ar) |
| **000-20-1953** | Conde Buades, Joaquín | Joaquin.conde@comunidad.ub.edu.ar |
| **000-20-1147** | Franco Dalla Via Oliveros | Franco.dalla@comunidad.ub.edu.ar |

**Grilla de calificación**

| **Concepto** | **Propuesta** | **Marco Teórico** | **Desarrollo propio** | **Conclusiones** | **Fuentes y Referencias** |
| --- | --- | --- | --- | --- | --- |
| **Sobresaliente (10)** |  |  |  |  |  |
| **Distinguido (9-8)** |  |  |  |  |  |
| **Bueno (7-6)** |  |  |  |  |  |
| **Aprobado (5-4)** |  |  |  |  |  |
| **Insuficiente (3-2-1)** |  |  |  |  |  |
| **Reprobado (0)** |  |  |  |  |  |
| **NOTA** |  | | | | |

| Comentario adicional del Profesor: |
| --- |

**PRÁCTICA 5**

* **Ejercicio 1:**

El código regresa estas líneas por consola:

*El metodo lanzarException*

*La excepcion se manejo en el metodo lanzarExcepcion*

*El metodo noLanzaExacpcion*

Esto ocurre porque al llamar a lanzarExcepcion(), este método captura la excepción lanzada dentro de su propio bloque try-catch, evitando que llegue al manejador en el main. Por lo tanto, el bloque catch del main no se ejecuta. Además, el método noLanzaExcepcion() no genera ninguna excepción, por lo que su bloque try imprime directamente la cadena correspondiente sin activar el bloque catch.

* **Ejercicio 2:**

El código regresa estas líneas por consola:

*El metodo lanzarExcepcion*

*La excepcion se manejo en el main*

*El metodo noLanzaExecpcion*

En este caso, no hay captura de excepción dentro del método lanzarExcepcion, por lo que la excepción viaja directamente al manejador en el main. En el código anterior, el método lanzarExcepcion tenía su propio bloque try-catch, lo que evitaba que la excepción llegara al main.

* **Ejercicio 3:**

1. Una excepción se atrapa con el método **try-catch**.
2. El programador encierra en un bloque **try** el código que puede generar una excepción.
3. El bloque try va seguido por **cero** o más cláusulas catch (No es una obligación que el bloque try tenga un catch, aunque se recomienda).

Verdadero o falso:

1. La clase Exception hereda de la clase Error.

FALSO, la clase Exception hereda de Throwable

1. El espíritu detrás del manejo de excepciones es permitir a los programas atrapar y manejar los errores, en vez de dejar que ocurran y sufrir las consecuencias.

VERDADERO, no manejar excepciones correctamente lleva a dificultades a la hora de hacer debugging para el programador, y confusión de uso para el usuario.

1. Cuando un método lanza una excepción, ésta debe ser atrapada en el mismo método.

FALSO, ya que hay casos donde esto no es cierto. Se puede declarar que un método lanza excepciones (en Java, con la palabra reservada “throws”) y dejar que estas excepciones se manejen en el catch de un método superior.

1. Una excepción puede ser propagada “hacia arriba”.

VERDADERO, se puede hacer que un método inferior lance una excepción, y que se maneje en el catch de un método superior.

1. El código encerrado en un bloque catch no puede ser vacío.

FALSO, por lo menos hasta donde yo lo intenté. Si se puede dejar vacío, pero esto no es recomendable ya que no tiene sentido escribir un bloque catch para dejarlo sin instrucciones. Conviene eliminarlo, o agregarle código.

Ejercicio 5:

// DVD.java

public class DVD {

private String titulo;

private String genero;

public DVD(String titulo, String genero) {

this.titulo = titulo;

this.genero = genero;

}

public String getTitulo() {

return titulo;

}

public String getGenero() {

return genero;

}

}

// GeneroNoEncontradoException.java

public class GeneroNoEncontradoException extends Exception {

public GeneroNoEncontradoException(String mensaje) {

super(mensaje);

}

}

// Coleccion.java

import java.util.ArrayList;

import java.util.List;

public class Coleccion {

private List<DVD> dvds = new ArrayList<>();

public void agregarDVD(DVD dvd) {

dvds.add(dvd);

}

public List<DVD> buscarPorGenero(String genero) throws GeneroNoEncontradoException {

List<DVD> resultado = new ArrayList<>();

for (DVD dvd : dvds) {

if (dvd.getGenero().equalsIgnoreCase(genero)) {

resultado.add(dvd);

}

}

if (resultado.isEmpty()) {

throw new GeneroNoEncontradoException("No se encontraron DVDs del género: " + genero);

}

return resultado;

}

}

// Main.java

import java.util.List;

public class Main {

public static void main(String[] args) {

Coleccion coleccion = new Coleccion();

coleccion.agregarDVD(new DVD("Matrix", "Accion"));

coleccion.agregarDVD(new DVD("Titanic", "Romance"));

coleccion.agregarDVD(new DVD("El Padrino", "Drama"));

try {

List<DVD> comedias = coleccion.buscarPorGenero("Comedia");

for (DVD dvd : comedias) {

System.*out*.println("Encontrado: " + dvd.getTitulo());

}

} catch (GeneroNoEncontradoException e) {

System.*out*.println("Error: " + e.getMessage());

}

}

}