Nicolás Arteaga código 1114541427

| **Throw** | **Palabra clave en Java que indica que el bloque siguiente de código maneja una excepción especificada.** |
| --- | --- |
| **Unchecked exceptions** | Una excepción que se maneja de manera opcional |
| **Excepciones controladas** | Una excepción que se DEBE manejar. |
| **Error** | Indica que hay un problema con la interpretación de su programa. |
| **Excepción** | Evita que el intérprete ejecute el resto del código hasta que éste encuentre una palabra catch. |
| **Syntax error** | Error que indica un problema con el formato de codificación. |
| **Capturar** | Error que se produce mientras se ejecuta el programa, también denominado excepción. |
| **Error de lógica** | Error que se produce como resultado de una lógica de programación incorrecta. |
| **Try/catch block** | Bloque de código que maneja excepciones al tratar casos donde se arroja una excepción. |
| **Runtime error** | Errores que se producen durante el tiempo de ejecución y que pueden ser corregidos o manejados por el código. |

1¿Describa la diferencia entre error de sintaxis, error lógico y excepción?

r/Cuando se produce un error de sintaxis, es fácil detectar el error porque la compilación especifica el tipo de error y la línea en la que se produce el error. Pero identificar un error lógico es difícil porque no hay un mensaje del compilador. La salida es incorrecta, incluso el programa ejecutado. Por lo tanto, el programador debe leer cada declaración e identificar el error por su cuenta

2 ¿ Cuál es el problema del siguiente código? ¡Debe imprimir “ Hello World! " cuatro veces a la pantalla

r/

String str = “Hello World”; R/ POR QUE UN FOR NO DEBE LLEVR UN PUNTO Y COMA Y

for(int i = 0; i < 4; i++){ CUANDO LO COMPILAMOS NOS DA ERROR , Y EL FOR ES K CICLO Y DEBE LLEVAR UN CORCHETE

System.out.println(str);

str+= ”!”;

}

3 ¿Describa una excepción que haya experimentado antes en su programa? Explique de qué manera se podría manejar con un bloque de código try/catch.?

UNA excepción que he tenido antes es tratar de convertir a entero un String que no contiene valores numéricos y tratar de dividir por cero.

Por qué lo puedo mejorar: con lo que lo puedo mejorar es un catch dado un valor que es obligatorio o un requisito poner un valor de numero entero y con método try se hace referencia que tengo que colocar aparte un método que me haga de solo llamar eso números enteros como puede ser nextInt eso me ayudara a codificar el codificar el código y cuando lo imprima no me salga error

**int** num1 = 4;

**int** num2 =0;

**try** {

**int** resultado = num1 / num2;

} **catch** (ArithmeticException e) {

System.***out***.println("No se puede dividir por cero.");

}

4 ¿

A r/ public static void main(String[] args) {

int x = 5;

int y = 10;

int sum = x + y;

System.out.println("el numero es: " + sum)

}

}

Esto es un arreror de sintasis por en la ultima línea le falta un punto y coma

B r/ public static void main(String[] args) {

int x = 10;

int y = 20;

if (x < y) {

System.out.println("x is greater than y");

} else {

System.out.println("y is greater than x");

}

}

}

Esto es un error lógico por que a condición en la línea 5 es x < y, lo que significa que "si x es menor que y, entonces haz esto". Pero el mensaje impreso en la línea 6 dice "x es mayor que y", lo que es incorrecto. Debería decir "x es menor que y". Por lo tanto, el error lógico aquí es que el mensaje impreso no refleja la condición lógica en la línea 5.

C r/ public class ExceptionExample {

public static void main(String[] args) {

try {

int[] numbers = {1, 2, 3};

System.out.println(numbers[3]);

} catch (ArrayIndexOutOfBoundsException e) {

System.out.println("Caught an exception: " + e);

}

}

}

Con el método try y catch

5 ¿ Cuál es la diferencia entre una excepción marcada y una excepción sin marcar?

Cuando ocurre una excepción marcada, la aplicación Java se conecta a un recurso externo. Este recurso puede ser un dispositivo como una impresora. Puede ser un archivo o una base de datos. Por lo tanto, el compilador verifica esas excepciones. La excepción IO es una excepción marcada. Ocurre debido a un error en el dispositivo. Cuando la aplicación accede a un archivo que no existe, causará una excepción FileNotFound. una aplicación se puede conectar a una base de datos como MySQL, Oracle, etc. para almacenar datos y cuando el compilador no verifica las excepciones no marcadas a diferencia de las excepciones marcadas, con las excepciones no marcadas, la aplicación Java no está conectada a un recurso externo como un archivo, una base de datos o un dispositivo.