

Conceptos fundamentales de Java

6-2: Manejo de errores Actividades prácticas

**Objetivos de la lección:**

* Descripción de distintos tipos de errores que pueden ocurrir y cómo se manejan en Java
* Descripción de excepciones utilizadas en Java
* Determinación de las excepciones que se arrojan para una clase base
* Escribir un código para manejar una excepción arrojada mediante el método de una clase base

**Vocabulario:**

Identifique el término correspondiente a cada una de las definiciones a continuación.

|  |  |
| --- | --- |
| Try | Palabra clave en Java que indica que el bloque siguiente de código maneja una excepción especificada. |
| Excepcion opcional | Una excepción que se maneja de manera opcional |
| Excepcion obligatoria | Una excepción que se DEBE manejar. |
| Problema de interpretacion | Indica que hay un problema con la interpretación de su programa. |
| Throw | Evita que el intérprete ejecute el resto del código hasta que éste encuentre una palabra catch. |
| Error de Sintaxis | Error que indica un problema con el formato de codificación. |
| Excepcion en tiempo de ejecucion | Error que se produce mientras se ejecuta el programa, también denominado excepción. |
| Errores logicos | Error que se produce como resultado de una lógica de programación incorrecta. |
| Bloque Catch | Bloque de código que maneja excepciones al tratar casos donde se arroja una excepción. |
| Excepcion de tiempos de ejecución incorregibles | Errores que se producen durante el tiempo de ejecución y que pueden ser corregidos o manejados por el código. |

Copyright © 2020 Oracle y/o sus filiales. Todos los derechos reservados. Oracle y Java son marcas comerciales registradas de Oracle y sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios.

**Inténtelo/resuélvalo:**

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

Los errores de sintaxis son errores básicos en la programación, que pueden ser detectados por el compilador durante el proceso de compilación. Estos errores impiden que el código sea traducido en un programa ejecutable, y deben ser corregidos antes de que el programa pueda ser compilado y ejecutado. Los errores lógicos, por otro lado, pueden ser más difíciles de detectar, ya que el código sigue una estructura correcta, pero no produce los resultados deseados. La corrección de errores lógicos implica revisar el algoritmo utilizado para resolver el problema y asegurarse de que está funcionando de manera adecuada.

Las excepciones, por su parte, son eventos imprevistos que pueden ocurrir durante la ejecución del programa. Estas pueden ser causadas por errores en el código del programa, como una división por cero o un acceso a una variable no inicializada, o por condiciones externas, como una falla en el sistema o una interrupción de la red. Las excepciones deben ser manejadas adecuadamente para evitar que el programa se detenga abruptamente y permitir que el programa siga ejecutándose de manera controlada. El manejo de excepciones puede incluir la captura y el registro de la excepción, la toma de medidas para corregir el problema y la continuación de la ejecución del programa.

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

String str = “Hello World”;

for(int i = 0; i < 4; i++);

{

System.out.println(str); str+= ”!”;

}

R// **El problema del siguiente código es que el for tiene un punto y coma al final, lo que hace que se convierta en un bucle vacío que no ejecuta nada**

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

try {

// Código que procesa el pedido en línea

} catch (IOException e) {

// Manejo de la excepción

System.out.println("Se ha producido un error al procesar su pedido: " + e.getMessage());

System.out.println("Por favor, inténtelo de nuevo más tarde o llame a la pizzería para realizar su pedido.");

}

1. Escriba un segmento de código que tenga:
   1. Un error de sintaxis.

try {

// Punto a: Error de sintaxis

System.out.println("Hola mundo!";

} catch (SyntaxError e) {

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

}

* 1. Un error lógico.

try {

// Punto b: Error lógico

int num1 = 10;

int num2 = 5;

int result = num1 / (num2 - 5);

System.out.println("El resultado es: " + result);

} catch (ArithmeticException e) {

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

}

* 1. Una excepción.

try {

// Punto c: Excepción

Scanner sc = new Scanner(System.in);

System.out.print("Ingrese un número entero: ");

int num = sc.nextInt();

int[] arr = new int[num];

System.out.println("El tamaño del arreglo es: " + arr.length);

} catch (Exception e) {

System.out.println("Ha ocurrido un error: " + e.getMessage());

}

}

}

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

R**// En Java, las excepciones se clasifican en excepciones marcadas y sin marcar. Las excepciones marcadas son aquellas que el compilador obliga a manejar mediante el uso de una declaración throws o un bloque try-catch. Las excepciones sin marcar, por otro lado, no requieren que el programador las maneje explícitamente y se originan típicamente de errores en tiempo de ejecución. La diferencia entre ambas radica en la forma en que se manejan y en si el compilador obliga o no al programador a manejarlas.**

Copyright © 2020, Oracle y/o sus filiales. Todos los derechos reservados. Oracle y Java son marcas comerciales registradas de Oracle y/o sus filiales. Todos los demás nombres pueden ser marcas comerciales de sus respectivos propietarios

2