1Descripción de distintos tipos de errores que pueden ocurrir y cómo se manejan en Java

R//Errores de sintaxis: se producen cuando hay un error en la estructura del código java, como una falta de un paréntesis, un punto y coma faltante o un error de ortografía. El compilador de java detecta estos errores durante la fase de compilación y muestra un mensaje de error en la consola

Excepciones de tiempo de compilación: estas excepciones ocurren durante la compilación y son causadas por problemas en el código que no se pueden detectar hasta que se ejecuta el programa. Estos errores se deben a problemas como la referencia a una variable que no existe, la división por cero, el acceso a una posición de memoria que no se ha asignado

Estas excepciones se manejan mediante declaraciones de excepciones en el método o mediante el uso de bloques try – catch para capturar la excepción y manejarla apropiadamente.

Excepciones de tiempo de ejecución: se producen durante la ejecución del programa y se deben a una variable que no existe, la división por cero, el acceso a una posición de memoria que no se ha asignado se manejan mediante el uso de try o catch o la declaración de excepciones en el metodo

2 Descripción de excepciones utilizadas en Java

R// Excepciones de tiempo de ejecución: son excepciones que se producen durante la ejecución del programa y que no se detectan durante la compilación. Estas excepciones se utilizan para errores como la división por cero, el acceso a una posición de memoria no asignada, o la llamada a un método inexistente.

Excepciones de tiempo de compilación: son excepciones que se detectan durante la compilación y que se deben a problemas en el código. Estas excepciones se utilizan para errores como la referencia a una variable que no existe o la falta de una biblioteca necesaria.

Excepciones de entrada/salida: son excepciones que se producen cuando hay problemas con la lectura o escritura de archivos o con la comunicación con dispositivos externos.

Excepciones personalizadas: son excepciones creadas por el programador para manejar errores específicos en su programa. Estas excepciones se definen mediante la creación de una clase que extiende la clase Exception o RuntimeException.

El manejo de excepciones en Java se realiza mediante el uso de bloques try-catch, que permiten al programador capturar la excepción y manejarla apropiadamente, o mediante la declaración de excepciones en el método, que indica que el método puede lanzar una excepción específica. Además, Java proporciona una jerarquía de clases de excepción que permite al programador manejar excepciones de manera más eficiente y específica.

3 Determinación de las excepciones que se arrojan para una clase base

R// Revisar la documentación de la clase base: La mayoría de las clases en Java tienen una documentación asociada que describe los métodos y excepciones que puede lanzar cada uno. Esta documentación se puede encontrar en la API de Java o en el IDE que se esté utilizando.

Analizar los métodos de la clase base: Revisar los métodos de la clase base y examinar las excepciones que cada uno lanza. Esto se puede hacer buscando los bloques throws en la definición del método.

Verificar las excepciones lanzadas por las clases heredadas: Si la clase base hereda de otra clase, es importante verificar las excepciones que se arrojan en los métodos de la clase heredada. Esto se puede hacer revisando la documentación y los bloques throws en la definición del método.

Utilizar herramientas de análisis de código: Algunos IDE y herramientas de análisis de código, como FindBugs y PMD, pueden analizar el código fuente y detectar las excepciones que se lanzan en la clase base y sus métodos.

|  |  |
| --- | --- |
| Catch | Palabra clave en Java que indica que el bloque siguiente de código maneja una excepción especificada. |
| NullpointerException | Una excepción que se maneja de manera opcional |
| Excepciones verificadas  Y no verificadas | Una excepción que se DEBE manejar. |
| Error de sintaxis | Indica que hay un problema con la interpretación de su programa. |
| Try | Evita que el intérprete ejecute el resto del código hasta que éste encuentre una palabra catch. |
| UnsupportedEncodingException | Error que indica un problema con el formato de codificación. |
| RunimeException | Error que se produce mientras se ejecuta el programa, también denominado excepción. |
| LogicException | Error que se produce como resultado de una lógica de programación incorrecta |
| Try - catch | Bloque de código que maneja excepciones al tratar casos donde se arroja una excepción. |
| Excepciones de java | Errores que se producen durante el tiempo de ejecución y que pueden ser corregidos o manejados por el código. |

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

R// el error de sintaxis es detectado por el compilador durante el proceso de compilación, mientras que el error lógico y las excepciones son detectados durante la ejecución del programa. El error lógico se produce cuando el código no produce el resultado esperado, mientras que la excepción se produce cuando se encuentra una situación excepcional que impide que el programa continúe su ejecución normal.

5 ¿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// En el ciclo for hay dos punto y comas lo cual uno después del cero y el otro después del paréntesis, el sistema de java determina que el ciclo solo llega hasta: for(int i = 0; gracias a este punto y coma lo demás a partir de ahí es considerado fuera del ciclo por eso no se logra imprimir “hello world” cuatro veces

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

R// diferencia entre una excepción marcada y una excepción no marcada en Java radica en el hecho de que las excepciones marcadas deben ser manejadas explícitamente en el código, mientras que las excepciones no marcadas no requieren un manejo explícito. Las excepciones marcadas son excepciones que pueden ocurrir en cualquier momento durante la ejecución del programa y que el programador debe tener en cuenta y manejar de manera adecuada para evitar problemas. Por otro lado, las excepciones no marcadas son excepciones que suceden durante la ejecución del programa y que en su mayoría se deben a errores de programación o de lógica en el código.