Centro Universitario de Ciencias Exactas e Ingenierías

Asignatura: **Computación tolerante a fallas**

Sección: D06

Ejercicio 2: Otras herramientas para manejar errores. Parte 2

Alumno: Luis Jaime Portillo Correa

Código: 217546155

Profesor: **Michel Emanuel López Franco**

Fecha: **24/08/2023**

Objetivo

Generar un reporte con otras herramientas para el manejo de errores en programación.

Desarrollo

Try Catch en C++

En C++, puedes utilizar un bloque try-catch para manejar excepciones cuando intentas convertir una cadena (string) en una variable de tipo entero (int). Esto es útil cuando no estás seguro de si la cadena contiene un valor que se puede convertir en un entero y deseas manejar posibles errores.

```cpp
#include <iostream>
#include <stdexcept> // Necesario para std::invalid_argument

int main() {
    try {
        std::string cadena;
        std::cin >> cadena;
        int numero = std::stoi(cadena); // Intenta convertir la cadena en un entero

        // Si la conversión tiene éxito, imprime el número
        std::cout << "Número convertido: " << numero << std::endl;
    } catch (const std::invalid_argument& e) {
        // Captura una excepción si la conversión falla debido a un argumento no válido
        std::cerr << "Error de conversión: " << e.what() << std::endl;
    } catch (const std::out_of_range& e) {
        // Captura una excepción si la conversión falla debido a un desbordamiento de rango
        std::cerr << "Error de rango: " << e.what() << std::endl;
    }

    return 0;
}
```

En el primer caso de prueba traté de convertir un string con un “Hola mundo” a entero, el programa detectó la excepción y pudo terminar la ejecución sin errores o warnings.

```
Hola mundo
Error de conversión: stoi


...Program finished with exit code 0
Press ENTER to exit console.
```

Para el segundo caso, puse un string numérico, pero con un rango bastante amplio que no alcanza a formar parte de un Int. El programa no sufrió interrupciones gracias al try catch.

```
100000000000000000000000000
Error de rango: stoi


...Program finished with exit code 0
Press ENTER to exit console.
```

Try-Catch en Java

Este programa permite al usuario ingresar un número, pero si el usuario ingresa algo que no es un número válido (como una letra o un símbolo), el bloque catch manejará la excepción y mostrará un mensaje de error, evitando que el programa se detenga de manera inesperada.

```java
import java.util.Scanner;

public class EjemploTryCatch2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        try {
            System.out.print("Ingrese un numero: ");
            int numero = sc.nextInt(); // Intenta leer un
                entero del usuario

            System.out.println("Numero ingresado: " + numero);
        } catch (java.util.InputMismatchException e) {
            System.err.println("Error: Ingrese un numero entero
                valido.");
        }

        sc.close();
    }
}
```

En el primer caso de prueba ingresamos un valor float, normalmente el programa dejaría de funcionar completamente pues estaría esperando un Int, pero con el manejo de excepciones logra terminar la ejecución correctamente, aunque no logre los resultados esperados.

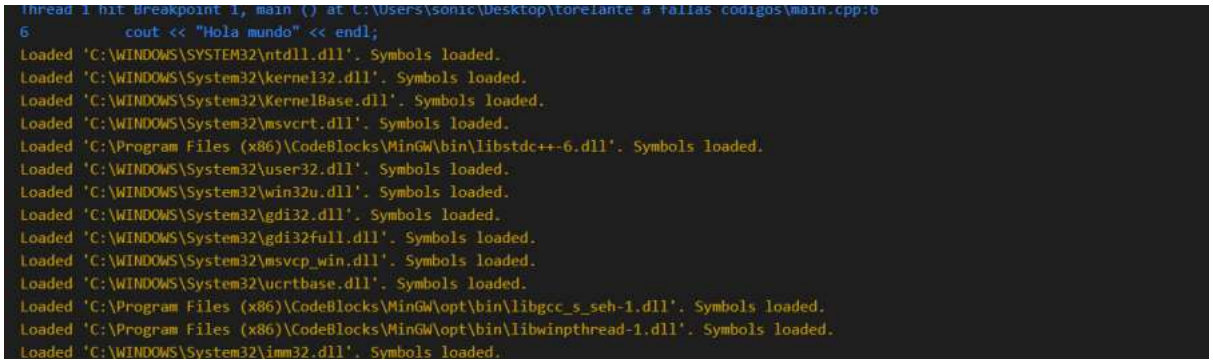
```
java -cp /tmp/jdah33Ko8N EjemploTryCatch2
Ingrese un numero: 15.874
ERROR!
Error: Ingrese un numero entero valido.
```

Para el segundo caso de prueba ingresaré una cadena de caracteres, en este caso el programa también tronaría directamente pues está esperando un entero, pero gracias a las excepciones logra terminar la ejecución correctamente.

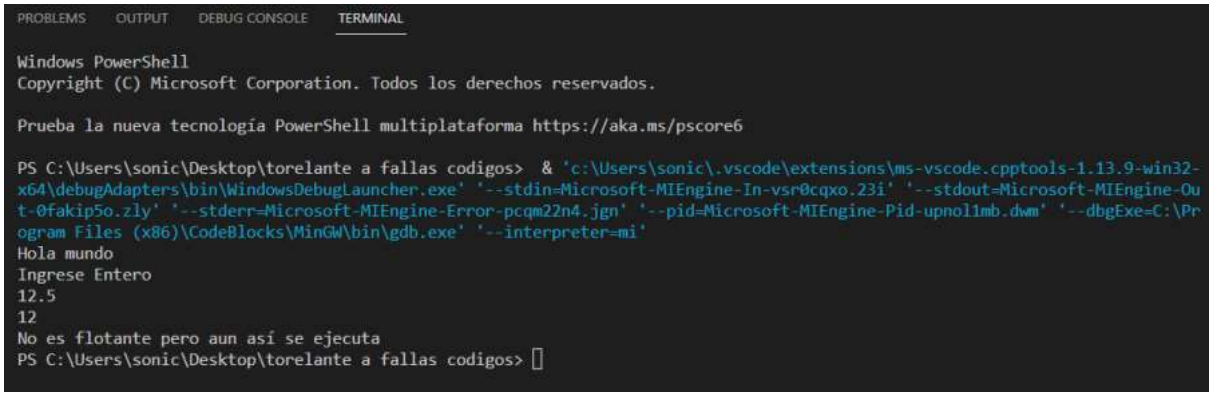
```
java -cp /tmp/jdah33Ko8N EjemploTryCatch2
Ingrese un numero: Hola mundo
ERROR!
Error: Ingrese un numero entero valido.
```

Veracode

Este programa se puede incorporar en Visual Studio Code y te dirá tus errores y posibles soluciones y los podrá ejecutar, aunque no sea los datos que ingresemos.



En esta parte es donde muestra los errores, en este caso fue en el hola mundo, pero aun así se ejecuta.



En este caso se ejecuta, e igual que en algunos ejemplos anteriores pide un número entero en C++, la diferencia es que al ingresar un flotante, la extensión de VS Code detectará el error en la entrada de datos y la corregirá. En la captura se observa que se ingresó un flotante, la extensión detectó la excepción y finalmente lo corrigió por un entero.

Conclusión

Considero que este tipo de técnicas para el control de excepciones y errores que hemos visto desde la actividad pasada son de vital importancia para el correcto funcionamiento de un sistema, de manera que unos errores sencillos no puedan tronar todo el funcionamiento.

También es importante mencionar que este tipo de herramientas son bastante básicas a la hora de programar, es decir, no pueden faltar en cualquiera de nuestros programas que pretendamos que tenga un correcto funcionamiento. Es muy probable que en la industria existan distintas herramientas mucho más complejas para evitar errores catastróficos en los sistemas y espero poco a poco irlas conociendo, ya que la tolerancia a fallas es de vital importancia en los sistemas computacionales.

Bibliografía

- https://github.com/LujaMX/Tolerancia-a-fallas/tree/main/Codigo/Ejercicio%203