CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERÍAS

COMPUTACIÓN TOLERANTE A FALLAS
I7036
SECCIÓN D06

Otras herramientas para el manejar errores Parte 1

BAUTISTA MARTINEZ EDEN ABDIEL
CÓDIGO: 221350524

PROFESOR: MICHEL EMANUEL LOPEZ FRANCO

Herramientas para el manejo de errores en C++, Java y C#

C++

Excepciones: En C++, la manera preferida de notificar y controlar los errores lógicos y los errores en tiempo de ejecución es usar excepciones. Las excepciones proporcionan una forma formal y bien definida para que el código que detecta errores pase la información a la pila de llamadas.

```
#include <iostream>
#include <stdexcept>

void divide(int a, int b) {
    if (b == 0) {
        throw std::runtime_error("Division por cero");
    }
    int result = a / b;
    std::cout << "Resultado: " << result << std::endl;
}

int main() {
    try {
        divide(10, 0);
    } catch (const std::exception& e) {
        std::cerr << "Error: " << e.what() << std::endl;
    }
    return 0;
}
```

Depuradores: Los depuradores son herramientas que permiten ver el estado del programa en tiempo real y analizar el flujo de ejecución del código, lo cual es muy útil a la hora de corregir errores.

```cpp
#include <iostream>



int main() {

    int a = 5;

    int b = 0;



    // Punto de ruptura para el depurador

    std::cout << "Antes de la division" << std::endl;



    // Depurador mostrara el estado actual de las variables y permitirá

    // inspeccionar el flujo de ejecución paso a paso.

    int result = a / b;



    std::cout << "Después de la division" << std::endl;



    return 0;

}
```

Java

Excepciones: En Java, las excepciones se utilizan para manejar situaciones excepcionales o de error que pueden ocurrir durante la ejecución de un programa. Se basan en tres palabras clave principales: try, catch y throw.

```java
public class DivisionExample {

    public static void main(String[] args) {

        try {

            divide(10, 0);

        } catch (ArithmeticException e) {
```

```
            System.err.println("Error: " + e.getMessage());
        }
    }

    static void divide(int a, int b) {
        if (b == 0) {
            throw new ArithmeticException("Division por cero");
        }
        int result = a / b;
        System.out.println("Resultado: " + result);
    }
}
```

Checkmarx, SonarQube y FindBugs: Estas herramientas inspeccionan el código en busca de patrones problemáticos, añadiendo un nivel de revisión adicional.

C#

Excepciones: En C#, las instrucciones throw y try se usan para trabajar con excepciones. Se utiliza la instrucción throw para producir una excepción y la instrucción try para detectar y controlar las excepciones que pueden producirse durante la ejecución de un bloque de código.

```
using System;


class Program
{
    static void Main()
    {
        try
        {
            Divide(10, 0);
```

```
        }
        catch (Exception ex)
        {
            Console.WriteLine("Error: " + ex.Message);
        }
    }

    static void Divide(int a, int b)
    {
        if (b == 0)
        {
            throw new DivideByZeroException("Division por cero");
        }
        int result = a / b;
        Console.WriteLine("Resultado: " + result);
    }
}
```

Conclusión

Los tres lenguajes anteriores, proporcionan mecanismos para el manejo de errores. Las excepciones son comunes en C++ y Java, mientras que en C# también se utilizan instrucciones especiales para trabajar con excepciones. Además, la capacidad de utilizar depuradores y herramientas de análisis estático contribuye significativamente a la identificación y corrección de errores durante el desarrollo.

Referencias

BillWagner. (2023, 5 junio). *Instrucciones de control de excepciones: throw y try, catch, finally - C#*. Microsoft Learn. https://learn.microsoft.com/es-es/dotnet/csharp/language-reference/statements/exception-handling-statements

Ciberaula. (s. f.). *Manejo de errores y excepciones en Java*. https://www.ciberaula.com/cursos/java/manejo_errores_excepciones_java.php

TylerMSFT. (2023, 3 abril). *Procedimientos recomendados de C++ moderno para el control de errores y excepciones*. Microsoft Learn. https://learn.microsoft.com/es-es/cpp/cpp/errors-and-exception-handling-modern-cpp?view=msvc-170