UNIVERSIDAD DE GUADALAJARA

CENTRO UNIVERSITARIO DE CIENCIAS EXACTAS E INGENIERIAS

Materia: Computación tolerante a fallas

Clave: I7036

Sección: D06

NRC: 179961

Actividad 3

Código estudiante: 221350567

Alumno: Rauf Alfonso Hamden Estrada

Carrera: Ingeniería en Computación

Fecha: 27/08/2023

Docente: Michel Emanuel Lopez Franco

2023B

Introducción

La programación es una disciplina que implica la creación y el mantenimiento de software, y a medida que los programas se vuelven más complejos, la gestión de errores y la validación de datos se vuelven esenciales para garantizar que haya menos fallos en las aplicaciones o softwares. Se utilizan diversas herramientas y técnicas para abordar estos desafíos.

Código de try..catch en java

Explicación:
El código detectara que el usuario le esta enviando un valor de cero por lo cual es algo que no se puede en una división el cual será un dato invalido , en caso contrario de que ingrese un valor que no sea cero le dará el resultado sin ningún problema

Código:

```
public static int dividir(int dividendo, int divisor) {
        return dividendo / divisor;
    }

try {

    // Código que podría lanzar una excepción

    int resultado = dividir(10, 0); // Intentar dividir por cero

    System.out.println("Resultado: " + resultado);

} catch (ArithmeticException e) {

    // Manejar la excepción

    System.err.println("Error: No se puede dividir por cero.");

    e.printStackTrace(); // Imprimir la excepción
}
```

Ejemplo de ejecución del código de forma correcta:

Si a mi función de dividir le mando cualquier numero que no sea 0 dividir(10, 10)

```
Resultado: 1
```

Ejemplo de ejecución del código de forma incorrecta:

Si a mi función de dividir le mando un 0 dividir(10, 0)

```
Error: No se puede dividir por cero.
```

Código en Python:

El código lee un correo electrónico y la librería detecta si el correo ingresado anteriormente es valido o no

```
from validators import email

email_address = "pesopluma@hotmail.com"


if email(email_address):
    print("La direccion de correo electronico es valida.")
else:
    print("La direccion de correo electronico no es valida.")
```

Ejemplo de ejecución del código de forma correcta:

Ejemplo de ejecución del código de forma incorrecta :

Validación de números en c++

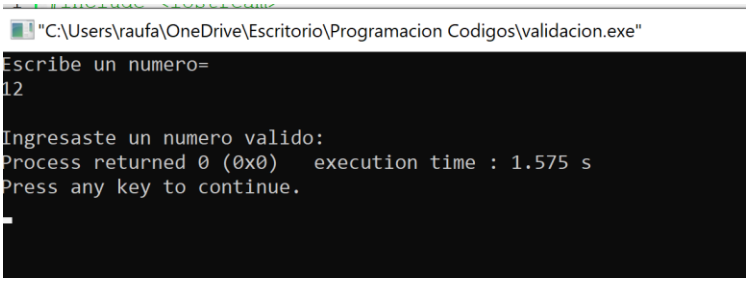
Explicación:

El código solamente recibirá números en caso de que detecte algún otro valor que no sea un numero le pedirá al usuario que ingrese otro nuevamente sin que el programa se detenga

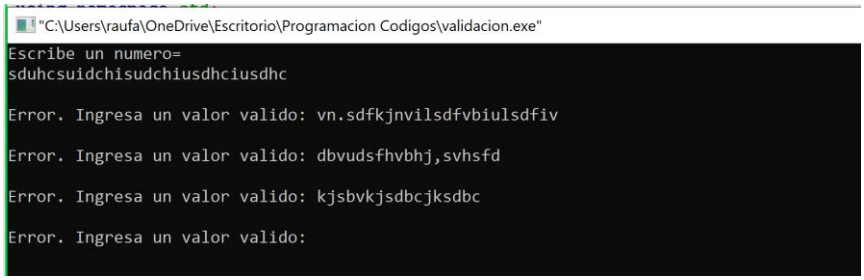
Código:

```
#include <iostream>
#include <cstdlib>
using namespace std;

bool validar(char *num) {
   // Verifica si el caracter esta en el rango de caracteres numericos en la tabla ASCII
   if ((char)*num >= 48 && (char)*num <= 57) {
      system("cls"); // Limpia la pantalla de la consola
      return true; // El caracter es un numero valido
   } else {
      cout << "\nError. Ingresa un valor valido: "; // Mensaje de error
      return false; // El caracter que se ingreso
   }
}

char valorChar[100], *ptrChar;
bool Validar_Numeros(int *dato);
int *ptr = NULL;
int main(int argc, char **argv) {
   cout << "Escribe un numero=" << endl;
   do {
      cin >> valorChar;
      cin.ignore(); // Limpia el bufer del teclado
      ptrChar = &valorChar[0]; // Asigna la direccion del primer caracter al puntero
   } while (!validar(ptrChar));

   return 0;
}
```

Ejemplo de ejecución del código de forma correcta:



Ejemplo de ejecución del código de forma incorrecta:



Conclusiones

El manejo de errores en programación es esencial para garantizar aplicaciones de alta calidad y confiabilidad. Las herramientas para la detección de errores como el try catch es uno de los mas utilizados y más eficaz, sin embargo , con las bibliotecas de python es aun mas sencillo ya que Python es un lenguaje que tiene muchas librerías que nos podrían ayudar a detectar errores y la validación de errores es algo que yo suelo utilizar para evitar que el programa cierre cuando el usuario ponga un carácter incorrecto. La elección de la herramienta adecuada depende de las necesidades específicas del proyecto y del lenguaje de programación utilizado.

Enlaces

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

Kantor, I. (s. f.). Manejo de errores, «Try. . .catch». https://es.javascript.info/try-catch

Errores y excepciones. (s. f.). Python documentation. https://docs.python.org/es/3/tutorial/errors.html

Corredera, P. Á. (2022, 27 diciembre). 5 herramientas para verificar tu código en Python. CIBERNINJAS. https://ciberninjas.com/python-5-herramientas-limpiar-codigo/