Skillnest

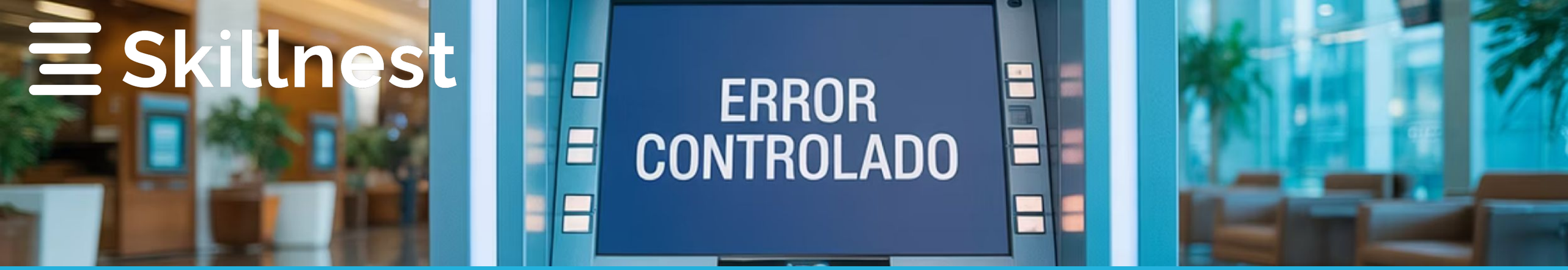
Bienvenida y título de la clase

Módulo: Programación Avanzada en Python

Aprendizaje Esperado Nº5 (AE5): ***"Utilizar sentencias de captura y generación de excepciones para el control del flujo de un programa acorde al lenguaje Python"***

Hoy: ***Manejo de Excepciones en Python*****.**

Skillnest



Aprendizaje Esperado N°5

Descripción completa: AE5 busca que aprendamos a:

1. **Capturar y generar excepciones.**
2. **Controlar el flujo de un programa evitando que falle inesperadamente.**
3. **Mejorar la experiencia del usuario al manejar errores de forma clara y segura.**

Aplicación: Evitar que un sistema de pago online (como Transbank) se caiga por un error en la entrada del RUT o un fallo en la conexión.

Skillnest

Temas del AE5 (visión general)

¿Qué es una excepción?	Tipos de excepciones	Sentencia try/except
Captura múltiple	Lanzamiento de excepciones (raise)	Excepciones personalizadas
Limpieza (finally y with)		

Otros temas futuros: herencia, encapsulamiento, polimorfismo (jueves y viernes).

¿Qué es una excepción?

Skillnest

Una excepción es un evento inesperado que interrumpe el flujo normal del programa. Ejemplos cotidianos en Chile:

- Dividir entre cero al calcular promedio PSU.
- Intentar abrir un archivo "notas2025.csv" que no existe.
- Ingresar texto en vez de un número al calcular el IVA (19%).

Skillnest

Tipos comunes de excepciones

ZeroDivisionError

Dividir por cero.

ValueError

Formato inválido (ej.: "hola" en vez de 500).

TypeError

Operaciones con tipos incorrectos.

FileNotFoundError

Archivo inexistente.

IndexError

Índice fuera de rango en una lista.

Pregunta para reflexionar: ¿Cuál de estas creen que ocurre más al programar sistemas contables en Chile?

try/except – la base del manejo de excepciones

```
try:
    precio = int(input("Ingrese precio del
producto: "))
    print(1000 / precio)
except ZeroDivisionError:
    print("El precio no puede ser cero")
```

Explicación:

- **try: código que podría fallar.**
- **except: cómo reaccionar si ocurre un error.**

Skillnest

Captura de múltiples excepciones

Skillnest

```
try:
    cantidad = int(input("Cantidad de entradas: "))
    print(2000 / cantidad)
except ValueError:
    print("Debe ingresar un número válido")
except ZeroDivisionError:
    print("La cantidad no puede ser cero")
```

Aplicación: Venta de entradas para un partido de la Roja, manejar errores sin que se caiga el sistema.

Captura múltiple en una sola línea

```
try:
    cantidad = int(input("Cantidad: "))
    print(2000 / cantidad)
except (ValueError, ZeroDivisionError):
    print("Ocurrió un error en la entrada")
```

Reflexión: ¿En qué casos conviene agrupar excepciones?

Lanzar excepciones con raise

```
def dividir(a, b):
    if b == 0:
        raise ZeroDivisionError("No se puede dividir por
    cero")
    return a / b
```

Aplicación: Un sistema de boletas electrónicas lanza error si el subtotal es cero.

Crear excepciones personalizadas

Skillnest

```
class SaldoInsuficiente(Exception):
    pass

def retirar(saldo, monto):
    if monto > saldo:
    raise SaldoInsuficiente("Saldo insuficiente")
```

Pregunta: ¿Qué otros errores personalizados serían útiles en un sistema bancario?

Limpieza con finally

```
try:
    archivo = open("ventas.txt", "r")
    contenido = archivo.read()
except FileNotFoundError:
    print("Archivo no encontrado")
finally:
    archivo.close()
```

Explicación: Siempre se ejecuta, ocurra o no una excepción.

Limpieza automática con with

```
with open("ventas.txt", "r") as archivo:
    contenido = archivo.read()
```

Ventaja: Python cierra el archivo automáticamente.

Aplicación: Manejo seguro de reportes del SII.

Mini desafío (5 minutos)

Crea una función leer_rut(nombre_archivo) que:

- **Abra el archivo con with.**
- **Lance excepción si no existe.**
- **Lance excepción personalizada si algún RUT tiene formato inválido.**

Dinámica lúdica

Trivia:

1. **¿Qué hace finally?**
2. **¿Qué excepción se lanza al dividir por cero?**
3. **¿Qué ventaja tiene with?**

Skillnest

Actividad final

Reto: Crear clase CuentaBancaria:

- **depositar() y retirar().**
- **Excepción personalizada si el saldo es insuficiente.**
- **Registro automático en archivo usando with.**

Skillnest

Estrategias robustas de manejo de errores

- No ocultar errores, explicarlos al usuario.
- Liberar recursos siempre.
- Anticipar casos comunes (ej.: cortes de luz en regiones).

Beneficios de manejar excepciones

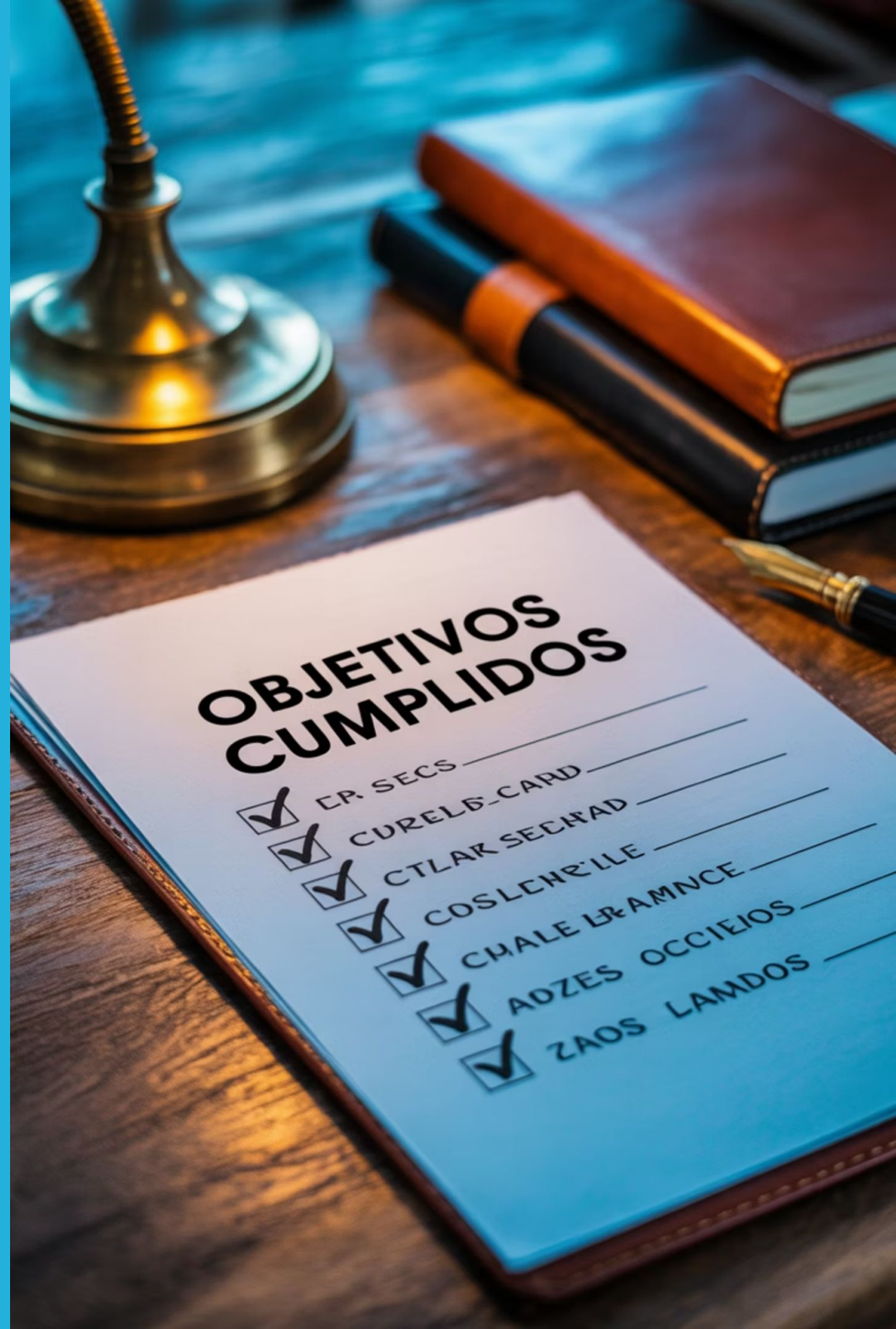
Skillnest

- **Control del flujo**
- **Mejor experiencia del usuario**
- **Depuración más fácil**

Skillnest

¿Qué aprendimos hoy?

- Qué es una excepción y sus tipos.
- Cómo manejarlas (try/except).
- Lanzar y crear excepciones personalizadas.
- Limpieza con finally y with.



Próximos pasos

Martes: Ejercicio Grupal y Quiz.

Miércoles: AE6 - Manipulación de archivos.

Skillnest

Cierre

> **"En programación, los errores no son obstáculos: son señales que nos enseñan a mejorar."**