Manejo de excepciones: try, except, finally (1 hora)

El manejo de excepciones es fundamental para prevenir errores en tiempo de ejecución y mantener la estabilidad de un programa. Python proporciona bloques `try`, `except`, y `finally` para manejar estos casos.

try-except

El bloque `try-except` permite capturar y manejar excepciones que puedan ocurrir durante la ejecución de un programa.

```
try:
    numero = int(input("Ingrese un número: "))
    print("El número ingresado es:", numero)
except ValueError:
    print("Ha ocurrido un error. Por favor, ingrese un número válido.")
```

try-except-else

El bloque `try-except-else` permite ejecutar un fragmento de código si no se produce ninguna excepción dentro del bloque `try`.

```
try:
    numero = int(input("Ingrese un número: "))
except ValueError:
    print("Ha ocurrido un error. Por favor, ingrese un número válido.")
else:
    print("El número ingresado es:", numero)
```

try-except-finally

El bloque `try-except-finally` permite ejecutar un fragmento de código sin importar si se produce o no una excepción dentro del bloque `try`.

```
try:
    numero = int(input("Ingrese un número: "))
except ValueError:
    print("Ha ocurrido un error. Por favor, ingrese un número válido.")
finally:
    print("Fin del programa.")
```

Capturando múltiples excepciones

Es posible capturar y manejar diferentes tipos de excepciones utilizando varios bloques `except`.

```python
try:
    numerador = int(input("Ingrese el numerador: "))
    denominador = int(input("Ingrese el denominador: "))
    resultado = numerador / denominador
    print("El resultado es:", resultado)
except ValueError:
    print("Ha ocurrido un error. Por favor, ingrese un número válido.")
except ZeroDivisionError:
    print("No se puede dividir por cero.")
```

Capturando excepciones y accediendo a la información de error

Al capturar una excepción, es posible acceder a información adicional sobre el error utilizando la palabra clave `as`.

```python
try:
    numero = int(input("Ingrese un número: "))
    resultado = 100 / numero
    print("El resultado es:", resultado)
except (ValueError, ZeroDivisionError) as e:
    print(f"Ha ocurrido un error: {e}")
```

Levantando excepciones

En ocasiones, es necesario generar excepciones de manera manual utilizando la palabra clave `raise`.

```python
def validar_edad(edad):
    if edad < 0:
        raise ValueError("La edad no puede ser negativa.")

try:
    edad = int(input("Ingrese su edad: "))
    validar_edad(edad)
except ValueError as e:
    print(f"Error: {e}")
```

Con estas notas y ejemplos, los estudiantes podrán comprender y aplicar el manejo de excepciones en Python, incluyendo el uso de bloques `try`, `except`, `else` y `finally`, la captura de múltiples excepciones y la generación manual de excepciones. Estas habilidades son esenciales para desarrollar programas robustos y resistentes a errores en tiempo de ejecución.