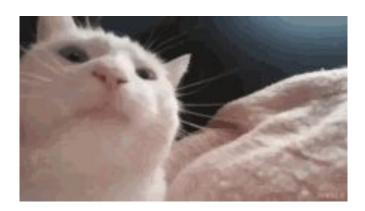
Programación Avanzada IIC2233 2025-2

Cristian Ruz - Pablo Araneda - Francisca Ibarra - Tamara Vidal - Daniela Concha

Anuncios

11 de septiembre de 2025



- 1. Hoy tendremos la Experiencia 2.
- Recuerden que la Tarea 2 se entrega mañana (viernes).
- Recuerden que la ECA abre de domingo a martes. Si responden 10 veces tienen un bonus en el promedio final.

Excepciones



Excepciones

- ¿Qué son las Excepciones?
- ¿Qué tipos de Excepciones se han enfrentado hasta el momento?
 - EOFError
 - TypeError
 - SyntaxError
 - IndentationError
 - NameError

- IndexError
- KeyError
- AttributeError
- ValueError

Levantamiento de Excepciones



Levantamiento de Excepciones

- ¿Cómo se levanta una Excepción?
- ¿Qué pasa cuando se levanta una Excepción?
- ¿Qué puede recibir como *input* una Excepción?

Ejemplo Traceback

main.py def func1(): return func2() def func2(): return func3() 6. def func3(): result = 10 / 0return result 10. 11. if __name__ == '__main__': 13. func1() 14.

Terminal

```
> python main.py
Traceback (most recent call last):
  File "main.py", line 13, in <module>
     func1()
File "main.py", line 2, in func1
      return func2()
            \Lambda \Lambda \Lambda \Lambda \Lambda \Lambda
  File "main.py", line 5, in func2
      return func3()
            \Lambda \Lambda \Lambda \Lambda \Lambda \Lambda
  File "main.py", line 8, in func3
      result = 10 / 0
           ~~~^~~
ZeroDivisionError: division by zero
```

Ejemplo raise

10.

11.

except ValueError as e:

print(f" (e)")

```
main.py
    def dividir(a: int, b: int) -> int:
        if b == 0:
             raise ValueError("No se puede dividir por cero")
        return a / b
5.
    numeros = [(10, 2), (5, 0), (8, 4)]
                                                         Terminal
    for x, y in numeros:
8.
                                                        > python main.py
        try:
                                                        10 \div 2 = 5.0
             print(f''\{x\} \div \{y\} = \{dividir(x, y)\}'')
```

```
10 ÷ 2 = 5.0

No se puede dividir por cero
8 \div 4 = 2.0
```

Manejo de Excepciones



Manejo de Excepciones

- ¿Cómo se maneja una Excepción?
- En Python, ¿es posible manejar todas las Excepciones?
- ¿Qué pasa con el flujo del programa cuando se hace try/except?
- ¿Cuántas Excepciones puede atrapar un bloque try/except?
- ¿Cómo complementan las sentencias else y finally un bloque try/except?
- ¿Es buena práctica hacer "except Exception:" o solo "except:"?
- ¿Es mejor usar if/else o try/except?

Ejemplo estilos de código

LBYL

Look Before You Leap
Mira antes de saltar

EAFP

Easier to Ask Forgiveness than Permission Mejor pedir perdón que permiso

```
archivo = 'archivo_inexistente.txt'

try:
    with open(archivo) as file:
        texto = file.readlines()
except FileNotFoundError as error:
    print(f'No existe {error.filename},
        agréguelo antes de volver a
        ejecutar el programa.')
```

Pregunta Evaluación Escrita

(Midterm 2023-2)

20. Tomando en cuenta el siguiente código, al levantar una excepción con el comando **raise**, es correcto afirmar que:

```
print("Comienza")
raise ValueError("Mensaje")
print("Error levantado")
print("Fin del código")
```

- A) El error es manejado automáticamente, se imprime el mensaje y el programa continúa su ejecución sin interrupción en la siguiente línea.
- B) El programa pausa su ejecución y se reinicia.
- C) No se ejecuta la siguiente línea del código automáticamente, sino que se detiene en el punto donde se levantó la excepción y finaliza.
- D) El flujo de ejecución del programa se mueve automáticamente a la última línea del programa, ignorando el código restante.
- E) El código completo no se ejecuta.

(Midterm 2023-2)

20. Tomando en cuenta el siguiente código, al levantar una excepción con el comando **raise**, es correcto afirmar que:

```
print("Comienza")
raise ValueError("Mensaje")
print("Error levantado")
print("Fin del código")
```

- A) El error es manejado automáticamente, se imprime el mensaje y el programa continúa su ejecución sin interrupción en la siguiente línea.
- B) El programa pausa su ejecución y se reinicia.
- C) No se ejecuta la siguiente línea del código automáticamente, sino que se detiene en el punto donde se levantó la excepción y finaliza.
- D) El flujo de ejecución del programa se mueve automáticamente a la última línea del programa, ignorando el código restante.
- E) El código completo no se ejecuta.

(Examen 2024-1)

15. Considere el siguiente código en python. Sin considerar saltos de línea, ¿qué imprime este código?

```
try:
    resultado = 10 / 0
    print("TRY")
except ZeroDivisionError:
    print("EXCEPT")
else:
    print("ELSE")
finally:
    print("FINALLY")
EXCEPT
         FINALLY
ELSE
         FINALLY
EXCEPT FINALLY
                  TRY
TRY
         EXCEPT FINALLY
TRY
         ELSE FINALLY
```

(Examen 2024-1)

15. Considere el siguiente código en python. Sin considerar saltos de línea, ¿qué imprime este código?

```
try:
    resultado = 10 / 0
    print("TRY")
except ZeroDivisionError:
    print("EXCEPT")
else:
    print("ELSE")
finally:
    print("FINALLY")
EXCEPT
        FINALLY
ELSE
         FINALLY
EXCEPT
         FINALLY
                  TRY
TRY
         EXCEPT
                 FINALLY
TRY
         ELSE FINALLY
```

Experiencia



Programación Avanzada IIC2233 2025-2

Cristian Ruz - Pablo Araneda - Francisca Ibarra - Tamara Vidal - Daniela Concha