



Gobierno del
CHACO

Ministerio
de la Producción y el Desarrollo
Económico Sostenible



INFORMATARIO

Informatorio Chaco | 2024 | Data Analytics

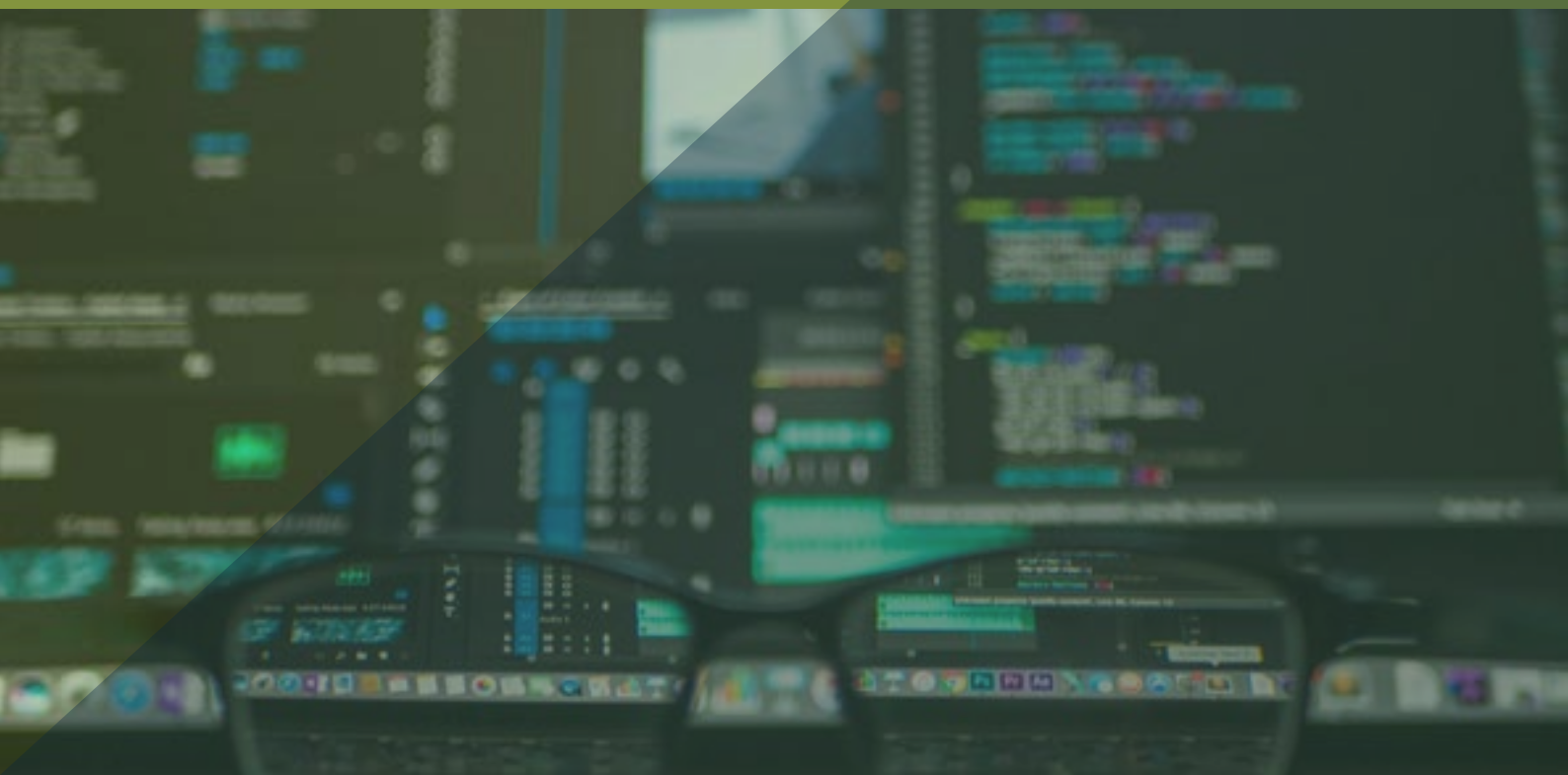
Clase 2

CLASE 2: FUNCIONES, MANEJO DE ERRORES Y EXCEPCIONES, MA- NEJO DE ARCHIVOS

*Funciones | Parámetros y argumentos | Ma-
nejo de errores y excepciones | Manejo de
archivos | Uso de contexto.*

módulo

2



Funciones

Definición y Uso de Funciones

Las funciones permiten agrupar un conjunto de instrucciones bajo un nombre, y pueden ser reutilizadas en diferentes partes del programa.

Sintaxis básica:

```
python Copiar código

def nombre_funcion(parametros):
    # Código de la función
    return resultado
```

Ejemplo:

```
python Copiar código

def suma(a, b):
    return a + b


resultado = suma(3, 5)
print(resultado) # Salida: 8
```

Parámetros y Argumentos

- **Parámetros:** Variables definidas en la declaración de la función.
- **Argumentos:** Valores pasados a la función cuando se llama.

Ejemplo:

python

 Copiar código


```
def saludar(nombre):  
    print(f"Hola, {nombre}")  
  
saludar("Ana") # Salida: Hola, Ana
```

Parámetros y Argumentos

Las funciones lambda son funciones anónimas de una sola línea.

Sintaxis


python

 Copiar código

```
lambda parametros: expresion
```

Ejemplo:

python

 Copiar código

```
suma = lambda a, b: a + b  
print(suma(3, 5)) # Salida: 8
```


Manejo de Errores y Excepciones

Bloques Try-Except

Los bloques try-except permiten manejar excepciones (errores) en el código.

Sintaxis básica:

python

 Copiar código

```
try:
    # Código que puede causar una excepción
except TipoDeExcepcion as e:
    # Código que se ejecuta si ocurre una excepción
else:
    # Código que se ejecuta si no ocurre una excepción
finally:
    # Código que se ejecuta siempre
```

Ejemplo:

python

 Copiar código

```
try:
    resultado = 10 / 0
except ZeroDivisionError as e:
    print("Error: División por cero")
else:
    print("La división se realizó correctamente")
finally:
    print("Bloque finally ejecutado")
```

Tipos de Excepciones Comunes

- ▷ **ValueError:** Se produce cuando una función recibe un argumento con el tipo correcto pero con un valor inapropiado.
- ▷ **TypeError:** Se produce cuando una operación o función se aplica a un objeto de un tipo inapropiado.
- ▷ **IndexError:** Se produce cuando se intenta acceder a un índice que está fuera del rango de una secuencia.
- ▷ **KeyError:** Se produce cuando se intenta acceder a una clave que no existe en un diccionario.


Manejo de Archivos

Operaciones Básicas

Python proporciona funciones para trabajar con archivos. Las operaciones más comunes son abrir, leer, escribir y cerrar archivos.

Abrir un archivo:


python

 Copiar código

```
f = open('archivo.txt', 'r') # Modo lectura
f = open('archivo.txt', 'w') # Modo escritura
f = open('archivo.txt', 'a') # Modo anexar
```

Leer un archivo:

python

 Copiar código

```
contenido = f.read()      # Lee todo el contenido del archivo
linea = f.readline()      # Lee una sola línea del archivo
lineas = f.readlines()    # Lee todas las líneas del archivo y las devuelve como una
```

Escribir en un archivo:


python

 Copiar código

```
f.write('texto')
```

Cerrar un archivo:

python

 Copiar código


```
f.close()
```

Uso de Contexto

La declaración **with** se utiliza para manejar archivos de manera más segura, asegurando que el archivo se cierre correctamente después de su uso.

Sintaxis:


python

 Copiar código

```
with open('archivo.txt', 'r') as f:  
    contenido = f.read()
```

Ejemplo:

python

 Copiar código

```
with open('archivo.txt', 'w') as f:  
    f.write('Hola, mundo')
```