SESIÓN FUNCIONES Y MÓDULOS EN PYTHON

CONTENIDOS:

- ¿Qué es una función y para qué sirve?
- Funciones preconstruidas de Python.
- Funciones personalizadas: definición, parámetros y retorno.
- Utilización de una función personalizada.
- Qué es un módulo y para qué sirven.
- Librería estándar de Python: math y stat.
- Importación de módulos.

¿QUÉ ES UNA FUNCIÓN Y PARA QUÉ SIRVE?

Una función es un bloque de código que realiza una tarea específica. Sirve para organizar el código, evitar repeticiones y mejorar la legibilidad, ya que se puede reutilizar en distintas partes del programa cuando sea necesario.

Ventajas de usar funciones:

- Facilita la organización y claridad del código.
- Permite la reutilización del código.
- Facilita la detección de errores.

```
def saludar():
    print("¡Hola!")
saludar()  # Esto imprime: ¡Hola!
```

Ilustración 1 Función saludar y llamada a la función

FUNCIONES PRECONSTRUIDAS DE PYTHON

Las funciones preconstruidas de Python son herramientas listas para usar que facilitan tareas comunes sin necesidad de escribir código adicional. Algunas de las más comunes incluyen print()

para mostrar datos, len() para medir longitudes, y type() para conocer el tipo de un dato. Simplifican y optimizan el desarrollo.

```python
texto = "Python"
print(len(texto))  # Resultado: 6
```

Ilustración 2 Uso de función print() y len()

FUNCIONES PERSONALIZADAS: DEFINICIÓN, PARÁMETROS Y RETORNOS

Las funciones personalizadas en Python son bloques de código definidos por el usuario para realizar tareas específicas, creando soluciones adaptadas a las necesidades del programa. Se definen con la palabra clave **def**, seguida del nombre de la función y paréntesis. Los **parámetros** son variables que reciben datos externos, permitiendo que la función trabaje con información variable en cada ejecución. Una función puede devolver un valor usando **return**, lo que permite obtener y reutilizar el resultado en otras partes del código. Las funciones personalizadas mejoran la modularidad, evitan la repetición y facilitan la legibilidad y el mantenimiento del código.

```python
def sumar(a, b):
    return a + b

resultado = sumar(3, 4)  # Resultado: 7
```

Ilustración 3 Función personalizada sumar()

UTILIZACIÓN DE UNA FUNCIÓN PERSONALIZADA

Las funciones se invocan escribiendo su nombre seguido de paréntesis con los argumentos necesarios.

```python
def multiplicar(x, y):
    return x * y

producto = multiplicar(5, 6)  # Resultado: 30
print("El producto es:", producto)
```

Ilustración 4 Función multiplicar

¿QUÉ ES UN MÓDULO Y PARA QUÉ SIRVE?

Un módulo es un archivo que almacena código Python, incluyendo funciones, variables y clases, que puede ser reutilizado en distintos programas. Los módulos permiten organizar el código en secciones más pequeñas y específicas, facilitando su mantenimiento y legibilidad. Además, Python cuenta con una gran cantidad de módulos en su biblioteca estándar, como **math** y **statistics**, que ofrecen herramientas para realizar operaciones matemáticas y estadísticas avanzadas. Importar módulos permite a los desarrolladores usar funciones ya creadas, ahorrando tiempo y evitando duplicación de código.

LIBRERÍA ESTÁNDAR DE PYTHON: MÓDULOS MATH Y STAT

Python cuenta con una librería estándar de módulos predefinidos.

Módulo math:

El módulo **math** en Python ofrece funciones matemáticas avanzadas para cálculos complejos. Incluye operaciones como raíces cuadradas (**sqrt**), funciones trigonométricas (**sin**, **cos**), logaritmos (**log**), constantes como pi (**pi**) y e (**e**). Es útil para proyectos que requieren precisión matemática y cálculos científicos.

```python
import math
raiz = math.sqrt(25)  # Resultado: 5.0
print("Raíz cuadrada de 25 es:", raiz)
```

Ilustración 5 Uso de la función sqrt() del módulo math

Módulo statistics:

El módulo **statistics** en Python proporciona funciones para cálculos estadísticos comunes, como media (**mean**), mediana (**median**), moda (**mode**) y desviación estándar (**stdev**). Facilita el análisis de datos sin necesidad de escribir algoritmos complejos, siendo ideal para proyectos de ciencia de datos y análisis estadístico.

```python
import statistics
datos = [1, 2, 3, 4, 5]
media = statistics.mean(datos)
print("La media es:", media)  # Resultado: 3
```

Ilustración 6 Uso de la función mean del módulo statistics

IMPORTACIÓN DE MÓDULOS

En Python, importar módulos permite utilizar código almacenado en archivos externos. Para importar un módulo completo, usa **import nombre_modulo**. Esto agrega todas las funciones y variables del módulo, que luego se pueden acceder con **nombre_modulo.función**.

Para mayor precisión, puedes importar solo funciones específicas de un módulo con **from nombre_modulo import función**. Esto permite usar la función directamente, sin el prefijo del módulo.

Además, puedes renombrar módulos usando as para asignarle un alias, útil si el nombre es largo o si deseas claridad en el código. Por ejemplo, **import statistics as stats** permite usar **stats.mean()** en lugar de **statistics.mean()**. Estas opciones optimizan la gestión de módulos y facilitan la escritura de código limpio y claro.

```python
import math
print(math.pi)  # Imprime el valor de pi

from math import sqrt
print(sqrt(16))  # Resultado: 4.0

import statistics as stats
print(stats.median([1, 2, 3, 4, 5]))  # Resultado: 3
```

Ilustración 7 Importación de Módulo completo, de una función específica y alias de un módulo

EJERCICIOS PRÁCTICO: IMPORTACIÓN DE MÓDULOS EN PYTHON

Paso 1: Importa un Módulo Completo

1. Abre un archivo de Python o el editor de tu preferencia.
2. Escribe import math para importar el módulo math.
3. Prueba una función del módulo, por ejemplo, math.sqrt(16), que calcula la raíz cuadrada de 16.
4. Ejecuta el código y observa el resultado.

```python
import math

# Usar la función sqrt para calcular la raíz cuadrada de 16
resultado = math.sqrt(16)
print("La raíz cuadrada de 16 es:", resultado)
```

Ilustración 8 Paso 1

```
La raíz cuadrada de 16 es: 4.0
```

Ilustración 9 Resultado

Paso 2: Importa una Función Específica de un Módulo

1. Ahora, importaremos solo la función mean del módulo statistics.
2. Escribe from statistics import mean.
3. Usa mean([5, 10, 15]) para calcular el promedio de estos números.
4. Ejecuta el código y observa el resultado.

```python
from statistics import mean

# Calcular el promedio de una lista de números
lista_numeros = [5, 10, 15]
promedio = mean(lista_numeros)
print("El promedio de la lista es:", promedio)
```

Ilustración 10 Paso 2

```
El promedio de la lista es: 10
```

Ilustración 11 Resultado

Paso 3: Renombra un Módulo con un Alias

1. Para hacer el código más legible, asignaremos un alias al módulo math.
2. Escribe import math as m.
3. Usa m.pow(2, 3) para calcular 2 elevado a la potencia de 3.
4. Ejecuta el código y observa el resultado.

```python
import math as m

# Usar la función pow para calcular 2 elevado a la potencia de 3
potencia = m.pow(2, 3)
print("2 elevado a la potencia de 3 es:", potencia)
```

Ilustración 12 Paso 3

```
2 elevado a la potencia de 3 es: 8.0
```

Ilustración 13 Resultado