Universidad Andrés Bello
Conectar · Innovar · Liderar

Programación para Física y Astronomía

Guía de estudios: Funciones & Clases

1^{ra} Actividad: Crear una función en Python

Escriba un programa para crear una función que tome dos argumentos, nombre y edad, e imprima su valor.

2^{ra} Actividad: Cálculo de Índice de Masa Corporal (IMC)

El Índice de Masa Corporal (IMC) es un método utilizado para estimar la cantidad de grasa corporal que tiene una persona, y determinar por tanto si el peso está dentro del rango normal, o por el contrario, se tiene sobrepeso o desnutrición.

El IMC se calcula de la siguiente manera:

```
IMC = peso(kg) / altura(m)^2
```

En este ejercicio, su tarea es crear una función en Python llamada calcular_imc que tome dos parámetros: peso (en kilogramos) y altura (en metros), y retorne el IMC de una persona.

Una vez que hayas definido la función, prueba su funcionamiento con diferentes valores de peso y altura para asegurarse de que funciona correctamente.

Ejemplo:

```
def calcular_imc(peso, altura):
    # Aquí iría su código
    pass

# Probando la función
print(calcular_imc(70, 1.75))  # Debería imprimir aproximadamente 22.86
```

Recuerde: La fórmula para calcular el IMC divide el peso de la persona por el cuadrado de su altura. Así que necesitarás utilizar operaciones aritméticas en tu función.

3^{ra} Actividad: Conversión de Temperaturas

La temperatura puede ser medida en diferentes escalas, las más comunes son Celsius, Fahrenheit y Kelvin. La relación entre estas escalas se da mediante las siguientes fórmulas de conversión:

- De Celsius a Fahrenheit: F = C * 9/5 + 32

- De Fahrenheit a Celsius: C = (F - 32) * 5/9
- De Celsius a Kelvin: K = C + 273.15
- De Kelvin a Celsius: C = K - 273.15

Tu tarea es definir cuatro funciones en Python para realizar estas conversiones de temperatura. Las funciones deben recibir un valor de temperatura y retornar la temperatura convertida a la escala correspondiente. Las funciones deben llamarse: celsius_a_fahrenheit, fahrenheit_a_celsius, celsius_a_kelvin, y kelvin_a_celsius.

Una vez que hayas definido las funciones, realiza algunas pruebas para verificar que funcionan correctamente.

Ejemplo:

```python
def celsius_a_fahrenheit(c):
    # Aquí iría tu código
    pass

def fahrenheit_a_celsius(f):
    # Aquí iría tu código
    pass

def celsius_a_kelvin(c):
    # Aquí iría tu código
    pass

def kelvin_a_celsius(k):
    # Aquí iría tu código
    pass

# Probando las funciones
print(celsius_a_fahrenheit(0))  # Debería imprimir 32
print(fahrenheit_a_celsius(32))  # Debería imprimir 0
print(celsius_a_kelvin(0))  # Debería imprimir 273.15
print(kelvin_a_celsius(273.15))  # Debería imprimir 0
```

Recuerde: Recuerda que para llevar a cabo estas conversiones, necesitarás realizar operaciones aritméticas en tus funciones.

4^{ra} Actividad: Sistema de Categorías basado en Edades

Vamos a imaginar que estás ayudando a desarrollar el sistema de un cine. Uno de los requerimientos es poder categorizar a los espectadores según su edad. Las categorías son las siguientes:

- “Niño” si la edad es menor de 13 años.
- “Adolescente” si la edad está entre 13 y 17 años.
- “Adulto” si la edad es de 18 años o más.

Tu tarea es crear una función en Python llamada categoria_edad que tome como argumento una edad (entero) y devuelva la categoría correspondiente. Si la edad ingresada es negativa, la función debe retornar ”Error: Edad inválida”.

Después de crear la función, debes probarla con diferentes edades para asegurarte de que funcione correctamente.

Ejemplo:

```
def categoria_edad(edad):
    # Aquí iría tu código
    pass

# Probando la función
print(categoria_edad(10))  # Debería imprimir "Niño"
print(categoria_edad(15))  # Debería imprimir "Adolescente"
print(categoria_edad(25))  # Debería imprimir "Adulto"
print(categoria_edad(-5))  # Debería imprimir "Error: Edad inválida"
```

Recuerde: La lógica de la función debe basarse en una serie de comprobaciones condicionales (if, elif, else).

5^{ra} Actividad: Análisis de ventas

Supón que trabajas para una empresa que tiene varias tiendas alrededor del mundo y quieres analizar las ventas de los diferentes productos.

Tienes una lista de todas las ventas realizadas durante el último mes en todas las tiendas. Cada venta es un diccionario con dos elementos: ”producto” y ”ganancia”. Por ejemplo:

```
venta = {"producto": "manzanas", "ganancia": 150}
```

La lista de todas las ventas se ve así:

```
ventas = [
    {"producto": "manzanas", "ganancia": 150},
    {"producto": "naranjas", "ganancia": 200},
    {"producto": "manzanas", "ganancia": 120},
    # Más ventas...
]
```

Tu tarea es crear dos funciones:

- producto_mas_vendido(ventas): Esta función debe recibir la lista de ventas y retornar el nombre del producto que generó más ganancia en total.
- total_ganancia(ventas, producto): Esta función debe recibir la lista de ventas y el nombre de un producto, y debe retornar la ganancia total generada por ese producto.

Asegúrate de probar tus funciones con diferentes datos para verificar que funcionan correctamente.

Ejemplo:

```python
def producto_mas_vendido(ventas):
    # Aquí iría tu código
    pass

def total_ganancia(ventas, producto):
    # Aquí iría tu código
    pass

# Probando las funciones
ventas = [
    {"producto": "manzanas", "ganancia": 150},
    {"producto": "naranjas", "ganancia": 200},
    {"producto": "manzanas", "ganancia": 120},
    {"producto": "naranjas", "ganancia": 180},
    {"producto": "manzanas", "ganancia": 90},
    {"producto": "peras", "ganancia": 300},
    {"producto": "peras", "ganancia": 250},
]

print(producto_mas_vendido(ventas))  # Debería imprimir "peras"
print(total_ganancia(ventas, "manzanas"))  # Debería imprimir 360
```

Recuerda: Para resolver este problema, tendrás que iterar a través de la lista de ventas y realizar cálculos basados en los datos de cada venta. Considera utilizar un diccionario para llevar un registro de las ganancias totales de cada producto.

6^{ra} Actividad: Cree una Clase con Atributos de Instancia

Escriba un programa en Python para crear una clase de vehículo con atributos de instancia de velocidad_maxima y kilometraje.

7^{ra} Actividad:

Cree una clase secundaria Bus que heredará todas las variables y métodos de la clase Vehículo.

8^{ra} Actividad:

Cree una clase Bus que sea herencia de la clase Vehículo. Asigne al argumento de capacidad Bus.asientos_capacidad() a un valor predeterminado de 50.

```python
class Vehiculo:
    def __init__(self, nombre, velocidad_maxima, kilometraje):
        self.nombre = nombre
        self.velocidad_maxima = velocidad_maxima
        self.kilometraje = kilonetraje

    def asientos_capacidad(self, capacidad):
        return f"La capacidad de asientos de un {self.nombre} es de {capacidad} pasajeros"
```

9^{ra} Actividad:

Defina una propiedad que debe tener el mismo valor para cada instancia de clase (objeto). Defina un atributo de clase "color" con un valor predeterminado blanco. Es decir, cada vehículo debe ser blanco.

Use el siguiente código para este ejercicio:

```
class Vehiculo:
    def __init__(self, nombre, velocidad_maxima, kilometraje):
        self.nombre = nombre
        self.velocidad_maxima = velocidad_maxima
        self.kilometraje = kilometraje

class Bus(Vehiculo):
    pass

class Car(Vehiculo):
    pass
```

10^{ra} Actividad: Herencia de Clase

Cree una clase secundaria Bus que herede de la clase vehículo. El cargo de tarifa predeterminado de cualquier vehículo es capacidad de asientos * 100. Si el vehículo es una instancia de autobús, debemos agregar un 10 % adicional a la tarifa completa como cargo de mantenimiento. Entonces, la tarifa total para la instancia de autobús se convertirá en el monto final = tarifa total + 10% de la tarifa total.

Nota: La capacidad de asientos del autobús es de 50, por lo que el monto de la tarifa final debe ser de 5500. Debe anular el método de tarifa() de una clase de vehículo en la clase de autobús.

Utilice el siguiente código para su clase de vehículo principal. Necesitamos acceder a la clase principal desde dentro de un método de una clase secundaria.

```
class Vehiculo:
    def __init__(self, nombre, kilometraje, capacidad):
        self.nombre = nombre
        self.kilometraje = kilometraje
        self.capacidad = capacidad

    def tarifa(self):
        return self.capacidad * 100

class Bus(Vehiculo):
    pass

Bus_escolar = Bus("Escuela Salamanca", 12, 50)
print("La tarifa total del autobús es:", Bus_escolar.tarifa())
```