CURSO DE PROGRAMACIÓN FULL-STACK

Python / Guía 6

Ejercicios

Funciones

egg

Ejercicios de aprendizaje

Con Visual Studio Code, debes crear un archivo para cada ejercicio, denominando a cada archivo de la siguiente manera: `ejercicio_6_1.py, ejercicio_6_2.py, etc.`

Ninguna función de los siguientes ejercicios debe devolver un print(). En todos los ejercicios, el usuario debe interactuar con el sistema (puedes usar el bucle while True, y una opción para salir del programa. Puedes usar una función principal (comúnmente llamada `'main'`) para interactuar con el usuario y desde allí invocar las demás funciones)

1. Crea una función llamada `área_rectángulo` que devuelva el área del rectángulo a partir de una base y una altura. El área de un rectángulo se obtiene al multiplicar la base por la altura. El usuario debe ingresar dos números, el resultado se guarda en una variable y se imprime el valor de la variable mostrándola al usuario. Documenta la función.

2. Crea una función llamada `área_círculo` que devuelva el área de un círculo a partir de un radio. El área de un círculo se obtiene al elevar el radio a dos y multiplicando el resultado por el número pi. Importa el módulo math para utilizar la variable/constante math.pi. Documenta la función.

3. Crea una función llamada `relación.` Recibirá 2 parámetros de tipo entero. Si el primer parámetro es mayor que el segundo, la función debe devolver 1; pero si el primer número es menor que el segundo, debe devolver -1. Si ambos números son iguales, debe devolver 0. Documenta la función.

4. Crea una función llamada `intermedio.` Recibirá dos argumentos de tipo float. Devolverá el número intermedio. El número intermedio de dos números corresponde a la suma de los dos números dividida entre 2. Documenta la función.

5. Crea una función llamada `recortar.` Recibirá tres parámetros. El primero es el número (a recortar), el segundo es el "límite inferior" y el tercero el "límite superior". La función hará lo siguiente:

Devolver el "límite inferior", si el número (a recortar) es menor que este.
Devolver el "límite superior", si el número es mayor que este.
Devolver el "número" sin cambios, si no se supera ningún límite.

6. Crea una función llamada separar. Recibirá **una lista** de números enteros. Devolverá **dos listas ordenadas.** La primera, con los números pares, y la segunda, con los números impares. Documenta la función.

Ver video "Ejemplo listas"
(verlo después de hacer el ejercicio)

7. Realiza el ejercicio anterior con el método de comprensión de listas.

8. Un usuario debe ingresar un número entero, y puede ingresar los dígitos precedidos (sin espacios) del signo '+' o el signo '-'. Ejemplo: `2000, -2000 o +2000.` Luego de que haya ingresado el número, la función 'validar_número' debe recibir como parámetro esa cadena de texto, analizarla, y devolver True si cumple las condiciones, si no, False. Si la función devuelve True, debes convertir la cadena a un entero. Documentar la función. Imprimir el resultado.

9. Crea una función que reciba 3 parámetros de enteros: horas, minutos y segundos, los convierta en "tiempo" y devuelva una tupla de enteros: días, horas, minutos, segundos. El usuario debe introducir los datos desde la consola y debe ver el resultado de forma legible. Una buena práctica de programación es crear varias funciones que solamente hagan una cosa. Trata de documentar brevemente cada función, y separar la lógica, la interacción con el usuario y lo que se muestra por pantalla:

Ver video "Ejemplo días"

Copia y pega el ejercicio anterior en un nuevo archivo. Crea otra función, `'año_bisiesto'` que reciba como parámetro un entero. La función debe

validar si el número ingresado corresponde a un año bisiesto o no. Devuelve True o False. Documentar la función. Cuando el usuario ingresa el input, la función `'validar_numero'` devuelve True o False. Si es True, llamar a `'año_bisiesto'` y mostrar al usuario si lo ingresado es o no un año bisiesto.

Ver video "Ejemplo año bisiesto"

10. A partir de la sección anterior, Colecciones de datos II, mejora el último ejercicio usando funciones. Solamente realiza la parte de los "artículos", pero con la siguiente estructura:

```
{
    "artículo": {
        {
            "id": int,
            "nombre": str,
            "precio": float,
            "cantidad": int
        },
    }
}
```

El "id" sería el "código del artículo", que no puede repetirse. Puedes crear un menú y agregar funcionalidades. Emplea las mejores prácticas que has aprendido.