




Desafío entregable 5 (Clase 9 y 10)

"Funciones"

1) Realiza una función llamada `area_rectangulo()` que devuelva el área del rectángulo a partir de una base y una altura. Calcula el área de un rectángulo de 15 de base y 10 de altura

 **Ayuda:** El área de un rectángulo se obtiene al multiplicar la base por la altura.


Respuesta

```
# Genero la función solicitada para el rectángulo:
def area_rectangulo (base, altura):
    area_rect = base*altura
    return area_rect

# Defino y asigno las variables que conforman el rectángulo, y la función anterior:
base = 15
altura = 10

# Calculamos el área y lo mostramos:
print(f"Un rectangulo de {base} de base y {altura} de altura tiene
{area_rectangulo(base,altura)} de area.")
```

2) Realiza una función llamada `area_circulo()` que devuelva el área de un círculo a partir de un radio. Calcula el área de un círculo de 5 de radio

 **Ayuda:** El área de un círculo se obtiene al elevar el radio a dos y multiplicando el resultado por el número pi. Puedes utilizar el valor 3.14159 como pi o importarlo del módulo `math`.

Respuesta

```
# Importo la librería para hacer uso de "pi":
import math

# Genero la función solicitada para el círculo:
def area_circulo (radio):
    area_circ = math.pi*(radio**2)
    return area_circ

# Defino y asigno la variable que conforman el círculo, y la función anterior:
radio = 5

# Calculamos el área y lo mostramos, limitado a 2 decimales:
print(f"Un círculo de {radio} de radio tiene {area_circulo(radio):.2f} de area.")
```

3) Realiza una función llamada `relacion()` que a partir de dos números cumpla lo siguiente:

1. Si el primer número es mayor que el segundo, debe devolver 1.
2. Si el primer número es menor que el segundo, debe devolver -1.
3. Si ambos números son iguales, debe devolver un 0.

Comprueba la relación entre los números: '5 y 10', '10 y 5' y '5 y 5'

Respuesta

```
# Definimos la función con las consignas del ejercicio:
def relacion(num_1, num_2):
    if num_1 > num_2:
        valor_a_devolver = 1
    elif num_1 < num_2:
        valor_a_devolver = -1
    elif num_1 == num_2:
        valor_a_devolver = 0


    return valor_a_devolver

# Probamos las 3 relaciones del enunciado:

# Para simpleza, creo una lista de numeros con 3 elementos, los cuales son una sublista (parejas de
# numeros):
listado_numeros = [[5,10],[10,5],[5,5]]

# Aplicamos "for" para recorrer la lista, y mostrar el resultado de la relacion
for pares_numeros in listado_numeros:
    print (f"El 1er y 2do número, '{pares_numeros[0]}' y '{pares_numeros[1]}' devuelve el valor de
    '{relacion(pares_numeros[0],pares_numeros[1])}' de la funcion 'relacion'.")
```

4) Realiza una función llamada `intermedio()` que a partir de dos números, devuelva su punto intermedio:

 **Ayuda:** El número intermedio de dos números corresponde a la suma de los dos números dividida entre 2

Comprueba el punto intermedio entre -12 y 24

Respuesta

Definimos la función del enunciado:

```
def intermedio (num_1, num_2):  
    promedio = (num_1+num_2)/2  
    return promedio
```

Definimos y asignamos las variables a comprobar:

```
numero_1 = -12  
numero_2 = 24
```

Calculamos el valor intermedio (promedio de ambos números). Por defecto lo dejamos como "float", en caso de haber decimales:

```
print(f"Un valor intermedio de {numero_1} y {numero_2} es:  
{intermedio(numero_1,numero_2)}")
```

5) Realizá una función llamada `recortar()` que reciba tres parámetros. El primero es el número a recortar, el segundo es el límite inferior y el tercero el límite superior. La función tendrá que cumplir lo siguiente:

1. Devolver el límite inferior si el número es menor que éste
2. Devolver el límite superior si el número es mayor que éste.
3. Devolver el número sin cambios si no se supera ningún límite.

Comprueba el resultado de recortar 15 entre los límites 0 y 10


Respuesta

```
# Definimos la función recortar:
def recortar (num_recortar, lim_inferior, lim_superior):
    if num_recortar < lim_inferior and num_recortar < lim_superior:
        valor_a_devolver = lim_inferior
    elif num_recortar > lim_inferior and num_recortar > lim_superior:
        valor_a_devolver = lim_superior
    elif num_recortar >= lim_inferior and num_recortar <= lim_superior:
        valor_a_devolver = num_recortar
    return valor_a_devolver

# Definimos las variables a comprobar:
numero_recortar = 15
limite_inferior = 0
limite_superior = 10

# Mostramos el resultado para el caso a comprobar:
print(f"La función recortar, considerando a {numero_recortar} como el número a recortar y {limite_inferior} y {limite_superior}, como límites inferior y superior, da como resultado: {recortar (numero_recortar, limite_inferior, limite_superior)}")
```

6) Realiza una función `separar()` que tome una lista de números enteros y devuelva dos listas ordenadas. La primera con los números pares, y la segunda con los números impares:

 **Ayuda:** Para ordenar una lista automáticamente puedes usar el método `.sort()`

Respuesta

```
# Definimos la función separar:
def separar(lista):

    # Creo lista vacía para alojar luego los valores pares e impares:
    pares=[]
    impares=[]

    # Evalúo cada elemento de la lista y defino si es par o no, para anidarlo a la lista correspondiente:
    for elemento in lista:
        if elemento%2==0:
            pares.append(elemento)
        else:
            impares.append(elemento)

    # Una vez generadas las listas, las ordenamos con "sort()":
    pares.sort()
    impares.sort()

    # Mostramos el resultado final de listas pares e impares, ambas ordenadas :
    print(f"La lista {lista} tiene las lista pares e impares ordenadas siguientes: \nPares:
    {pares} \nImpares: {impares}")

# Comprobamos según la lista a evaluar:
lista_evaluar = [6,5,2,1,7]
separar(lista_evaluar)
```