



Desafío entregable 3 (Clase 5)

"Control de flujo"

1) Escribí un programa que lea dos números por teclado y permita elegir entre 4 opciones en un menú:

1. Mostrar una suma de los dos números
2. Mostrar una resta de los dos números (el primero menos el segundo)
3. Mostrar una multiplicación de los dos números
4. Si elige esta opción se interrumpirá la impresión del menú y el programa finalizará
5. En caso de no introducir una opción válida, el programa informará de que no es correcta.

Respuesta

```
# Ingreso los numeros, con "float" debido a que pueden ser con decimales:
num_1=float(input("Ingrese el primer valor: "))
num_2=float(input("Ingrese el segundo valor: "))

# Ingreso la condicion (suma, resta, multiplicacion, salir) y aplico "upper()" para despreciar "key
sensitive":
condicion_input=str(input("Qué desea efectuar? suma, resta, multiplicacion, o salir?: "))
condicion=condicion_input.upper()

# Almaceno la suma, resta y multiplicaicon en una variable:
suma_num=num_1+num_2
resta_num=num_1-num_2
multiplicacion_num=num_1*num_2

# Inicio el "while" segun la primer condicion ingresada:
# Si la condicion ingresada es "salir", no pasa por el "while":
while condicion!="SALIR":

    # Si la condicion ingresada es "suma", "resta" o "multiplicacion", hace tal operacion:
    if condicion == "SUMA":
        print(f"La suma es: {suma_num}")
    elif condicion == "RESTA":
        print(f"La resta es: {resta_num}")
    elif condicion == "MULTIPLICACION":
        print(f"La multiplicacion es: {multiplicacion_num}")
```

```
# Si la condicion ingresada no es "suma", "resta", "multiplicacion" o "salir", advierte una opcion no valida:
else:
    print(f'"{condicion}" no es una opcion valida')

# Ingreso la nueva condicion luego de haber efectuado la operacion, siempre y cuando la anterior no haya sido "salir":
condicion_input=str(input("Qué desea efectuar? suma, resta, multiplicacion, o salir?: "))
condicion=condicion_input.upper()
```

2) Escribí un programa que lea un número impar por teclado. Si el usuario no introduce un número impar, debe repetirse el proceso hasta que lo introduzca correctamente.

Respuesta

```
# Asigno una condicion inicial de que el numero es "par":
numero_par_impar="par"

# Si fuese "par", entra en el "while". Luego se pide que ingrese el valor impar, y se evalua si es "impar". Si fuese impar, se asigna "impar":
while (numero_par_impar=="par"):
    numero=int(input("Ingrese cualquier numero impar: "))
    if (numero%2!=0):
        numero_par_impar="impar"
        print(f"El numero ingresado ({numero}) es {numero_par_impar}.")
    else:
        print(f"El numero ingresado ({numero}) es {numero_par_impar}.")
```

3) Escribí un programa que sume todos los números enteros impares desde el 0 hasta el 100:

👉 **Ayuda:** Podes utilizar la funciones `sum()` y `range()` para hacerlo más fácil. El tercer parámetro en la función `range(inicio, fin, salto)` indica un salto de números.

Respuesta

```
# Genero una lista vacia y defino los rangos minimos, maximos y los steps.  
lista_de_valores=[]  
valor_min=0  
valor_max=100  
step=2  
  
# Con "range" limito los valores de cada lista, poniendo "+1" en el "end" para contemplar los  
rangos maximos.  
# Ademas, por ser impares, modifiko el "start" del "range" poniendole "+1".  
lista_de_valores = list(range(valor_min+1,valor_max+1,step))  
  
# Muestro la sumatoria:  
print(f"La sumatoria de los valores impares, entre '{valor_min}' y '{valor_max}' es:  
{sum(lista_de_valores)}")
```

4) Escribí un programa que pida al usuario cuantos números quiere introducir. Luego lee todos los números y realiza una media aritmética:

Respuesta

```
# Le pido al usuario que ingrese la cantidad de valores a evaluar:
cantidad_numeros_ingresados=int(input("Introduzca la cantidad de valores:"))

# Creo una lista nula con un tamaño igual a la cantidad de valores a ingresar:
lista_numeros_ingresados=[0]*cantidad_numeros_ingresados


# Voy a ir completando la lista nula, desde el índice "0". Por esto, i=0:
i=0

# Inicio el "while" hasta la cantidad de numeros ingresados:
while i<cantidad_numeros_ingresados:

    # Voy cargando los valores ingresados por el usuario en los indices de la lista nula:
    lista_numeros_ingresados[i]=int(input(f"Introduzca el valor N°{i+1}:"))
    i=i+1

# Muestro los valores ingresados (en formato de lista), la sumatoria, y su media:
print(f"Los valores ingresados son: {lista_numeros_ingresados}")
print(f"La suma de los valores ingresados es: {sum(lista_numeros_ingresados)}")
print(f"La media de los valores ingresados es:
{sum(lista_numeros_ingresados)/cantidad_numeros_ingresados}")
```

5) Escribí un programa que pida al usuario un número entero del 0 al 9, y que mientras el número no sea correcto se repita el proceso. Luego debe comprobar si el número se encuentra en la lista de números y notificarlo:

 **Ayuda:** La sintaxis "valor in lista" permite comprobar fácilmente si un valor se encuentra en una lista (devuelve True o False)

Respuesta

```
# Defino una lista de numero a evaluar (la del enunciado):
numeros = [1, 3, 6, 9]

# Defino una variable. Indica si el numero ingresado es correcto o incorrecto (segun si esta dentro
de 0 y 9). Inicialmente vale:
estado_num_ingresado = "incorrecto"

# Mediante el "while", siempre que el numero ingresado sea "incorrecto", se va a solicitar ingresar
un valor:
while estado_num_ingresado == "incorrecto":
    numero_ingresado = int(input("Ingrese un numero entre 0 y 9 inclusive: "))


    # Ahora bien, si el valor ingresado pertenece al rango deseado, cambia el estado del numero
    ingresado a "correcto", sale del "while" y deja de pedir valores:
    if (numero_ingresado >=0 and numero_ingresado <=9):
        estado_num_ingresado = "correcto"

    # Si el valor fuese "correcto", se evalua si pertenece o no a la lista:
    if numero_ingresado in numeros:
        print(f"El numero ingresado '{numero_ingresado}' es '{estado_num_ingresado}' y SI
pertenece a la lista {numeros}.")
    else:
        print(f"El numero ingresado '{numero_ingresado}' es '{estado_num_ingresado}'
pero NO pertenece a la lista {numeros}.")

# Si el valor ingresado sigue siendo incorrecto, se notifica en pantalla:
else:
    print(f"El numero ingresado '{numero_ingresado}' es '{estado_num_ingresado}'.")
```

6) Utilizando la función `range()` y la conversión a listas genera las siguientes listas dinámicamente:

- Todos los números del 0 al 10 [0, 1, 2, ..., 10]
- Todos los números del -10 al 0 [-10, -9, -8, ..., 0]
- Todos los números pares del 0 al 20 [0, 2, 4, ..., 20]
- Todos los números impares entre -20 y 0 [-19, -17, -15, ..., -1]
- Todos los números múltiplos de 5 del 0 al 50 [0, 5, 10, ..., 50]

 **Ayuda:** la conversión de listas es `mi_lista=list(range(inicio,fin,salto))`

Respuesta

```
# Para todos los casos, primero genero una lista vacia y defino los rangos minimos, maximos y los steps.
# Luego con "range" limito los valores de cada lista, poniendo "+1" en el "end" para contemplar los
rangos maximos.
# Solamente para el anteultimo caso, por ser impares, es que modifico el "start" del "range" poniendole
"+1".

# Todos los números del 0 al 10 [0, 1, 2, ..., 10]
lista_de_valores=[]
valor_min=0
valor_max=10
step=1
lista_de_valores = list(range(valor_min,valor_max+1,step))
print(f"Lista con todos los números del 0 al 10: {lista_de_valores}")

# Todos los números del -10 al 0 [-10, -9, -8, ..., 0]
lista_de_valores=[]
valor_min=-10
valor_max=0
step=1
lista_de_valores = list(range(valor_min,valor_max+1,step))
print(f"Lista con todos los números del -10 al 0: {lista_de_valores}")

# Todos los números pares del 0 al 20 [0, 2, 4, ..., 20]
lista_de_valores=[]
valor_min=0
valor_max=20
step=2
lista_de_valores = list(range(valor_min,valor_max+1,step))
print(f"Lista con números pares, del 0 al 20: {lista_de_valores}")

# Todos los números impares entre -20 y 0 [-19, -17, -15, ..., -1]
lista_de_valores=[]
valor_min=-20
valor_max=0
step=2
lista_de_valores = list(range(valor_min+1,valor_max+1,step))
print(f"Lista con números impares, del -20 al 0: {lista_de_valores}")
```

```
# Todos los números múltiples de 5 del 0 al 50 [0, 5, 10, ..., 50]
lista_de_valores=[]
valor_min=0
valor_max=50
step=5
lista_de_valores = list(range(valor_min,valor_max+1,step))
print(f"Lista con números multiplos de 5, entre 0 y 50: {lista_de_valores}")
```

7) Dadas dos listas, debes generar una tercera con todos los elementos que se repitan en ellas, pero no debe repetirse ningún elemento en la nueva lista:

Respuesta

```
# Se asignan las listas del enunciado:
lista_1 = ["h",'o','l','a',' ', 'm','u','n','d','o']
lista_2 = ["h",'o','l','a',' ', 'l','u','n','a']

# Defino la "lista_mayor" y la "lista_menor" vacias. La primera es la que tiene mas cantidad de
elementos, y la segunda, la que menos tiene.
# Esto lo hago para poder independizarme a futuro de que la "lista_1" sea mayor a "lista_2", en cuanto
a cantidad de elementos:
lista_mayor=[]
lista_menor=[]

# Si la primer lista ingresada es mayor a la segunda, defino que la primer lista sea la "mayor". Sino, la
segunda lista será la mayor:
if len(lista_1)>=len(lista_2):
    lista_mayor=lista_1
    lista_menor=lista_2
else:
    lista_mayor=lista_2
    lista_menor=lista_1

# Creo una lista vacia, donde voy a colocar los elementos que se repiten entre las listas del enunciado:
lista_nueva=[]

# Para cada elemento de la "lista_mayor", si tal elemento es igual al elemento de la "lista_menor", le
indico que complete la "lista_nueva":
for elemento in lista_mayor:
    if elemento == elemento in lista_menor:
        lista_nueva.append(elemento)

#Muestro la lista con elementos en comun:
print(f"La lista con elementos en comun es: {lista_nueva}.")

# Creo una nueva lista vacia, que va a tener los elementos de la lista sin repetirse:
lista_nueva_sin_repetir=[]

# Para cada elemento de "lista_nueva", si no se encuentra dentro de "lista_nueva_sin_repetir", agrega
tal elemento.
# Si algun elemento SI estuviese en "lista_nueva_sin_repetir", evite su agregado:
for elemento in lista_nueva:
    if elemento not in lista_nueva_sin_repetir:
        lista_nueva_sin_repetir.append(elemento)

#Muestro finalmente la lista con elementos en comun, sin repeticiones:
print(f"La lista con elementos en comun, sin repeticiones, es: {lista_nueva_sin_repetir}.")
```