Python

CONSIDERACIONES

Leer el orientador de la clase y haber completado la Guía Nº3

Guía de Ejercicios Nº06

Objetivo: Aplicar todo lo aprendido hasta el momento para resolver las diferentes problemáticas que presenta en Trabajo práctico. Al terminar podrá desarrollar programas complejos que tomen decisiones

Temas: funciones

Problemas Para Resolver

Con estos ejercicios entramos en calor, para entender las ventajas de trabajar con funciones

Funciones

1- Resuelva de nuevo el ejercicio "Bucle for (2) - 5" de manera que el dibujo lo realice una función que reciba dos parámetros (alto y ancho).

2- El enunciado del ejercicio era:
Escriba un programa que pida la anchura y altura de un rectángulo y lo dibuje con caracteres producto (*):

```
Anchura del rectángulo: 5
Altura del rectángulo: 3
* * * * *
* * * * *
* * * * *
```

La solución propuesta de este ejercicio era:

```
anchura = int(input("Anchura del rectángulo: "))
altura = int(input("Altura del rectángulo: "))

for i in range(altura):
    for j in range(anchura):
        print("* ", end="")
    print()
```

3- Escriba un programa que pida la anchura y altura de un rectángulo y el carácter a utilizar en el dibujo:

```
Anchura del rectángulo: 5
Altura del rectángulo: 3
Carácter a utilizar: o
o o o o o
o o o o o
o o o o o
```

Nota: El carácter a utilizar en el dibujo se puede enviar a la función como tercer parámetro.

4- Resuelva de nuevo el ejercicio "Bucle for (2) - 14" de manera que el dibujo lo realicen **dos** funciones que reciban un parámetro (ancho), una que dibuje la primera mitad del triángulo y la otra que dibuje la segunda mitad.
El enunciado del ejercicio era:
Escriba un programa que pida la anchura de un triángulo y lo dibuje con caracteres producto (*):

```
Anchura del triángulo: 4
*
* *
* * *
* * * *
* * *
* *
*
```

5- La solución propuesta de este ejercicio era:

```
anchura = int(input("Anchura del triángulo: "))

for i in range(1, anchura + 1):
    for j in range(i):
        print("* ", end="")
    print()

for i in range(1, anchura):
    for j in range(anchura - i):
        print("* ", end="")
    print()
```

6- Escriba un programa que pida la anchura de un triángulo y dibuje una sucesión de triángulos con caracteres producto (*):

```
Anchura del triángulo: 3
*
* *
* * *
* *
*
*
* *
*
*
```

7- Resuelva de nuevo el ejercicio "if ... elif ... else ... 6" de manera que el cálculo lo realice una función que reciba un parámetro (el año) y devuelva True o False según que el año sea o no bisiesto.

El enunciado del ejercicio era:

Escriba un programa que pida un año y que escriba si es bisiesto o no.
Se recuerda que los años bisiestos son múltiplos de 4, pero los múltiplos de 100 no lo son, aunque los múltiplos de 400 sí.

Estos son algunos ejemplos de posibles respuestas: 2012 es bisiesto, 2010 no es bisiesto, 2000 es bisiesto, 1900 no es bisiesto.

```
COMPROBADOR DE AÑOS BISIESTOS
Escriba un año y le diré si es bisiesto: 2000
El año 2000 es un año bisiesto.
```

Nota: El enunciado original pedía que el programa escribiera la razón por el que el año era o no bisiesto, pero aquí se ha simplificado el ejercicio omitiendo esa explicación.

La solución propuesta de este ejercicio era:

```
print("COMPROBADOR DE AÑOS BISIESTOS")
fecha = int(input("Escriba un año y le diré si es bisiesto: "))

if fecha % 400 == 0 or (fecha % 100 != 0 and fecha % 4 == 0):
    print("El año", fecha, "es un año bisiesto.")
else:
    print("El año", fecha, "no es un año bisiesto.")
```

8- Escriba un programa que pida dos años y escriba cuántos años bisiestos hay entre esas dos fechas (incluidos los dos años):

```
CONTADOR DE AÑOS BISIESTOS
Escriba un año: 2000
Escriba otro año posterior a 2000: 1999
1999 no es mayor que 2000. Inténtelo de nuevo: 2013
De 2000 a 2013 hay 14 años, 4 de ellos bisiestos.
```

9- Resuelva de nuevo el ejercicio "Listas (1) - 1" de manera que la lista sea solicitada por una función (que no necesita parámetros).

El enunciado del ejercicio era:

Escriba un programa que permita crear una lista de palabras (que puede ser vacía). Para ello, el programa tiene que pedir un número y luego solicitar ese número de palabras para crear la lista. Por último, el programa tiene que escribir la lista.

```
Dígame cuántas palabras tiene la lista: 3
Dígame la palabra 1: Alberto
Dígame la palabra 2: Benito
Dígame la palabra 3: Carmen
```

```
La lista creada es: ['Alberto', 'Benito', 'Carmen']
```

10- La solución propuesta de este ejercicio era:

```
numero = int(input("Dígame cuántas palabras tiene la lista: "))

if numero < 1:
    print("¡Imposible!")
else:
    lista = []
    for i in range(numero):
        print("Dígame la palabra", str(i + 1) + ": ", end="")
        palabra = input()
        lista += [palabra]
    print("La lista creada es:", lista)
```

11- **Nota**: En el enunciado original, la lista no podía ser vacía, pero en este ejercicio se permite que la lista tenga 0 elementos. En caso de que el usuario escriba un número negativo, el programa puede avisar al usuario o simplemente crear una lista vacía.

Escriba un programa que pida cuántas listas se quieren crear y las solicite a continuación:

```
Generador de listas
¿Cuántas listas quiere escribir? 2

Dígame cuántas palabras tiene la lista 1: 3
Dígame la palabra 1: Ana
Dígame la palabra 2:  Bárbara
Dígame la palabra 3:  Carmen
La lista 1 es: ['Ana', 'Bárbara', 'Carmen']

Dígame cuántas palabras tiene la lista 2:  2
Dígame la palabra 1: Alberto
Dígame la palabra 2: Bernardo
La lista 2 es: ['Alberto', 'Bernardo']
```

Nota**: Se puede añadir un parámetro a la función que crea la lista, el número de lista.**

Modifique el programa anterior de manera que las listas se escriban al final del programa:

```
Generador de listas
¿Cuántas listas quiere escribir? 2

Dígame cuántas palabras tiene la lista 1: 3
Dígame la palabra 1: Ana
Dígame la palabra 2:  Bárbara
Dígame la palabra 3:  Carmen

Dígame cuántas palabras tiene la lista 2:  2
Dígame la palabra 1: Alberto
Dígame la palabra 2: Bernardo

La lista 1 es: ['Ana', 'Bárbara', 'Carmen']
```

```
La lista 2 es: ['Alberto', 'Bernardo']
```

Nota: Se puede crear una lista de listas en la que se vayan introduciendo las listas a medida que se generan

Resuelva de nuevo el ejercicio "Listas (1) - 5" de manera que las listas sean solicitadas por una función (que no necesita parámetros) y que las palabras sean eliminadas por una función (con dos parámetros, las dos listas).

El enunciado del ejercicio era:

12- Escriba un programa que permita crear dos listas de palabras y que, a continuación, elimine de la primera lista los nombres de la segunda lista.

```
Dígame cuántas palabras tiene la lista: 5
Dígame la palabra 1: Carmen
Dígame la palabra 2: Carmen
Dígame la palabra 3: Alberto
Dígame la palabra 4: Benito
Dígame la palabra 5: David
La lista creada es: ['Carmen', 'Carmen', 'Alberto', 'Benito', 'David']

Dígame cuántas palabras tiene la lista de palabras a eliminar: 3
Dígame la palabra 1: Benito
Dígame la palabra 2: Juan
Dígame la palabra 3: Carmen

La lista de palabras a eliminar es: ['Benito', 'Juan', 'Carmen']
La lista es ahora: ['Alberto', 'David']
```

La solución propuesta de este ejercicio era:

```
numero = int(input("Dígame cuántas palabras tiene la lista: "))

if numero < 1:
    print("¡Imposible!")
else:
    lista = []
    for i in range(numero):
        print("Dígame la palabra", str(i + 1) + ": ", end="")
        palabra = input()
        lista += [palabra]
    print("La lista creada es:", lista)

    numero2 = int(input("Dígame cuántas palabras tiene la lista de palabras a
eliminar: "))

    if numero2 < 1:
        print("¡Imposible!")
    else:
        eliminar = []
        for i in range(numero2):
```

```
            print("Dígame la palabra", str(i + 1) + ": ", end="")
            palabra = input()
            eliminar += [palabra]
        print("La lista de palabras a eliminar es:", eliminar)

        for i in eliminar:
            for j in range(len(lista)-1, -1, -1):
                if lista[j] == i:
                    del(lista[j])
        print("La lista es ahora:", lista)
```

Nota: En el enunciado original, la lista no podía ser vacía, pero en este ejercicio se permite que la lista tenga 0 elementos. En caso de que el usuario escriba un número negativo, el programa puede avisar al usuario o simplemente crear una lista vacía.

Modifique el programa anterior de manera que el programa solicite dos listas sin repeticiones y escriba el resultado al final:

```
Dígame cuántas palabras tiene la lista 1: 5
Dígame la palabra 1: Carmen
Dígame la palabra 2: Carmen
Carmen ya está en la lista. Dígame la palabra 2: Pepe
Dígame la palabra 3: Alberto
Dígame la palabra 4: Benito
Dígame la palabra 5: David

Dígame cuántas palabras tiene la lista 2: 3
Dígame la palabra 1: Benito
Dígame la palabra 2: Juan
Dígame la palabra 3: Carmen

Las listas creadas inicialmente son:
['Carmen', 'Pepe', 'Alberto', 'Benito', 'David']
['Benito', 'Juan', 'Carmen']
Las listas tras borrar en la primera lista los elementos que aparecen en la segunda
lista son:
['Pepe', 'David']
['Benito', 'Juan', 'Carmen']
```

Nota: Se puede añadir un parámetro a la función que crea la lista, el número de lista.

Modifique el programa anterior de manera que se escriban varias listas y el programa elimine de las listas anteriores las palabras que aparecen en las siguientes:

```
Generador de listas
¿Cuántas listas quiere escribir? 3
Dígame cuántas palabras tiene la lista 1: 3
Dígame la palabra 1: Ana
Dígame la palabra 2:  Bárbara
Dígame la palabra 3:  Carmen

Dígame cuántas palabras tiene la lista 2:  3
Dígame la palabra 1: Alberto
```

```
Dígame la palabra 2: Bernardo
Dígame la palabra 3: Bárbara
Dígame cuántas palabras tiene la lista 3: 2
Dígame la palabra 1: Carmen
Dígame la palabra 2: Alberto

Las listas creadas inicialmente son:
['Ana', 'Bárbara', 'Carmen]
['Alberto', 'Bernardo', 'Bárbara']
['Carmen', 'Alberto']
Las listas tras borrar en cada lista las palabras de listas posteriores:
['Ana']
['Bernardo', 'Bárbara']
['Carmen', 'Alberto']
```