Trabajo práctico nro. 5

UTN FACULTAD REGIONAL MENDOZA	**Asignatura: Programación I**	
	Cursado: Primer Trimestre	**Horas semanales**: 4
	Carrera: *Tecnicatura Universitaria en Programación*	**Nivel (Año):** ☐ 1° ☐ 2° ☐ 3°
	Ciclo Lectivo: 2023	

Integrantes de la Cátedra:

- **DOCENTES:**

Nombre del Profesor	Periodo	Cantidad horas materia
		6 horas

1. Escribir una función que, dado un número de DNI, retorne True si el número es válido y False si no lo es. Para que un número de DNI sea válido debe tener entre 7 y 8 dígitos.

```python
def is_valid_dni(number_dni):
    # Convertir el número de DNI a una cadena para contar los dígitos
    number_dni_str = str(number_dni)

    # Verificar si el número de DNI tiene entre 7 y 8 dígitos
    if 7 <= len(number_dni_str) <= 8:
        return True
    else:
        return False

# Ejemplo de uso:
dni = int(input("Ingrese el dni que desea verificar: "))  # Ingresa el número de DNI que desea verificar
result = is_valid_dni(dni)
if result:
    print("El número de DNI es válido.")
else:
    print("El número de DNI no es válido.")
```

2. Escribir una función que, dado un string, retorne la longitud de la última palabra. Se considera que las palabras están separadas por uno o más espacios. También podría haber espacios al principio o al final del string pasado por parámetro.

```python
def lenght_last_word(string):
    # Eliminar espacios en blanco al principio y al final de la cadena
    string = string.strip()

    # Dividir la cadena en palabras usando espacios en blanco como separador
    words = string.split()

    # Verificar si hay al menos una palabra en la lista de palabras
    if words:
        # Obtener la última palabra y calcular su longitud
        last_word = words[-1]
        return len(last_word)
    else:
        # Si no hay palabras en la cadena, la longitud es 0
        return 0

# Ejemplo de uso:
string = "   Esta es una cadena de ejemplo   "
lenght = lenght_last_word(string)
print(f"La longitud de la última palabra es {lenght}")
```

3. Escribir un programa que permita al usuario obtener un identificador para cada uno de los socios de un club. Para eso ingresará nombre completo y número de DNI de cada socio, indicando que finalizará el procesamiento mediante el ingreso de un nombre vacio.

   Precondición: el formato del nombre de los socios será: *nombre apellido*. Podría ingresarse más de un nombre, en cuyo caso será: *nombre1, nombre2, apellido*. Si un socio tuviera más de un apellido, el usuario solo ingresará uno.

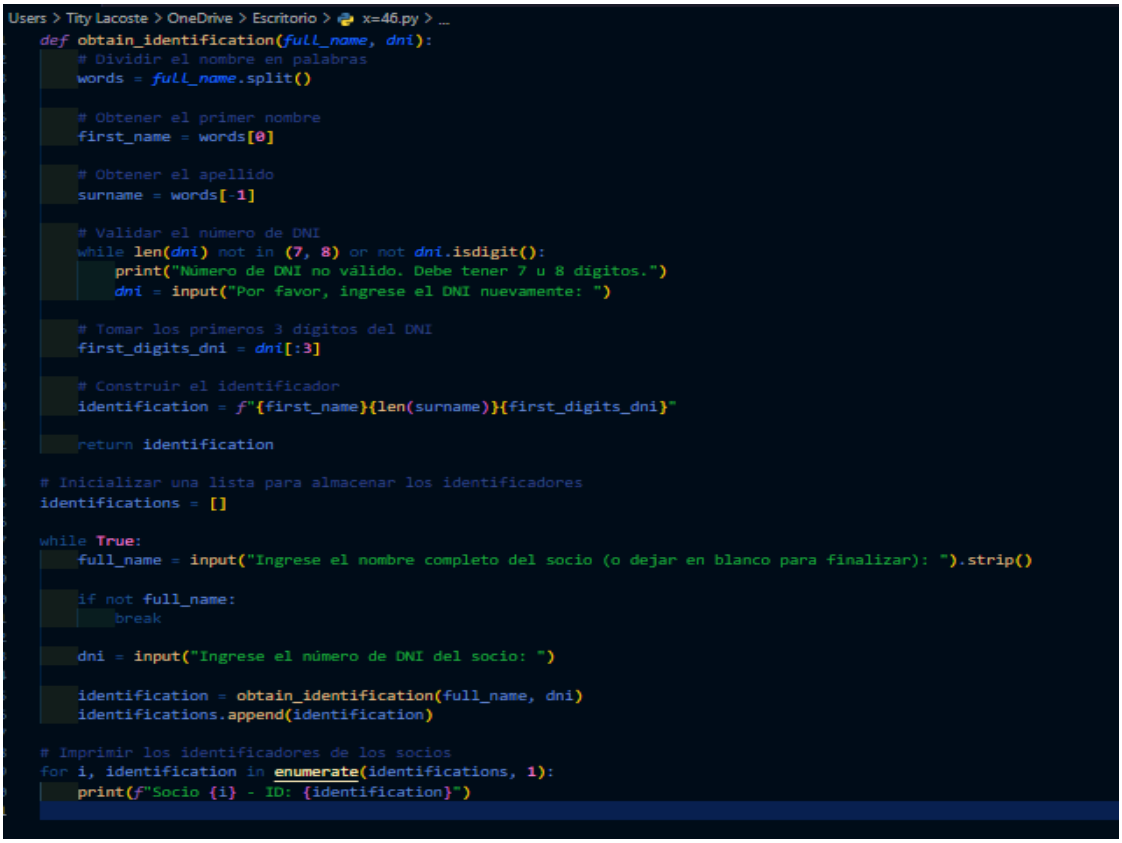
   Se debe validar que el número de DNI tenga 7 u 8 dígitos. En caso contrario, el programa debe dejar al usuario en un bucle hasta que ingrese un DNI correcto.

   Por cada socio se debe imprimir su identificador único, el cual estará formado por: el primer nombre, la cantidad de letras del apellido y los 3 primeros dígitos de su DNI. Ejemplo:

   Nombre: María Ines Rosales

   DNI: 25469648

   ID -> Maria7254



```
Users > Tity Lacoste > OneDrive > Escritorio > x=46.py > ...
def obtain_identification(full_name, dni):
    # Dividir el nombre en palabras
    words = full_name.split()

    # Obtener el primer nombre
    first_name = words[0]

    # Obtener el apellido
    surname = words[-1]

    # Validar el número de DNI
    while len(dni) not in (7, 8) or not dni.isdigit():
        print("Número de DNI no válido. Debe tener 7 u 8 dígitos.")
        dni = input("Por favor, ingrese el DNI nuevamente: ")

    # Tomar los primeros 3 dígitos del DNI
    first_digits_dni = dni[:3]

    # Construir el identificador
    identification = f"{first_name}{len(surname)}{first_digits_dni}"

    return identification

# Inicializar una lista para almacenar los identificadores
identifications = []

while True:
    full_name = input("Ingrese el nombre completo del socio (o dejar en blanco para finalizar): ").strip()

    if not full_name:
        break

    dni = input("Ingrese el número de DNI del socio: ")

    identification = obtain_identification(full_name, dni)
    identifications.append(identification)

# Imprimir los identificadores de los socios
for i, identification in enumerate(identifications, 1):
    print(f"Socio {i} - ID: {identification}")
```

4. Crea un programa que pida dos número enteros al usuario y diga si alguno de ellos es múltiplo del otro. Crea una función que reciba los dos números, y devuelve si el primero es múltiplo del segundo.

```python
# Definir una función que verifica si el primer número es múltiplo del segundo
def is_multiple(number1, number2):
    return number1 % number2 == 0

# Solicitar al usuario dos números enteros
num1 = int(input("Ingresa el primer número entero: "))
num2 = int(input("Ingresa el segundo número entero: "))

# Verificar si alguno de los números es múltiplo del otro utilizando la función
if is_multiple(num1, num2) or is_multiple(num2, num1):
    print("Al menos uno de los números es múltiplo del otro.")
else:
    print("Ninguno de los números es múltiplo del otro.")

```

5. Crear una función que calcule la temperatura media de un día a partir de la temperatura máxima y mínima. Crear un programa principal, que utilizando la función anterior, vaya pidiendo la temperatura máxima y mínima de cada día y vaya mostrando la media. El programa pedirá el número de días que se van a introducir.

```python
# Definir la función para calcular la temperatura media
def temperature_media(temperature_max, temperature_min):
    return (temperature_max + temperature_min) / 2

# Solicitar al usuario el número de días a introducir
number_days = int(input("Ingresa el número de días para los que deseas calcular la temperatura media: "))

# Inicializar una lista para almacenar las temperaturas medias
temperatures_medias = []

# Pedir la temperatura máxima y mínima para cada día y calcular la temperatura media
for day in range(1, number_days + 1):
    temperature_maxima = float(input(f"Día {day}: Ingresa la temperatura máxima: "))
    temperature_minima = float(input(f"Día {day}: Ingresa la temperatura mínima: "))
    media = temperature_media(temperature_maxima, temperature_minima)
    temperatures_medias.append(media)

# Mostrar las temperaturas medias calculadas
print("Temperaturas medias de los días:")
for day, temperature_media in enumerate(temperatures_medias, 1):
    print(f"Día {day}: {temperature_media}°C")
```

6. Crea una función que reciba como parámetro un texto y devuelve una cadena con un espacio adicional tras cada letra. Por ejemplo, “Hola, tú” devolverá “H o l a , t ú “. Crea un programa principal donde se use dicha función.

```python
# Definir la función para agregar un espacio adicional tras cada letra
def add_spaces(text):
    # Usar join para agregar un espacio después de cada letra y luego unir todo
    text_with_spaces = ' '.join(text)
    return text_with_spaces

# Programa principal
original_text = input("Ingresa un texto: ")

text_spaces = add_spaces(original_text)

print("Texto con espacios adicionales:")
print(text_spaces)
```

7. Crea una función que recibe una lista con valores numéricos y devuelve el valor máximo y el mínimo. Crea un programa que pida números por teclado y muestre el máximo y el mínimo, utilizando la función anterior.

```python
# Definir la función para encontrar el valor máximo y mínimo
def find_max_min(values):
    if not values:
        return None, None  # Si la lista está vacía, retornamos None para máximo y mínimo
    maximo = minimo = values[0]  # Inicializamos máximo y mínimo con el primer valor de la lista
    for value in values:
        if value > maximo:
            maximo = value
        elif value < minimo:
            minimo = value
    return maximo, minimo

# Programa principal
values = []

while True:
    number = input("Ingresa un número (o deja en blanco para finalizar): ")
    if not number:
        break  # Salir del bucle si se deja en blanco
    if number.replace(".", "", 1).isdigit():
        num = float(number)
        values.append(number)
    else:
        print("Entrada no válida. Por favor, ingresa un número válido.")
if values:
    maximo, minimo = find_max_min(values)
    print(f"El valor máximo es: {maximo}")
    print(f"El valor mínimo es: {minimo}")
else:
    print("No se ingresaron valores numéricos.")
```

8. Diseñar una función que calcule el área y el perímetro de una circunferencia. Utiliza dicha función en un programa principal que lea el radio de una circunferencia y muestre su área y perímetro.

```python
import math  # Importar el módulo math para acceder a la constante pi

# Definir la función para calcular el área y el perímetro de una circunferencia
def calcule_area_perimeter_circle(radio):
    # Calcular el área de la circunferencia
    area = math.pi * (radio ** 2)

    # Calcular el perímetro de la circunferencia
    perimeter = 2 * math.pi * radio

    return area, perimeter

# Programa principal
radio = float(input("Ingresa el radio de la circunferencia: "))

area, perimeter = calculate_area_perimeter_circle(radio)

print(f"El área de la circunferencia es: {area:.2f}")
print(f"El perímetro de la circunferencia es: {perimeter:.2f}")
```

9. Crear una subrutina llamada “login”, que recibe un nombre de usuario y una contraseña y te devuelve Verdadero si el nombre de usuario es “usuario1” y la contraseña es “asdasd”. Además recibe el número de intentos que se ha intentado hacer login y si no se ha podido hacer login incremente este valor.
   Crear un programa principal donde se pida un nombre de usuario y una contraseña y se intente hacer login, solamente tenemos tres oportunidades para intentarlo.

```python
# Definir la subrutina para el login
def login(user_name, password, tries):
    valid_user_name = "usuario1"
    valid_password = "asdasd"

    if user_name == valid_user_name and password == valid_password:
        return True, tries
    else:
        tries += 1
        return False, tries

# Programa principal
tries = 0
max_tries = 3

while tries < max_tries:
    name_user = input("Nombre de usuario: ")
    passw = input("Contraseña: ")

    valid_access, tries = login(name_user, passw, tries)

    if valid_access:
        print("Acceso concedido. Bienvenido, usuario1.")
        break
    else:
        print("Acceso denegado. Intento número:", tries)

if tries >= max_tries:
    print("Has alcanzado el máximo de intentos permitidos. Acceso bloqueado.")
```

10. Escribir una función que aplique un descuento a un precio. Esta función tiene que recibir un diccionario con los precios y porcentajes del carrito de compra, aplicar los descuentos a los productos del carrito y devolver el precio final de la compra.

```python
def calculate_final_price(shopping_cart):
    final_price = 0
    for product, price in shopping_cart.items():
        if product in discounts:
            discount_price = price * (1 - discounts[product] / 100)
            final_price += discount_price
        else:
            final_price += price
    return final_price
# Ejemplo de un carrito de compra con precios y porcentajes de descuento
shopping_cart = {
    "producto1": 100,
    "producto2": 50,
    "producto3": 75,
}
# Descuentos por producto (en porcentaje)
discounts = {
    "producto1": 10,
    "producto3": 20,
}
total_price = calculate_final_price(shopping_cart)
print(f"Precio total de la compra con descuentos: {total_price:.2f}")
```

11. Escribir una función que reciba otra función y una lista, y devuelva otra lista con el resultado de aplicar la función dada a cada uno de los elementos de la lista.

```python
def apply_function_to_list(function, a_list):
    # Usamos una comprensión de lista para aplicar la función a cada elemento de la lista
    results = [function(element) for element in a_list]
    return results
# Ejemplo de una función que eleva un número al cuadrado
def square(number):
    return number ** 2
# Lista de números
numbers = [1, 2, 3, 4, 5]
# Aplicar la función cuadrado a la lista de números
results = apply_function_to_list(square, numbers)

print("Números originales:", numbers)
print("Números al cuadrado:", results)
```

12. Escribir una función que reciba una frase y devuelva un diccionario con las palabras que contiene y su longitud.

```python
def len_of_words(sentence):
    # Dividir la frase en palabras
    words = sentence.split()
    # Crear un diccionario para almacenar las palabras y su longitud
    result = {}
    # Iterar a través de las palabras y calcular su longitud
    for word in words:
        result[word] = len(word)
    return result

# Ejemplo de una frase
sentence = "Escribir una función en Python"

# Llamar a la función para obtener el diccionario de palabras y sus longitudes
dict_of_len = len_of_words(sentence)

# Mostrar el resultado
for word, lenght in dict_of_len.items():
    print(f"'{word}': {lenght} caracteres")
```

13. Escribir una función que calcule el módulo de un vector.

```
Users > Tity Lacoste > OneDrive > Escritorio > x=46.py > ...
import math
def module_vector(vector):
    squares_sume = sum(componente ** 2 for componente in vector)
    a_module = math.sqrt(squares_sume)
    return a_module

# Ejemplo de uso
vector = [3, 4]  # Vector (3, 4)

a_module = module_vector(vector)
print("El módulo del vector es:", a_module)
```

14. Requerir al usuario que ingrese un número entero e informar si es primo o no, utilizando una función booleana que lo decida.

```python
def is_primo(number):
    if number <= 1:
        return False  # Los números menores o iguales a 1 no son primos
    if number <= 3:
        return True  # 2 y 3 son primos
    if number % 2 == 0 or number % 3 == 0:
        return False  # Los múltiplos de 2 y 3 no son primos
    # Comprobamos si es divisible por otros números
    i = 5
    while i * i <= number:
        if number % i == 0 or number % (i + 2) == 0:
            return False
        i += 6
    return True  # Si no se encontraron divisores, es primo

# Programa principal
numb = int(input("Ingresa un número entero: "))

if is_primo(numb):
    print(f"{numb} es un número primo.")
else:
    print(f"{numb} no es un número primo.")
```

15. Escribir un programa que pida números al usuario, motrar el factorial de cada uno y, al finalizar, la cantidad total de números leídos en total. Utilizar una o más funciones, según sea necesario.

```python
# Definir una función para calcular el factorial de un número
def calculate_factorial(number):
    factorial = 1
    for i in range(1, number + 1):
        factorial *= i
    return factorial
# Inicializar una variable para contar la cantidad de números leídos
quantity_numbers = 0
while True:
    number_str = input("Ingresa un número (o deja en blanco para finalizar): ")
    if number_str == "":
        break  # Salir del bucle si se deja en blanco
    if number_str.isdigit():
        numb = int(number_str)
        quantity_numbers += 1
        factorial = calculate_factorial(numb)
        print(f"El factorial de {numb} es {factorial}")
    else:
        print("Entrada no válida. Por favor, ingresa un número válido.")

print(f"Se han leído un total de {quantity_numbers} números.")
```

16. Solicitar al usuario un número entero y luego un dígito. Informar la cantidad de ocurrencias del dígito en el número, utilizando para ello una función que calcule la frecuencia.

```
Users > Tity Lacoste > OneDrive > Escritorio > x=46.py > ...
def calculate_frequency(number, digit):
    # Convertir el dígito a una cadena para comparar caracteres
    digit_str = str(digit)

    # Convertir el número a una cadena para buscar el dígito
    number_str = str(number)

    # Usar el método count() para contar las ocurrencias del dígito en el número
    frequency = number_str.count(digit_str)

    return frequency

# Programa principal
numb = int(input("Ingresa un número entero: "))
digit = int(input("Ingresa un dígito para contar su frecuencia: "))

frequency = calculate_frequency(numb, digit)

print(f"El dígito {digit} aparece {frequency} veces en el número {numb}.")
```

17. Solicitar al usuario el ingreso de números primos. La lectura finalizará cuando ingrese un número que no sea primo. Por cada número, mostrar la suma de sus dígitos. También solicitar al usuario un dígito e informar la cantidad de veces que aparece en el número (frecuencia). Al finalizar el programa, mostrar el factorial del mayor número ingresado.

```python
def calculate_factorial(numb):
    factorial = 1
    for i in range(1, numb + 1):
        factorial *= i
    return factorial
# Función para verificar si un número es primo
def is_primo(numb):
    if numb <= 1:
        return False
    if numb <= 3:
        return True
    if numb % 2 == 0 or numb % 3 == 0:
        return False
    i = 5
    while i * i <= numb:
        if numb % i == 0 or numb % (i + 2) == 0:
            return False
        i += 6
    return True
# Inicializar variables
major_primo = 0
factorial_major_primo = 1
while True:
    number = int(input("Ingresa un número primo (o un número no primo para finalizar): "))
    if not is_primo(number):
        break
    # Calcular la suma de los digitos del número
    sum_digits = sum(int(digit) for digit in str(number))
    print(f"La suma de los digitos de {number} es: {sum_digits}")
    digit = int(input("Ingresa un digito para contar su frecuencia en el número: "))
    frequency = str(number).count(str(digit))

    print(f"El digito {digit} aparece {frequency} veces en el número {number}")

    if number > major_primo:
        major_primo = number

# Calcular el factorial del mayor número ingresado
if major_primo > 0:
    factorial_major_primo = calculate_factorial(major_primo)

print(f"El factorial del mayor número primo ingresado ({major_primo}) es: {factorial_major_primo}")
```