**Ejercicio Nro. 1:**

**Solicitar al usuario que ingrese su dirección email. Imprimir un  
mensaje indicando si la dirección es válida o no, valiéndose de una  
función para decidirlo. Una dirección se considerará válida si contiene el símbolo "@".**

En primer lugar, se solicita al usuario que ingrese su dirección de correo electrónico utilizando la función `input()`. El valor ingresado se almacena en la variable `direccionMail`. En segundo lugar, se define la función `esMail(direccion)`, que toma un parámetro `direccion`, que representa la dirección de correo electrónico a validar. Dentro de la función esMail, la dirección ingresada se divide en partes utilizando split('@'). Esto devuelve una lista con dos elementos: la parte antes de "@" y la parte después de "@". Luego, se verifica si la lista tiene exactamente dos partes (len(partes) == 2) y si la parte después de "@" contiene al menos un punto "." ('.' in partes[1]). Si ambas condiciones se cumplen, se considera que la dirección de correo electrónico es válida; de lo contrario, se considera inválida. Finalmente, se llama a la función `esMail(direccionMail)` para validar la dirección de correo electrónico ingresada por el usuario.

TESTEO:

* Una dirección de correo electrónico válida: [usuario@gmail.com](mailto:usuario@gmail.com)

RESULTADO OBTENIDO:

[usuario@gmail.com](mailto:usuario@gmail.com)  
Es una direccion de email válida

* Una dirección de correo electrónico sin el símbolo "@": "usuario.gmail.com"

RESULTADO OBTENIDO:

usuario.gmail.com

No es una dirección de email válida

* Una dirección de correo electrónico con múltiples símbolos "@": "usuario@gmail@com"

RESULTADO OBENIDO:

usuario@gmail@com

No es una dirección de email válida

* Una dirección de correo electrónico vacía: ""

RESULTADO OBTENIDO:

No es una dirección de email válida

**Ejercicio Nro. 2:**

**Solicitar números al usuario hasta que ingrese el cero. Por cada  
uno, mostrar la suma de sus dígitos (utilizando una función que  
realice dicha suma).**

Se inicializa una variable bandera como True como variable de control del bucle while que solicita los números al usuario. También se inicializa una lista vacía llamada listaDigitos que se utilizará para almacenar los dígitos de cada número ingresado.

Luego, se define la función sumarDigitos(digitos) que toma una lista de dígitos como argumento y devuelve la suma de esos dígitos. Dentro de la función, se itera sobre cada elemento de la lista y se acumula la suma de ellos en sumaDigitos.

En el siguiente paso, se ejecuta un bucle while mientras bandera sea True. Dentro del bucle, se le solicita al usuario ingresar un número. Si el número ingresado es cero, se cambia el valor de bandera a False para salir del bucle y se imprime un mensaje de finalización. Si el número ingresado no es cero, se continúa con el procesamiento.

Luego, por cada número ingresado, se itera sobre sus dígitos y se agregan a la lista listaDigitos como números enteros.  
for numero in numeroIngresado :

listaDigitos.append(int(numero))

Luego, se llama a la función sumarDigitos(listaDigitos) para sumar los dígitos del número ingresado y se imprime el resultado. Finalmente, se reinicia la lista listaDigitos para el próximo número ingresado.

TESTEO:

* Ingresar número de una sola cifra: 1

RESULTADO OBTENIDO:  
La suma de los digitos es: 1

* Ingresar números de varias cifras: 123

RESULTADO OBTENIDO:

La suma de los digitos es: 6

* Ingresar el número cero para finalizar el programa.

RESULTADO OBTENIDO:

Finalizó el programa

**Ejercicio Nro. 3:**

**Solicitar números al usuario hasta que ingrese el cero. Por cada uno, mostrar la suma de sus dígitos. Al finalizar, mostrar la sumatoria de todos los números ingresados y la suma de sus  
dígitos. Reutilizar la misma función realizada en el ejercicio 2.**

En este programa como primer paso, se inicializan variables como bandera (para controlar el bucle while), listaDigitos (para almacenar los dígitos de cada número) y listaNumeros (para almacenar todos los números ingresados). Luego, se define la función sumarDigitos(digitos) que toma una lista de dígitos como argumento y devuelve la suma de esos dígitos. Esta función es la misma que se utilizó en el Ejercicio Nro. 2.

También, se define la función sumarNumeros(listaNumeros) que toma una lista de números como argumento y devuelve la suma de esos números Y se define la función generarListaDigitos(numeroIngresado) que toma un número como argumento. Dentro de ella se itera sobre cada digito del número ingresado y se agrega cada dígito como un entero a la lista listaDigitos.

Como siguiente paso, se ejecuta un bucle while que solicita al usuario que ingrese un número. Si el número ingresado es cero, se cambia el valor de bandera a False para salir del bucle. Antes de salir, se imprime un mensaje de finalización y se muestra la sumatoria total de todos los números ingresados haciendo una llamada a la función (sumarNumeros(listaNumeros)) y la suma de sus dígitos haciendo una llamada a la función (sumarDigitos(listaDigitos)). Si el número ingresado no es cero, se agrega a la lista listaNumeros y se llama a la función generarListaDigitos para generar la lista de dígitos de ese número. Luego, se imprime la suma de los dígitos del número ingresado mediante una llamada a la función (sumarDigitos(listaDigitos)).

TESTEO:

* Ingresar número de una sola cifra: 1,2,3,0

RESULTADO OBTENIDO:  
ingrese un número o cero '0' para finalizar: 1  
La suma de los digitos es: 1

ingrese un número o cero '0' para finalizar: 2  
La suma de los digitos es: 2

ingrese un número o cero '0' para finalizar: 3  
La suma de los digitos es: 3

ingrese un número o cero '0' para finalizar: 0  
Finalizó el programa

Sumatoria total de los números ingresados: 6

La suma de los digitos es: 6

* Ingresar números de varias cifras: 123, 234 ,456,0

RESULTADO OBTENIDO:

ingrese un número o cero '0' para finalizar: 123

La suma de los digitos es: 6

ingrese un número o cero '0' para finalizar: 234

La suma de los digitos es: 9

ingrese un número o cero '0' para finalizar: 456

La suma de los digitos es: 15

ingrese un número o cero '0' para finalizar: 0

Finalizó el programa

Sumatoria total de los números ingresados: 813

La suma de los digitos es: 12

* ingresar el número cero para finalizar el programa.

RESULTADO OBTENIDO:

Finalizó el programa

**Ejercicio Nro. 4:**

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

Se define la función esPrimo(numero) que toma un número como argumento y devuelve un valor booleano que indica si el número es primo o no. Dentro de la función esPrimo, se inicializa un contador en 0. Luego, se itera sobre cada número desde 1 hasta el número ingresado (range(1, numero+1)). Por cada iteración, se incrementa el contador si el número es divisible entre el iterador sin dejar resto es decir, si numero % i == 0.

Luego, se aplican las siguientes condiciones para determinar si el número es primo:

* Si número es igual a 1 o si número es negativo, se establece esPrimo como False, ya que según la definición de números primos, el número 1 y los números negativos no son primos.
* Si el contador es igual a 2, se establece esPrimo como True, ya que indica que el número solo tiene dos divisores: 1 y él mismo, cumpliendo así la definición de un número primo.
* En cualquier otro caso, se establece esPrimo como False.

Finalmente, se solicita al usuario que ingrese un número entero y se llama a la función esPrimo con ese número como argumento. El resultado se imprime en pantalla para indicar si el número ingresado es primo o no.

TESTEO:

* Números primos: 2, 3, 11

RESULTADO OBTENIDO:

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: 2  
True

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: 3  
True

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: 11  
True

* Números no primos: 4, 6,12

RESULTADO OBTENIDO:

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: 4

False

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: 6

False

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: 12

False

* Números negativos: -2, -3, -15

RESULTADO OBTENIDO:

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: -2

False

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: -3

False

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: -15

False

* Número 1: ya que 1 no se considera primo.

RESULTADO OBTENIDO:

----------------¿ES UN NÚMERO PRIMO?---------------------

Ingrese un número entero: 1  
False

**Ejercicio Nro. 5:**

**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.**

En primer lugar, se define la función dniValido(numeroDni) que toma un número de DNI como argumento. Dentro de esta, se utiliza una estructura condicional if para verificar si la longitud del número de DNI es menor que 7 o mayor que 8. Si se cumple alguna de estas condiciones, se establece la variable dni como False. De lo contrario, se establece dni como True. Finalmente, la función devuelve el valor de dni.

Por último, se solicita al usuario que ingrese un número de DNI y se llama a la función dniValido con el número ingresado como argumento. El resultado (True o False) se imprime en pantalla para indicar si el número de DNI es válido o no.

TESTEO:

* Número de DNI con 7 dígitos: 1234567

RESULTADO OBTENIDO:

----------------¿NÚMERO DE DNI VALIDO?---------------------

Ingrese el número de D.N.I.: 1234567

True

* Número de DNI con 8 dígitos: 12345678

RESULTADO OBTENIDO:

----------------¿NÚMERO DE DNI VALIDO?---------------------

Ingrese el número de D.N.I.: 12345678

True

* Número de DNI con menos de 7 dígitos: 12345

RESULTADO OBTENIDO:

----------------¿NÚMERO DE DNI VALIDO?---------------------

Ingrese el número de D.N.I.: 12345

False

* Número de DNI con más de 8 dígitos: 123456789

RESULTADO OBTENIDO:

----------------¿NÚMERO DE DNI VALIDO?---------------------

Ingrese el número de D.N.I.: 123456789

False

**Ejercicio Nro. 6:**

**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.**

Se define la función longitudUltimaPalabra(texto) que toma un string como argumento. Dentro de la función, se utiliza el método split() para dividir el string en una lista de palabras. Por defecto, el método split() divide el string utilizando espacios en blanco como separadores. Luego, se accede al último elemento de la lista (la última palabra) utilizando el índice -1[[1]](#endnote-1). Esto asegura que se obtenga la última palabra, incluso si hay espacios al final del string. Finalmente, se devuelve un string que indica la longitud de la última palabra, utilizando la función len() para obtener la longitud de la palabra.

Fuera de la función, se solicita al usuario que ingrese un texto y se llama a la función longitudUltimaPalabra con ese texto como argumento. El resultado se imprime en pantalla para indicar la longitud de la última palabra.

TESTEO:

* "Hola mundo": Devolverá la longitud de la palabra "mundo", que es 5.

RESULTADO OBTENIDO:

----------------LONGITUD DE LA ULTIMA PALABRA---------------------

ingrese un texto: Hola mundo

La longitud de la palabra mundo es: 5

* " Esto es una prueba ": Devolverá la longitud de la palabra "prueba", que es 6.

RESULTADO OBTENIDO:

----------------LONGITUD DE LA ULTIMA PALABRA---------------------

ingrese un texto: Esto es una prueba

La longitud de la palabra prueba es: 6

* " Última palabra ": Devolverá la longitud de la palabra "palabra", que es 7.

RESULTADO OBTENIDO:

ingrese un texto: Última palabra

La longitud de la palabra palabra es: 7

**Ejercicio Nro. 7**

**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 vacío. 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 sólo 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 primeros 3 dígitos de su DNI. Ejemplo:**

**Nombre: Alba María Linares**

**DNI: 25834910**

**Alba7258.**

Se define un diccionario llamado socios para almacenar los datos de los socios, una bandera para controlar el bucle y una variable posición para llevar un registro de la posición actual en la lista de claves del diccionario.

Se utiliza la función dniValido(numeroDni) del ejercicio N°5 para validar que el número de DNI tenga entre 7 y 8 dígitos. También, se vuelve a utilizar la función generarListaDigitos(numeroIngresado) del ejercicio N°3 para generar una lista de dígitos a partir del número de DNI ingresado. Además, se reutiliza y se adapta al programa la función para obtener la longitud de la última palabra del ejercicio N°6 como longitudApellido(texto) para obtener la longitud del apellido.

Se hace uso de un bucle while para permitir al usuario ingresar datos de los socios hasta que ingresa un nombre vacío. Dentro de este bucle suceden los siguientes pasos:

* Solicita al usuario ingresar el nombre o dejar vacío para terminar el programa.
* Divide el nombre ingresado en una lista de palabras utilizando el método split().
* Solicita al usuario que ingrese el apellido. Se guarda la longitud del mismo en la variable cantidadLetras llamando a la función longitudApellido(texto)
* Solicita al usuario ingresar el número de DNI
* Utiliza la función dniValido(numeroDni) para verificar si el número de DNI ingresado es válido. Si el DNI no es válido, solicita al usuario que ingrese un número de DNI hasta que este sea correcto mediante el uso de un bucle while.
* Se define la variable primerosNumeros como vacía
* Utiliza la función generarListaDigitos(numeroIngresado) para obtener los primeros 3 dígitos del número de DNI ingresado. Se utiliza un bucle for para iterar sobre los primeros tres números del DNI. Esto se realiza con range(3), que crea un rango de números del 0 al 2. En cada iteración del bucle, se llama a la función generarListaDigitos(dni). Se accede a los primeros tres dígitos del DNI utilizando la notación de indexación [numero]. Cada dígito se concatena a la variable primerosNumeros, lo que acumula los tres primeros dígitos del DNI.
* Agrega al diccionario socios una clave única que consiste en el primer nombre (nombre[0]), la cantidad de letras del apellido (cantidadLetras) y los primeros 3 dígitos del DNI (primerosNumeros), y se le asigna una lista vacía como valor. Luego, se añaden los datos del socio a la lista asociada con la clave única mediante la función append() (Nombre, Apellido y DNi)
* Se imprime el nombre y apellido del socio. Verificando la longitud de la lista nombre. Si el nombre consiste en dos partes (nombre y segundo nombre), se imprimen ambos; de lo contrario, se imprime solo el primer nombre.
* Se imprime el número de DNI del socio.
* Se imprime la clave única asociada al socio, accediendo a la lista de claves del diccionario socios y seleccionando la clave en la posición especifica que contiene el valor de la variable posición.
* Se incrementa la variable posición en una unidad para apuntar a la próxima posición en la lista de claves del diccionario socios.
* Continúa solicitando datos de socios hasta que el usuario ingresa un nombre vacío.

TESTEO:

1)

Nombre: Javier Alejandro

Apellido: Dileo

DNI: 1234567

RESULTADO OBTENIDO:

Ingrese Nombre (Para finalizar, no ingrese el nombre): Javier Alejandro  
Ingrese Apellido (si posee dos apellidos sólo ingrese el primero): Dileo  
Ingrese el N° de D.N.I.: 1234567

Nombre: Javier Alejandro Dileo  
DNI: 1234567  
Clave unica: Javier5123

2)

Nombre: Javier Alejandro

Apellido: Dileo

DNI: 123456 , 7654321

RESULTADO OBTENIDO:

Ingrese Nombre (Para finalizar, no ingrese el nombre): Javier Alejandro  
Ingrese Apellido (si poseé dos apellidos sólo ingrese el primero): Dileo  
Ingrese el N° de D.N.I.: 123456  
Ingrese el N° de D.N.I.: 7654321

Nombre: Javier Alejandro Dileo  
DNI: 7654321  
Clave unica: Javier5765

3)

Nombre vacío, debe finalizar el programa.

RESULTADO OBTENIDO:

Ingrese Nombre (Para finalizar, no ingrese el nombre):  
Finalizó el programa

**Ejercicio Nro. 8:**

**Escribir la función titulo(), la cual recibe un string y lo retorna convirtiendo la primera letra de cada palabra a mayúscula y las demás letras a minúscula, dejando inalterados los demás caracteres. Precondición: el separador de palabras es el espacio: " ". Agregar doctests con suficientes casos de prueba para validar que la función retorna el valor esperado ante distintos argumentos.**

Se define la función titulo(cadena). Toma un argumento cadena, que es la cadena que se va a formatear según lo especificado en el enunciado. Se inicializa una cadena vacía nuevaCadena que se utilizará para almacenar la cadena reformateada. La cadena de entrada “cadena” se divide en palabras utilizando el método split(). Luego, se itera sobre cada palabra en la lista de palabras resultante “cadena”. Para cada palabra, se utiliza el método capitalize() para convertir la primera letra en mayúscula y las demás letras en minúscula. Las palabras modificadas se almacenan en una nueva lista llamada “palabras”. Las palabras de la lista palabras se unen de nuevo en una cadena utilizando el método join(), [[2]](#endnote-2)con espacios como separador. Esto restaura la cadena capitalizada. Finalmente se retorna la nueva cadena reformateada.

Se han agregado doctests que incluyen varios casos de prueba para validar que la función retorna el valor esperado ante distintos argumentos. Estos casos de prueba cubren diferentes situaciones, como palabras en mayúsculas, minúsculas o mixtas.[[3]](#endnote-3)

Nota: Los test cases dan como resultado Fail, pero el programa hace lo solicitado. Quizás sea algún error de como maneja el doctest los casos de prueba.

Failed example:

titulo('HOLA COMO ESTAS')

Expected:

'Hola Como Estas'

Got:

'Hola Como Estas'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Failed example:

titulo("hOlA cOmO eStAs")

Expected:

'Hola Como Estas'

Got:

'Hola Como Estas'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Failed example:

titulo("hoLA coMO esTAs")

Expected:

'Hola Como Estas'

Got:

'Hola Como Estas'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Failed example:

titulo("holA comO estaS")

Expected:

'Hola Como Estas'

Got:

'Hola Como Estas'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Failed example:

titulo("HoLa CoMo EsTaS")

Expected:

'Hola Como Estas'

Got:

'Hola Como Estas'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1 items had failures:

5 of 5 in \_\_main\_\_.titulo

\*\*\*Test Failed\*\*\* 5 failures.

1. Referencia obtenida por medio del chatGPT [↑](#endnote-ref-1)
2. Referencia obtenida por medio del chatGPT [↑](#endnote-ref-2)
3. Referencia obtenida por medio del chatGpt [↑](#endnote-ref-3)