**Ejercicios conceptos básicos Python**

Abrir temario playlist\_play

Lo lees en 1 Min.

 Activar modo nocturno

Los siguientes, son una serie de ejercicios que tienen como finalidad el que tu practiques los conocimientos adquiridos a lo largo de este segundo bloque.

* Dado de los valores ingresados por el usuario (base, altura) calcular y mostrar en pantalla el área de un triángulo.
* Convertir la cantidad de dólares ingresados por un usuario a pesos colombianos y mostrar el resultado en pantalla.
* Convertir los grados centígrados ingresados por un usuario a grados Fahrenheit y mostrar el resultado en pantalla.
* Mostrar en pantalla la cantidad de segundos que tiene un lustro.
* Calcular la cantidad de segundos que le toma a la luz viajar del sol a Marte y mostrar el resultado en pantalla.
* Calcular el número de vueltas que da una llanta en 1 km, dado que el diámetro de la llanta es de 50 cm, mostrar el resultado en pantalla.
* Calcular y mostrar en pantalla la longitud de la sombra de un edificio de 20 metros de altura cuando el ángulo que forman los rayos del sol con el suelo es de 22º.
* Mostrar en pantalla True o False si la edad ingresada por dos usuarios es la misma.
* Mostrar en pantalla la cantidad de meses transcurridos desde la fecha de nacimiento de un usuario.

Tenemos un [artículo](https://codigofacilito.com/articulos/fechas-python) en donde explicamos más en detalle el trabajo con fechas

* Mostrar en pantalla el promedio de un alumno que ha cursado 5 materias (Español, Matemáticas, Economía, Programación, Ingles).

# Ejercicios de estructuras Python

Abrir temario playlist\_play

Lo lees en 1 Min.

 Activar modo nocturno

Los siguientes, son una serie de ejercicios que tienen como finalidad el que tu practiques los conocimientos adquiridos a lo largo de estos bloques.

* Dado un diccionario, el cual almacena las calificaciones de un alumno, siendo las llaves los nombres de las materia y los valores las calificación, mostrar en pantalla el promedio del alumno.

Ejemplo: calificaciones = {calculo:10, dibujo:5}

* A partir del diccionario del ejercicio anterior, mostrar en pantalla la materia con mejor promedio.
* Crear una lista la cual almacene 10 números positivos ingresados por el usuario, mostrar en pantalla: la suma de todos los números, el promedio, el número mayor y el número menor.
* Dado una lista de frases ingresadas por el usuario, mostrar en pantalla todas aquellas que sean palíndromo.
* Mostrar en pantalla la palabra que más se repita junto con la cantidad de veces que lo hace del capituló número uno de [Frankenstein](http://umich.edu/~umfandsf/other/ebooks/frank10.txt)
* Remplazar cada letra de una frase dada por el usuario por la posición que le corresponde en el abecedario y mostrar el nuevo string en pantalla. (Los espacios no se remplazan) .Ejemplo: frase : 'Hola' salida : 815121 H(8) o(15) l(12) a(1)
* Mostrar en pantalla la cantidad de vocales que existe en una frase dada por el usuario.
* Mostrar en pantalla la frecuencia de aparición de vocales en una frase dada por el usuario.

Ejemplo : 'Hola Mundo' salida : o=2, a=3, u=1

* Eliminar todas las vocales de una frase dado por el usuario y mostrar el nuevo string en pantalla.
* Listar todos los números pares del 0 al 100

# Documentación de las funciones

Abrir temario playlist\_play

Lo lees en 1 Min.

 Activar modo nocturno

Cómo mencionamos anteriormente, una vez que nosotros definimos una función, podemos llamarla **n** cantidad de veces, inclusive, fuera de nuestro script, cómo lo veremos más adelante (módulos y paquetes) es por ello que una muy buena practica de programación es documentar nuestras funciones.

Para que nosotros podamos documentar una función lo haremos mediante un comentario, comentario, el cual debe de encontrarse dentro de la función y utilizando triples comillas dobles, cómo podemos observar en el siguiente ejemplo.

def mi\_funcion(\*args):

"""Esta es la documentación de mi\_función"""

print(args)

Recordemos que al utilizar triples comillas dobles podemos colocar un comentario con saltos de línea.

Podemos trabajar con la documentación a través de su atributo \_\_\_\_doc\_\_\_\_

print(mi\_funcion.\_\_doc\_\_)

Ahora veamos un ejemplo en el cual podemos sacar provecho a nuestra documentación.

def suma(a, b):

"""Función suma"""

return a + b

def resta(a, b):

"""Función resta"""

return a - b

opciones = {'a' : suma, 'b': resta}

print("Ingrese la opción deseada")

for opcion, funcion in opciones.items():

mensaje = '{}) {}'.format(opcion, funcion.\_\_doc\_\_)

print(mensaje)

opcion = input("Opción : ")

Almacenamos las funciones dentro de nuestro diccionario, posteriormente iteramos los elementos del diccionario y en cada iteración imprimimos la documentación.

# == o is

Abrir temario playlist\_play

Lo lees en 1 Min.

 Activar modo nocturno

### == O IS

En el vídeo de [Operadores relacionales y lógicos](https://codigofacilito.com/videos/operadores-relacionales-y-logicos" \t "_blank), comentamos que es posible conocer si dos valores enteros son iguales mediante el uso de **==** y la palabra reservada **is**; Sin embargo, ahora que ya contamos con más conocimiento del que teníamos en ese vídeo es importante mencionar cuando usaremos **==** y cuando usaremos **is**.

Si nosotros ejecutamos la siguiente línea de código :

[1,2,3] == [1,2,3]

Tendremos cómo resultado **True**, debido a que ambas listas son iguales. Ahora, ¿Qué pasa si nosotros ejecutamos la siguiente línea de código?,

[1,2,3] is [1,2,3]

¿Nuevamente tendremos **True**? No, el resultado será **False**; Esto se debe a que con **==**compararemos que dos valores sean iguales y con **is** compareremos que dos objetos sean iguales, cosas completamente diferentes.

Hagamos un par de modificaciones para que nos quede más en claro. A la primera lista la llamaremos **a** y a la segunda **b**

a = [1,2,3]

b = [1,2,3]

Si nosotros imprimimos el **id** de cada objeto lista, observaremos que son valores completamente diferentes, con lo cual concluimos que son dos objetos diferentes.

print(id(a))

print(id(b))

Si nosotros ejecutamos el siguiente script.

a = [1, 2, 3]

b = a

a is b

Obtendremos cómo resultado **True**, debido a que ambos objetos son iguales.

De esta forma sabremos cuando utilizar **==** y cuando utilizar **is**.