**Los lenguajes de programación más usados en la actualidad**

El desarrollo de software está a la orden del día, y cada vez son más las empresas que contratan profesionales de este sector para elaborar sus propias aplicaciones móviles, para encargarse de la web corporativa, de producto o el e-commerce.

### Java: **ofrece una gran portabilidad y puede ejecutarse en casi cualquier sistema**. Es un lenguaje de tipo estático, por lo que es rápido y fácil de mantener, con pocos errores.

Si es el diseño Web, de one.

Se divide en varios lenguajes, Javascript, AQL MySycual (Bases de datos) V:

### Python: **capacidades en**[***machine learning***](https://www.universia.net/es/actualidad/orientacion-academica/ventajas-de-implementar-machine-learning.html)**, ciencia de datos e inteligencia artificial. se utiliza para desarrollar aplicaciones de todo tipo**. aplicaciones web, el análisis de datos y el desarrollo de algoritmos. Tiene una sintaxis fácil de usar y se centra en la legibilidad y la simplicidad del código convirtiéndolo en un candidato ideal para los desarrolladores de nivel de entrada, especialmente para aquellos que persiguen una carrera en inteligencia artificial, big data, robótica o ciberseguridad.

Este lenguaje es sencillo, Python te dice todo, pon un espacio, Es el “Java para principiantes”, empieza simple, y se vuelve complejo.

Diseño de apps para pc, la mayoría están basadas en Python. Se maneja una cantidad grande de información. Librería de Pandas (Bases de datos, sin necesidad de instalar controladores). Es mejor para manejar grandes cantidades de datos.   
Puentes de conexión interlenguaje (sirven para ir entre lenguajes).

### C/C++: **algunos sistemas operativos están creados en este lenguaje** y lo puedes encontrar en una amplia variedad de software, como navegadores web, sistemas de distribución, software de base de datos o videojuegos.

SI O SI NECESITO ESTE CONOCIMIENTO.

### C#: efectivo **a la hora de crear aplicaciones de escritorio y juegos de Windows**. Se puede usar para desarrollar aplicaciones web y [móviles](https://www.universia.net/es/actualidad/orientacion-academica/la-revolucion-de-los-desarrolladores-de-apps-moviles.html) y, a menudo, se usa para aplicaciones de grandes empresas, como en la industria del procesamiento de transacciones bancarias.  es popular en la industria de los juegos ya que el 34% de los juegos móviles se realizan con C#, junto con aplicaciones de realidad virtual. Está presente en juegos que usan Unity, como Temple Run Trilogy o Assassin's Creed.

Se pronuncia (CI CHAR :V)

### Kotlin: Google declaró esta tecnología prioritaria, es decir, los nuevos desarrollos se harían para Kotlin y, una vez terminados, se implementarían en Java. Esto hizo que el sector móvil se diese cuenta de la importancia de dicho lenguaje y se centrara en él. Actualmente, **más del 80% de las 1.000 aplicaciones más populares para Android usan Kotlin**. **se recomienda comenzar aprendiendo Java.**

### Swift: **te permite publicar tus aplicaciones en la AppStore** así que si quieres hacerte un hueco en este mercado debes empezar por aprender Swift.  **se caracteriza por ser claro y conciso.**

### PHP y HTML: **es uno de los lenguajes que mueven la web. (Aprender tambn)**

### Javascript: **se usa, sobre todo, en el desarrollo web**para agregar interactividad a un sitio, ya sea en forma de juegos, botones, animaciones o formularios. Hay más de 1.700 millones de sitios web en el mundo y se utiliza en aproximadamente el 95 % de ellos, lo que hace que el alcance de JavaScript sea muy grande.

## Front-end, back-end, ¿en qué se diferencian?

El **desarrollo front-end** se refiere a la programación de todo lo que está a la vista del usuario: todo lo que se ejecuta del lado del cliente, la capa de presentación o capa *User Interface*.

Este también es responsable del diseño de interacción, es decir, de la manera en que el usuario se relaciona con la aplicación a través del navegador, o de la interfaz táctil (en el caso de dispositivos móviles).

Como programador front-end debes conocer los diferentes lenguajes, frameworks y librerías de desarrollo, pero también debes tener conocimientos de diseño. Así, podrás encontrar la mejor manera de presentar la información según los objetivos de la aplicación.

Vamos ahora con el perfil de **desarrollo de back-end**. En este caso nos estamos refiriendo a la parte de la programación que no es visible para el usuario. Es todo lo que *va*detrás de la interfaz, la lógica interna. Esta es la parte que se ejecuta en un servidor que, a su vez, puede ser local o estar ubicado [en la nube](https://www.universia.net/es/actualidad/empleo/quieres-trabajar-en-la-nube-te-contamos-los-perfiles-mas-buscados-en-cloud-computing.html).

La programación back-end está muy relacionada con el diseño y la creación de la arquitectura de software. De esta manera, todo lo que es la lógica del negocio es tarea del *back-end developer*, que será quien se encargue de procesar toda la información, relacionar datos y otras tareas.

Debes dominar múltiples lenguajes y frameworks, bases de datos, protocolos como HTTP y multitud de herramientas, manejar conceptos abstractos y lógica compleja.

## ¿Y Full-Stack?

Un **desarrollador Full-Stack** es capaz de moverse entre el Front-End y el Back-End. El perfil del desarrollador Full-Stack tiene una enorme demanda en la actualidad porque puede incorporarse a muchos tipos de operaciones y ser productivo en cualquier parte del desarrollo de una aplicación web.

Este perfil debe conocer y dominar muchas tecnologías y herramientas. Debe ser capaz de entenderlas y saber qué frameworks necesita para llevar a cabo cada proyecto. El desarrollador Full-Stack es un perfil todoterreno. Además, debe tener las habilidades comunicativas necesarias para estar en sintonía con el resto de los desarrolladores del equipo.

## Otras salidas profesionales para los desarrolladores de software

Además, los **desarrolladores software** pueden realizar muchas **otras tareas y especializaciones**. Resumimos las posibilidades:

* Desarrollo de páginas web y de aplicaciones móviles.
* Diseño y arquitectura de software.
* Programación y análisis.
* Administración de redes y sistemas.
* Diseño y Administración de BBDD.
* Gestión de entornos de seguridad.
* Inteligencia artificial, para mejorar procesos y desarrollo de sistemas que facilitan la vida a las personas. Es una de las ramas con mayor expansión a futuro.
* Otras especializaciones como:

o   diseño de videojuegos;

o   docencia;

o   investigación;

o   puestos de dirección;

o   técnico de procesos de calidad;

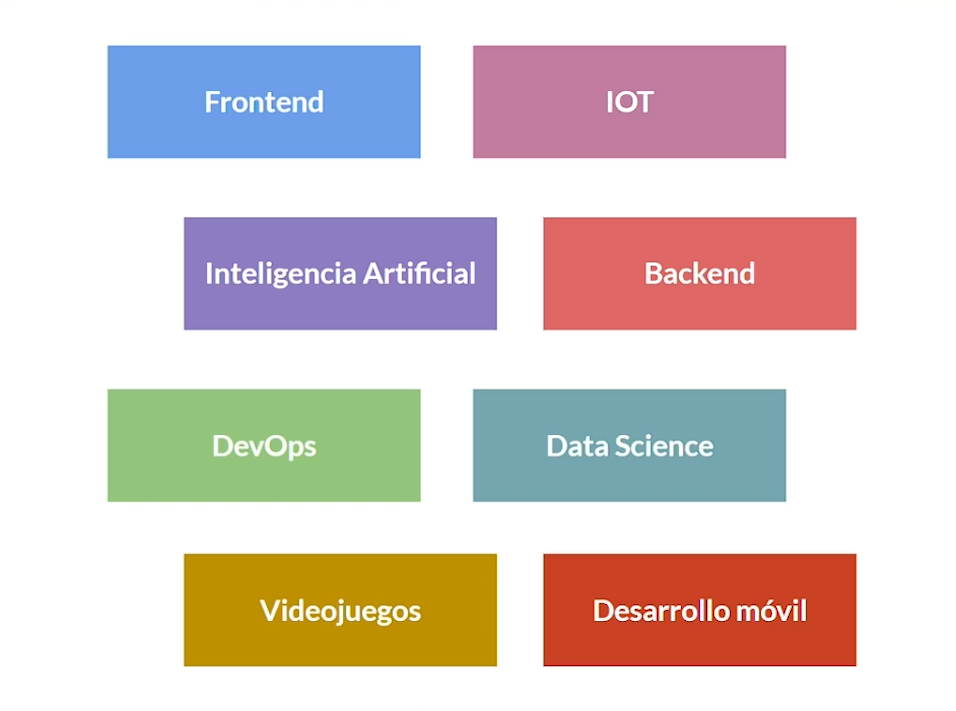
o   emprendimiento tecnológico.

IMPORTANTE:

Es super importante aprender GIT. Ver Tutorial en YT para abrir un Github (Para guardarlo y crear un portafolio).

Que es programar: Darle instrucciones a la computadora, para que resuelva un problema.

A que me quiero enfocar con mi proyecto?



Me gustaría Inteligencia Artificial!!!

O desarrollo de videojuegos.

O la app para mejorar en Lol mecánicamente.

IOT= Internet de las cosas, Como conectar a internet electrodomésticos.

Python funciona muy bien en IOT, IA, Backend y Data Science

Ventajas: Buenas prácticas: “Espaciado”, y paralelo al inglés.

**Desafio**: Buscar una app que Python sea usada.

**Solución:**

Los programas en Python más famosos:

\*Pinterest.

\*Panda 3D.

\*Dropbox.

\*Spotify.

\*Netflix.

\*Uber.

\*Instagram.

\*Reddit.

\*TY.

***COMENZAMOS CON LA TEORIA SOBRE PYTHON!! (En Platzi)***

**El Algoritmo:**

\*Serie de pasos ordenados para resolver un problema V:

Planeamos, Ejecutamos, y repetimos el proceso.

\*Es Finito y no ambiguo (tiene una finalidad en particular, y no que cambie dependiendo del contexto).

- Un ejemplo puede ser un DIAGRAMA DE FLUJO –

Que necesitamos para programar?

1. Visual Studio Code (TXT para programar)
2. Lenguaje Phyton.
3. Consola: Cmder.

Principio de la consola: Cmder!

Básicos para usar la consola:

clear = Limpiar pantalla  
CD = Change Directory  
… = Carpeta padre  
CD… = Cambiar de directorio a la carpeta padre  
Alt + 92 = Es el Backslash \  
ls = list  
mkdir = Crear Carpetas.  
touch = para crear archivos (touch \*Espacio\* .txt)

touch archivo{01,02,03}.txt (Crea Varios archivos)

**Otros comandos utilizados**

ls -la Listar y ver archivos ocultos

pwd Ver en que directorio estoy

rm Borrar

mv Mover archivo

rmdir Borrar directorio

ls -t Ordena los archivos por fecha de modificacion

ls -x Ordena elementos por nombre y extension

ls -X Orcena elementos por extension y nombre

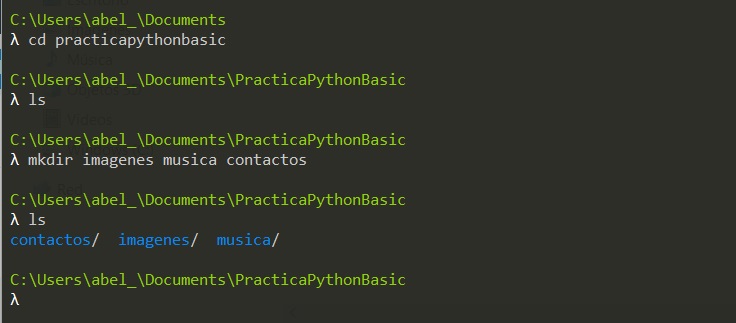
ls -S Ordena los archivos por tamaño de archivo

cat Muestra el contenido completo de un archivo

head Muestra las primeras lineas de un archivo

tail Muestra las ultimas lineas de un archivo

**CREAR VARIOAS CARPETAS:**



Que es lo más sencillo en Python? Operaciones matemáticas.

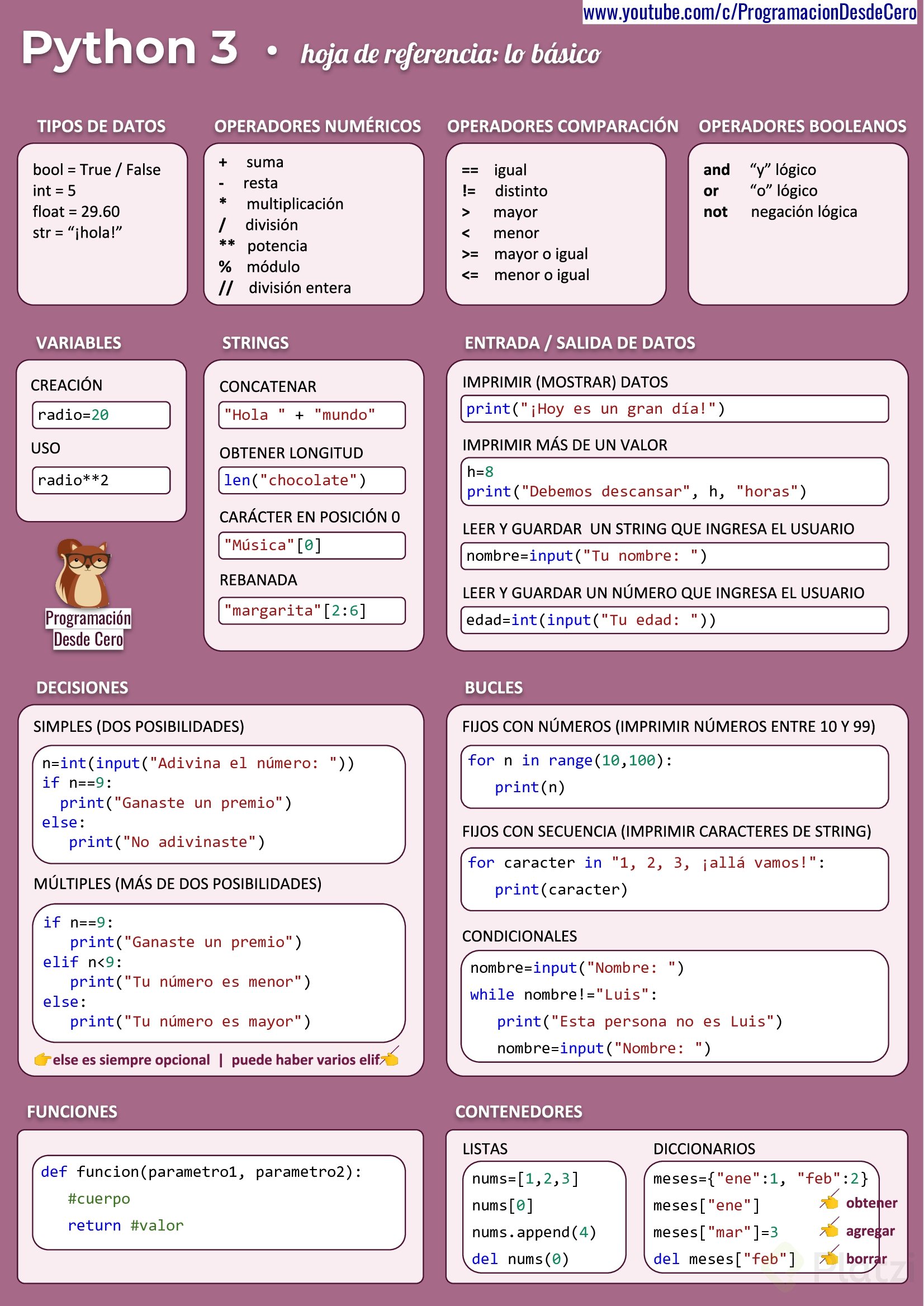
Operaciones Matematicas.

Consola interactiva de Python.

Consola interactiva= Ejecutar lieas de código, de manera independiente, ejecutarla de una vez.

Cuando necesitemos escribir más de una línea, nos iremos al editor de código.

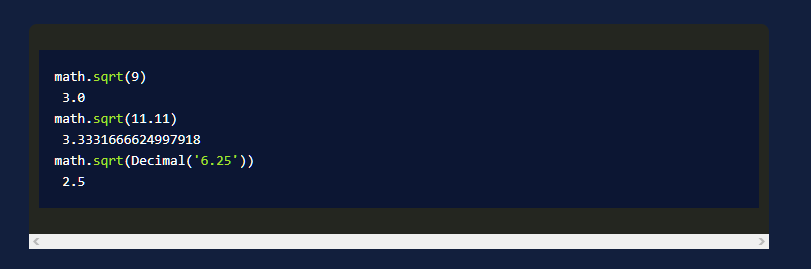
Para entrar a la consola escribimos “py” y para salir, escribimos “Exit ()”



¿Como calcular raíz cuadrada en python?  
A.-Vía la función estándar math.sqrt() (square root):

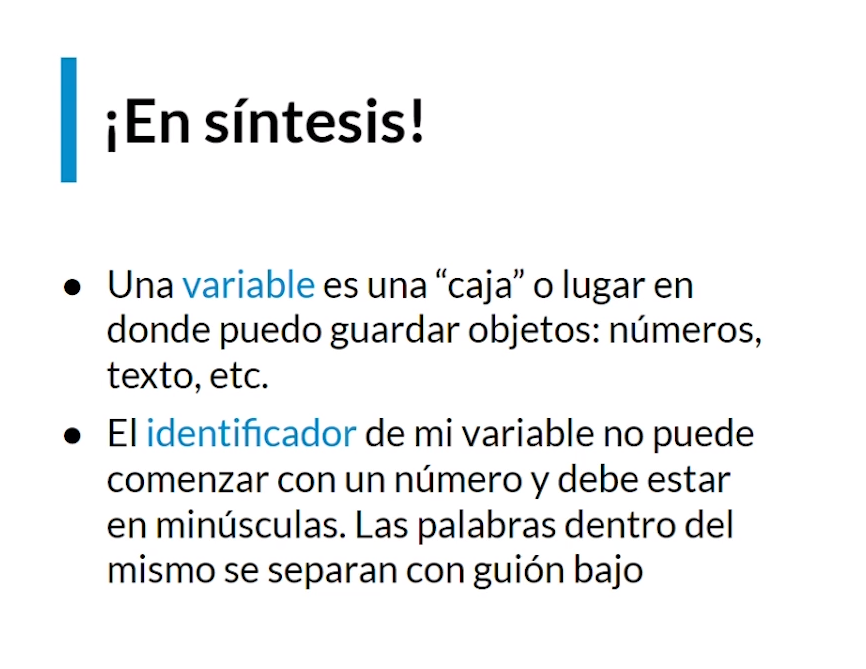
from math import sqrt  
sqrt(81)  
9.0  
B.-Una alternativa es elevar al exponente 0.5:

81 \*\* 0.5  
9.0



**Que es una variable?**

Símbolo constituyente de un predicado, formula o algoritmo. Designa una cantidad susceptible de posibilidades. (Es una caja con distintas posibilidades) El nombre es identificador.



## Asignación de variables

En Python, creamos las variables asignándoles un valor de la siguiente manera:  
<identificador> = <valor>  
en este caso el signo = se lee como “asignar”

## Reasignación de variables

Podemos en cualquier momento cambiar el valor de nuestra variable volviendo a asignar un valor al mismo identificador:  
<identificador> = <nuevo\_valor>

## Tipos de variables en Python

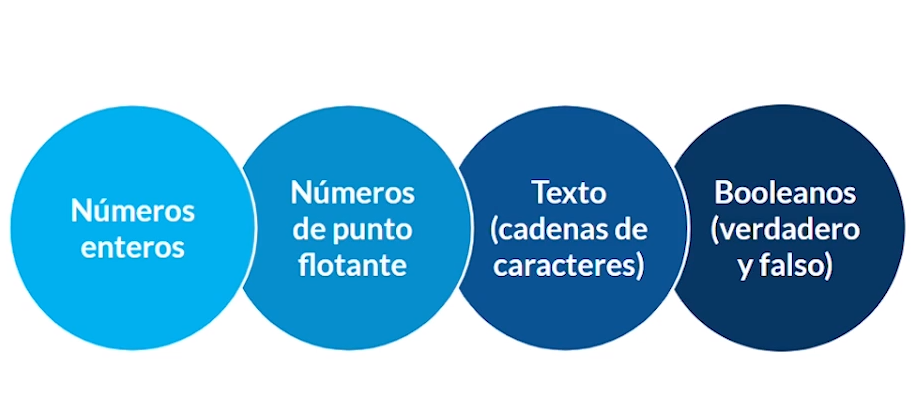
a = 28 → int (entero)  
b = 1.5 → float (decimales)  
c = “Hello” → str (string o cadena de texto)  
d = True → boolean (verdadero o falso)  
e = None → NoneType (Sin valor)  
f = “5” → str (5 y “5” no son lo mismo. La primera es un entero y la segunda una cadena de texto)

“Programación orientada a objetos”= Paradigma de programación.

Objetos= Tipos de Datos.

“Todo es un objeto”.

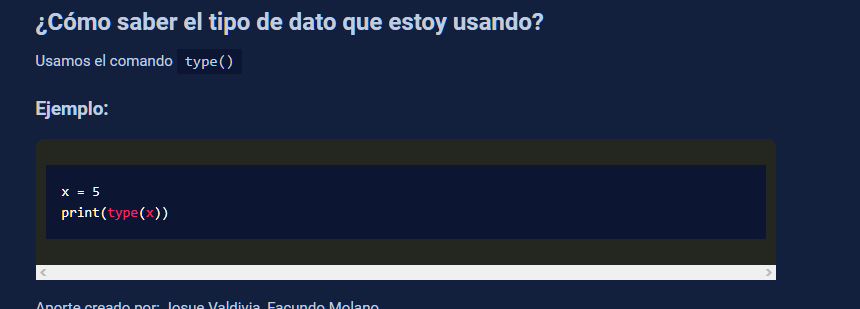
4 tipos de datos clásicos:



Todo lo que este entre comillas es TEXTO, cadena de caracteres

## Tipos de dato adicionales

* Datos en texto: str
* Datos numéricos: int, float, complex
* Datos en secuencia: list, tuple, range
* Datos de mapeo: dict
* Set Types: set, frozenset
* Datos booleanos: bool
* Datos binarios: bytes, bytearray, memoryview



Convertir un dato a un tipo diferente:



Operadores lógicos y de comparación:

* **and**  Para comparar si dos valores son verdaderos.  
  -**or**  Para comparar si uno de los 2 valores es verdadero.  
  -**not**  Para invertir el valor booleano.  
  -**==**  Compara dos valores y te dice si son iguales o no.  
  -**!=**  Compara dos valores y te dice sin son diferentes o no.  
  -**>**  Compara si es mayor que otro valor.  
  -**>**  Compara si es menor que otro valor.  
  **>=**  igual o mayor que el valor a comparar.  
  **<=**  igual o menor que el valor a comparar.

**Nuestro primer programa (Conversor de monedas):**

Este es el código: 3

Un atajo para las triples comillas que pueden usar es **Shift + Alt + A**

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

        Hola, nuestro programa puede calcular tu cambio de divisa de COP a dolares, o viceversa.💲🤑 🎨

            [1] COP a Dolares.

            [2] Dolares a COP.

        Haz seleccionado: """))

print ("        \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

#COP a Dolares

if calcular == 1:

    cop = input("¿Cuántos pesos Colombianos tienes?: ")

    cop = float(cop)

    valor\_dolar = 4369

    dolares = cop / valor\_dolar

    dolares = round(dolares, 2)

    dolares = str(dolares)

    print("Tienes: $" + dolares + " dólares.")

#Dolares a COP

elif calcular == 2:

    dolares = input("Cuantos Dolares tienes?: ")

    dolares = float(dolares)

    valor\_dolar = 4369

    cop = dolares \* valor\_dolar

    cop = int(cop)

    cop = str(cop)

    print("Tienes $" + cop + " Pesos Colombianos")

#Otro

else:

        print('Ingresa solo un numero de la lista')

## Diferencias entre if, else y elif

### if:

if se encarga de iniciar el condicional y solicitar un requisito para ejecutar todo el código por debajo, que conocemos como bloque de código.

### else:

Si se desea ejecutar otro código en caso de que no se cumpla el if. Por ejemplo: el usuario no elige la opción 1, entonces (else)…

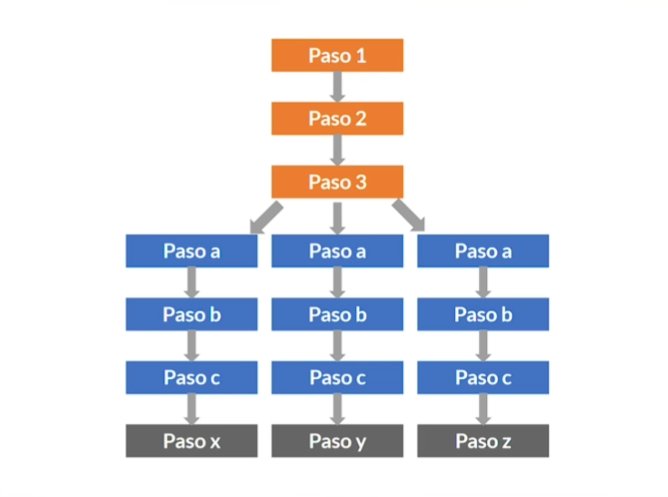
### elif:

Se utiliza cuando utilizamos múltiples condiciones, lo que en el código de esta clase son la opción 2 y 3. En esta clase, teníamos la opción 1, pero debemos también evaluar qué pasa si el usuario elige la opción 2 o 3, por lo que decimos “que estamos evaluando múltiples condiciones”.

## Añadir comentarios en Python

Para realizar un comentario (de una sola línea), empleamos el “#”. Un comentario es simplemente texto, el cual no es ejecutado y no afecta en absoluto en el código. Se utiliza para explicar las líneas de código que hemos creado y hacerlas más fáciles de entender.

Lo malo de este código es que repetimos código, Con las funciones solucionaremos esto:





**Código de mi función=**

def conversacion(num):

    print('Hola')

    print('Como estas?')

    print("Elegiste la opción " + num + ".")

    print('adios')

opcion = int(input('Elige una opción (1, 2, 3: )'))

if opcion == 1:

    conversacion('1')

elif opcion == 2:

    conversacion('2')

elif opcion == 3:

    conversacion('3')

else:

    print('Escribe la opción correcta')

**Modulizando nuestro conversor de monedas:**

def conversor(tipo\_pesos, valor\_dolar):

    pesos = input("¿Cuántos pesos " + tipo\_pesos + " tienes?: ")

    pesos = float(pesos)

    dolares = pesos / valor\_dolar

    dolares = round(dolares, 2)

    dolares = str(dolares)

    print("Tienes: $" + dolares + " dólares.")

menu = """ \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

        Hola, nuestro programa puede calcular tu cambio de divisa de COP, Pesos argentions o Mexicanos a dolares, o viceversa.💲🤑 🎨

            [1] COP a Dolares.

            [2] Pesos argentinos a dolares.

            [3] Pesos mexicanos a Dolares.

        Haz seleccionado: """

calcular= int(input(menu))

print ("        \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*")

#COP a Dolares

if calcular == 1:

    conversor("colombianos", 3875)

#Argentinos a Dolares

elif calcular == 2:

    conversor("argentinos", 65)

#Mexicanos a Dolares

elif calcular == 3:

    conversor("Mexicanos", 24)

#Otro

else:

        print('Ingresa solo un numero de la lista')

**Trabajando con Texto: Cadena de caracteres.**

LOS METODOS:

variable.upper() = 'todos los caracteres en MAYÚSCULAS'

variable.capitalize() = 'solo la primera en MAYÚSCULA'

variable.lower() = 'todos los caracteres en minúscula'

variable.strip() = 'eliminar espacios basura del string'

variable.replace('o', 'a') = remplazar caracter

FUNCIONES BUILT IN: ya vienen por defecto en Python y no toca crearlas.

len()

print()

input()

Índices:

Se escriben entre corchetes al lado de la variable y son apuntadores numéricos a cada caracter. Por ejemplo, para el nombre Facundo, cuando utilizamos la variable nombre[1], aparece la letra ‘a’, dado que dicha variable tiene almacenada en ese momento la cadena de caracteres ‘Facundo’ donde la ‘a’ es el segundo caracterer.

Aclaración: se comienza a contar caracteres desde el 0 (que es el primer número en informática). Siguiendo el ejemplo, la letra ‘F’ de ‘Facundo’ es el caracter número 0. Por ende, nombre[0], nos devolvería una F.

Si ejecutamos:

**dir**(variable)

Obtendremos todos los métodos disponibles para ejecutar de un objeto, variable o tipo de dato.

y si ejecutamos:

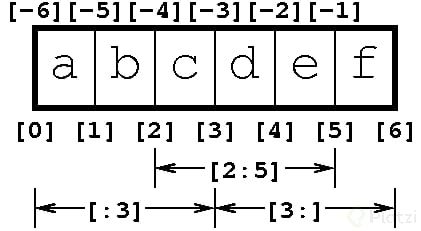
**help**(variable)

Nos entregará la documentación disponible para esa variable.

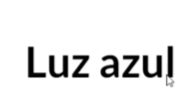
**Slices: Rebanadas.**



nombre = "Francisco" || nombre[0 : 3) = Fra || nombre[1:7] = "rancis" || nombre[1:7:2] = "rni" || nombre[1::3] = "rcc" || nombre[::-1] = "ocsicnarF"



Proyecto Palíndromo: Palabra leida al derecho y al revez, y es igual.



def palindromo(palabra):

    palabra = palabra.replace(' ', '')

    palabra = palabra.lower()

    palabra\_invertida = palabra[::-1]

    if palabra == palabra\_invertida:

        return True

    else:

        return False

def run():

    palabra = input('Escribe una palabra: ')

    es\_palindromo = palindromo(palabra)

    if es\_palindromo == True:

        print('Es palíndromo')

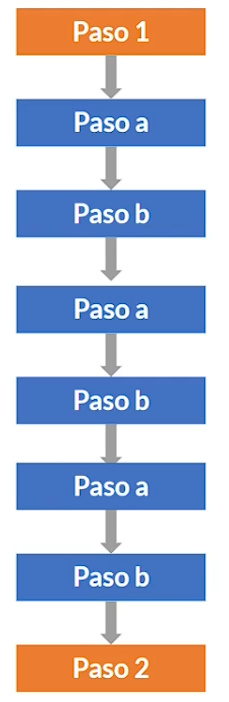
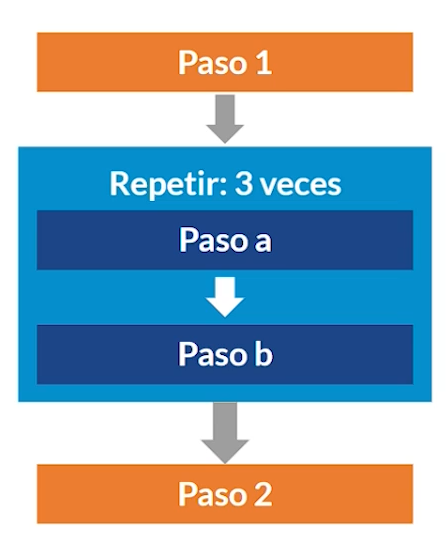
    else:

        print('No es palíndromo')

if \_\_name\_\_ == '\_\_main\_\_':

    run()

Aprendiendo Bucles:



Tarea: Realizar un programa para imprimir todas las potencias de un número hasta el mil:

MI CODIGO:

#Imprimir todas las potencias de 2 hasta llegar al numero mil.

def operacion(numero, potencia):

    resultado = numero\*\*potencia

    while (resultado <= 1000):

        resultado = numero\*\*potencia

        print(str(numero) + " Elevado a la " + str(potencia) + " Es igual a: " + str(resultado))

        potencia = potencia + 1

    else:

        print("Hemos terminado")

def run():

    numero = input("Escribe el numero al que quieres hallarle las potencias: ")

    print("Comenzamos con " + numero)

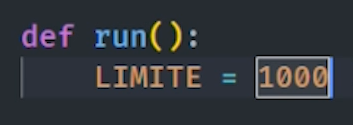
    numero = int(numero)

    operacion(numero, 2)

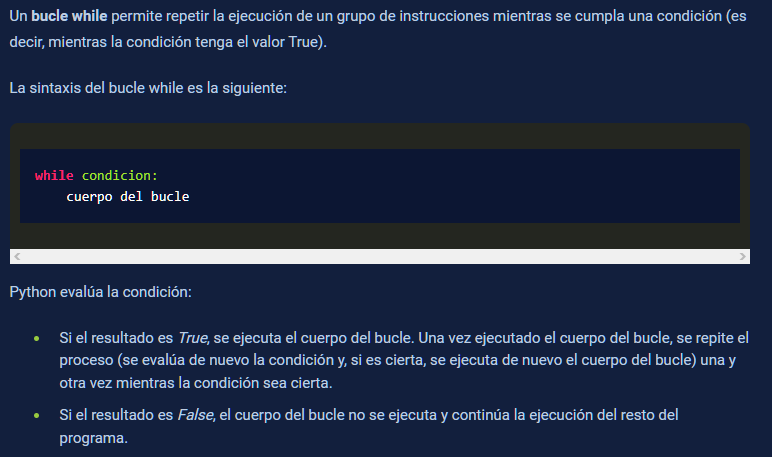
if \_\_name\_\_ == '\_\_main\_\_':

    run()

Para definir CONSTANTES, o sea variables que no cambian, se definen con nombres en mayúsculas.



**El ciclo While**



Mi código While:

def run():

    LIMITE = 1000000

    contador = 0

    potencia\_2 = 2\*\*contador

    while potencia\_2 < LIMITE:

        print(" 2 Elevado a la " + str(contador) +

         " Es igual a: " + str(potencia\_2))

        contador = contador + 1

        potencia\_2 = 2\*\*contador

if \_\_name\_\_ == '\_\_main\_\_':

    run()

Consejo: se puede cancelar un ciclo infinito si presionas Ctrl + c.

contador += 1

Es igual a Contador = contador + 1.

**CICLO FOR:**

Su sintaxis es:  
variable = variable + constante(al incrementar)  
variable = variable - constante(al decrementar)

o de manera resumida:

variable += constante  
variable -= constante

Mi código:

Tabla del 11:

for i in range(10):

    result= 11\*i

    print("11 por " + str(i) + " es igual a= " + str(result))

MI CODIGO DE TABLAS DE MULTIPLICAR:

def run():

    for contador in range(5, 11):

        resultado = int(tabla) \* int(contador)

        print(str(tabla) + " multiplicado por " + str(contador) + " = " + str(resultado))

if \_\_name\_\_ == '\_\_main\_\_':

    tabla = input(

        """\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Bienvenid@ al programa de Tablas!!!

Cual tabla quieres saber?

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

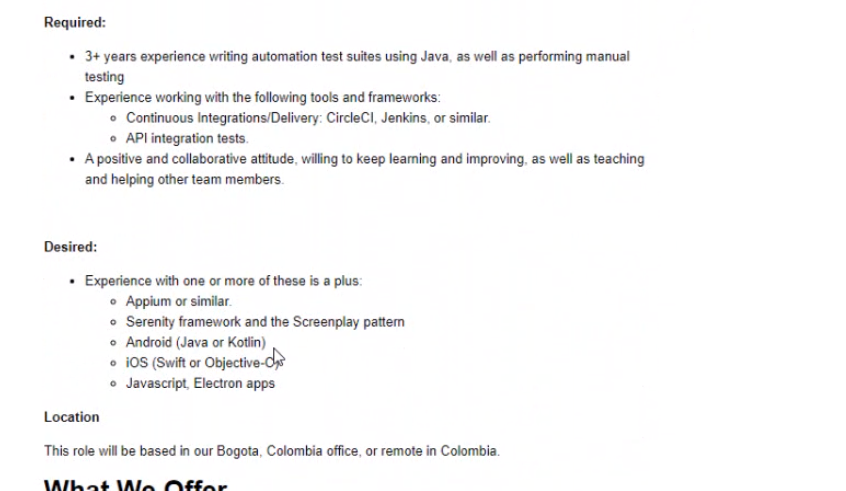
Tu respuesta= """)

    run()

**\*\*\*Hoja de tema diferente al curso\*\*\***

Buscar perfiles en las ofertas de trabajo en linkedin.

Consultorías sobre lo que buscan las empresas (Que Skills buscan)





Skills que necesitaría:

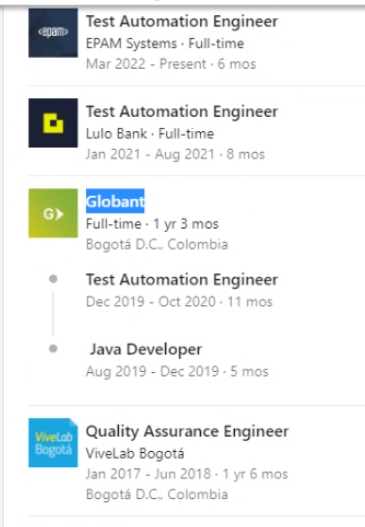
\*Perfill Java.

\*“Diseño de producto” en Platzi.

\*“Programar en Android”.

\*“Base de Datos Mysyqual y php”.

Yo comenzaría comenzando por 6 u 8 meses en una empresa pequeña, y después postularse en una empresa como Globant.



**\*\*\*Continuamos con el Curso\*\*\***

**Como recorrer un String con FOR.**

Recorrer **un string con el ciclo For** es tomar la cadena de caracteres y separarlas una a una. De este modo, quedaría el comando:

def run():

    frase = input("Escribe una frase: ")

    for caracter in frase:

        print(caracter.upper())

if \_\_name\_\_ == '\_\_main\_\_':

    run()

**Interrumpiendo ciclos con break y continue**

La **instrucción continue en Python** devuelve el control al comienzo del ciclo while o ciclo for. Esta instrucción rechaza todas las declaraciones restantes en la iteración actual del ciclo y mueve el control de regreso a la parte superior del mismo.

La instrucción break en Python termina el ciclo actual y reanuda la ejecución en la siguiente instrucción. En otras palabras, break rompe el ciclo entero mientras que continue solo rompe la vuelta actual.

def run():

    #for contador in range(1000):

    #    if contador % 2 != 0:

    #        continue

    #    print(contador)

    #for i in range(10000):

    #    print(i)

    #    if i == 5678:

    #        fin()

    #        break

#def fin():

#    print("Hemos llegado al final")

#EN TEXTO TAMBIEN FUNCIONA

    texto = input("Escribe un texto: ")

    for letra in texto:

        if letra == 'o':

            break

        print(letra)

if \_\_name\_\_ == '\_\_main\_\_':

    run()

**Cómo averiguar si un número es primo en Python #PROYECTO:**

En este ejemplo, aprenderemos **cómo detectar si un número es primo en Python**. Esto se conoce como prueba de primalidad. Un número primo es todo número que puede dividirse únicamente por sí mismo y por 1. Todos los números primos, excepto el 2, son impares.

En la matemática aplicada, los números primos son utilizados para generar códigos criptográficos seguros. Esto se logra empleando los números primos de Mersenne (números muy grandes).

**Código programa:**

def es\_primo(numero):

    if numero == 1:

        return False

    else:

        contador = 0

    for i in range(1 , numero+1):

        if i == 1 or i == numero:

            continue

        if numero % i == 0:

            contador += 1

    if contador == 0:

        return True

    else:

        return False

def run():

    numero = int(input("Escribe un número: "))

    if es\_primo(numero):

        print(str(numero) + " es primo")

    else:

        print(str(numero) +  " NO es primo")

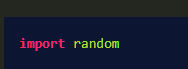
if \_\_name\_\_ == "\_\_main\_\_":

    run()

**Vamos a programar un mini juego de adivinanza.**

Vamos a programar un juego

Para el siguiente ejemplo, utilizaremos módulos en Python para crear la funcionalidad de nuestro código. Un módulo es un archivo con funciones ya predefinidas, que tenemos disponibles para ejecutarlas. Para traer o invocar un módulo, debemos escribir lo siguiente:



En este caso, “importamos” la función random, que trae un conjunto de funciones que nos permiten trabajar con la aleatoridad.

Gracias al punto ., accedemos a las funciones que trae el módulo. En este caso, generamos una variable y le asignamos random.randint(1, 100), lo cual nos genera un número aleatorio entero desde un número hasta otro (en este caso del 1 al 100).

Mi Codigo:

import random

def run():

    numero\_aleatorio = random.randint(1, 100)

    numero\_elegido = int (input("Elige un número del 1 al 100: "))

    while numero\_elegido != numero\_aleatorio:

        if numero\_elegido < numero\_aleatorio:

            print("Busca un número más grande")

        else:

            print("Busca un número más pequeño")

        numero\_elegido = int(input("Elige otro número: "))

    print ("Ganaste!!")

if \_\_name\_\_ == '\_\_main\_\_' :

    run()

Mi CODIGO MEJORADO CON VIDAS:

from operator import truediv

import random

def compara(numero\_aleatorio, numero\_elegido):

  if numero\_elegido < numero\_aleatorio:

      print("Busca un número más grande!!!")

      return False

  if numero\_elegido > numero\_aleatorio:

      print("Busca un número más pequeño!!!")

      return False

  if numero\_aleatorio == numero\_elegido:

      print("GANASTE!!!")

      return True

def run():

    vidas = 5

    numero\_aleatorio = random.randint(1, 10)

    numero\_elegido = int (input("""\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

    Elige un número del 1 al 10:

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"""))

    while vidas >0 and numero\_aleatorio != numero\_elegido:

      compara(numero\_aleatorio, numero\_elegido)

      print("Tienes: " + str(vidas) + " Vidas")

      numero\_elegido = int(input("Elige otro número: "))

      vidas -= 1

    if compara(numero\_aleatorio, numero\_elegido) == False:

      print("Perdiste Totalmente")

    else:

      print("""\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

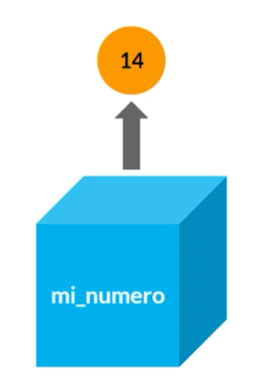
\*\*\*Lo lograste\*\*\*

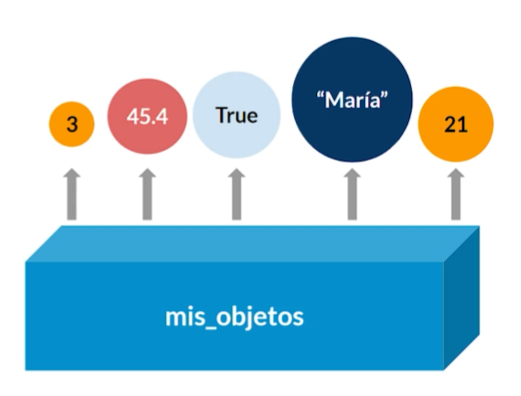
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*""")

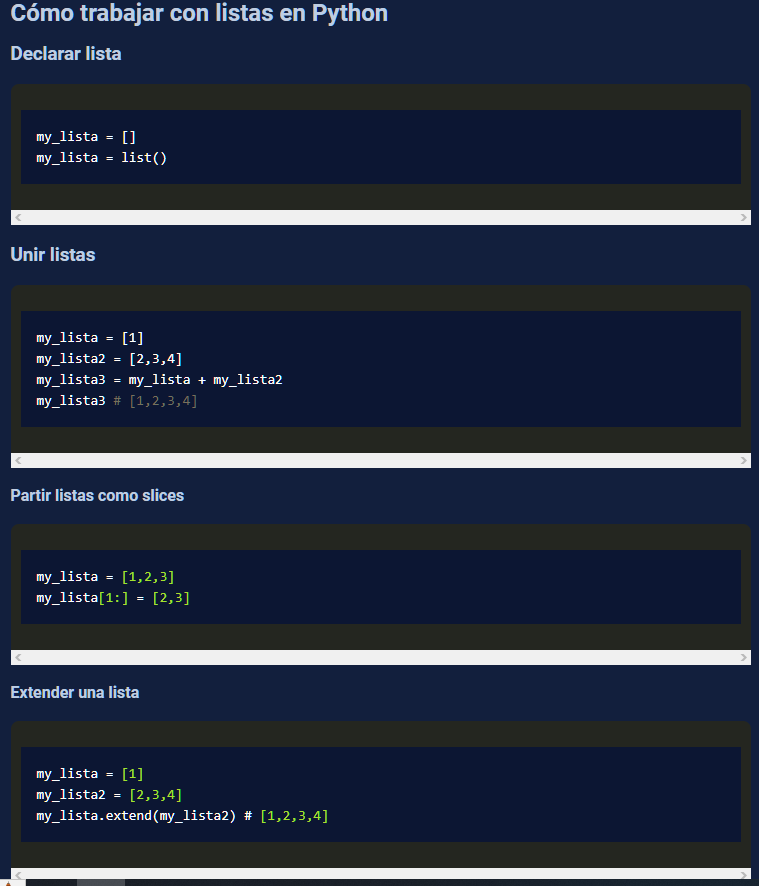
if \_\_name\_\_ == '\_\_main\_\_' :

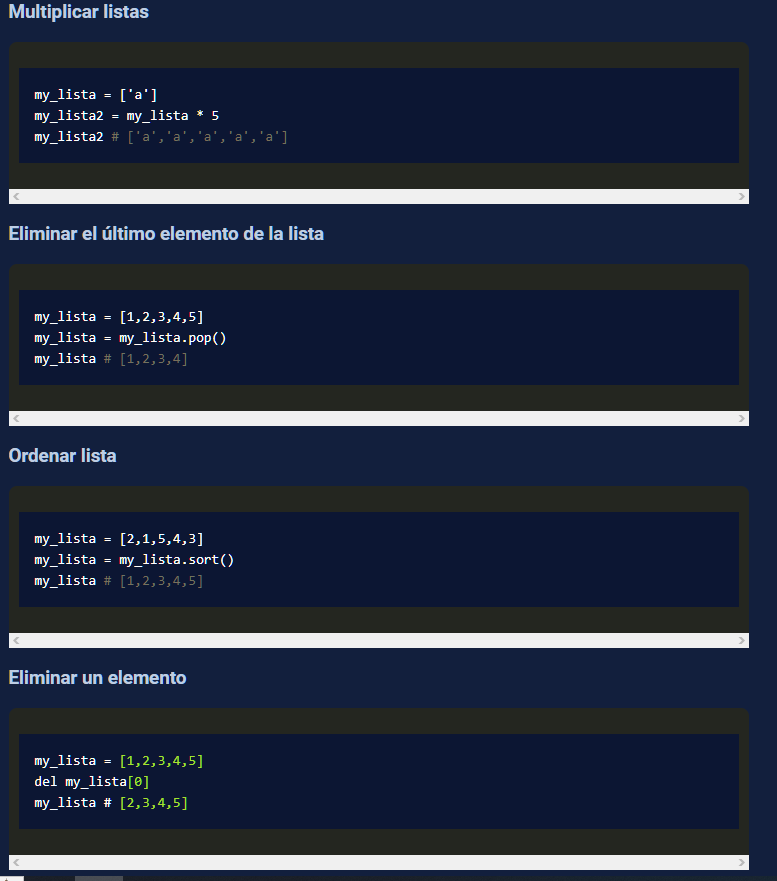
    run()

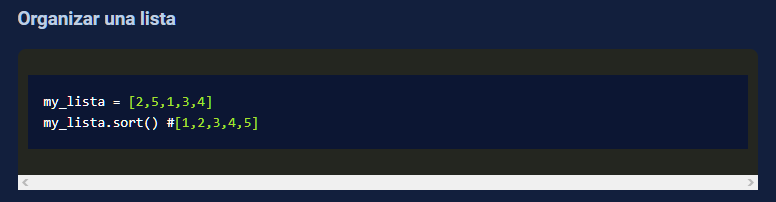
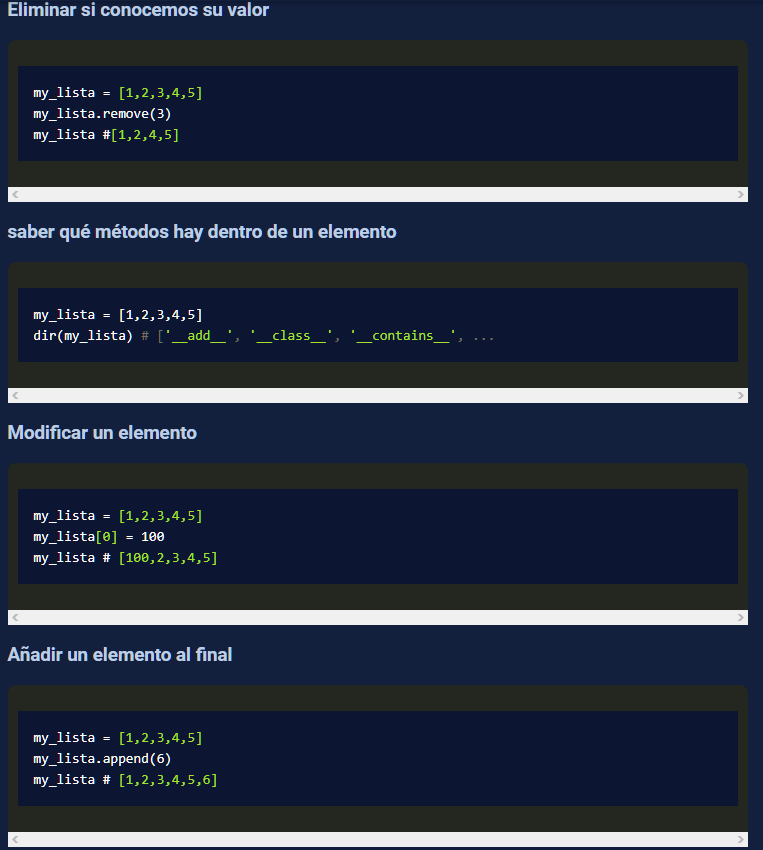
**Variables: Listas**

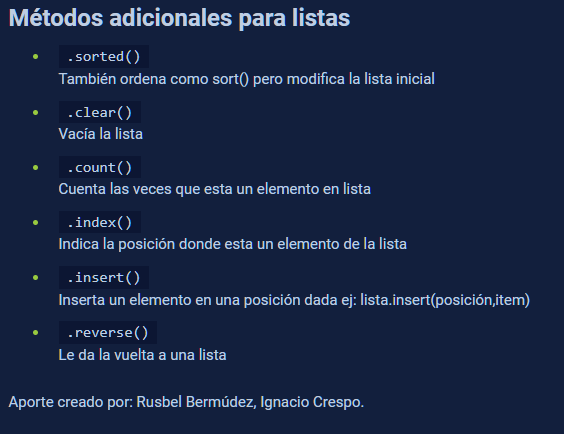
Las **listas** nos permiten guardar múltiples valores en una sola variable. Estas listas en Python nos permiten guardar elementos del mismo tipo o diferentes, por lo que podemos tener strings, números enteros y decimales juntos en una misma variable. Las listas también son conocidas como arrays en otros lenguajes programación. 

****

****

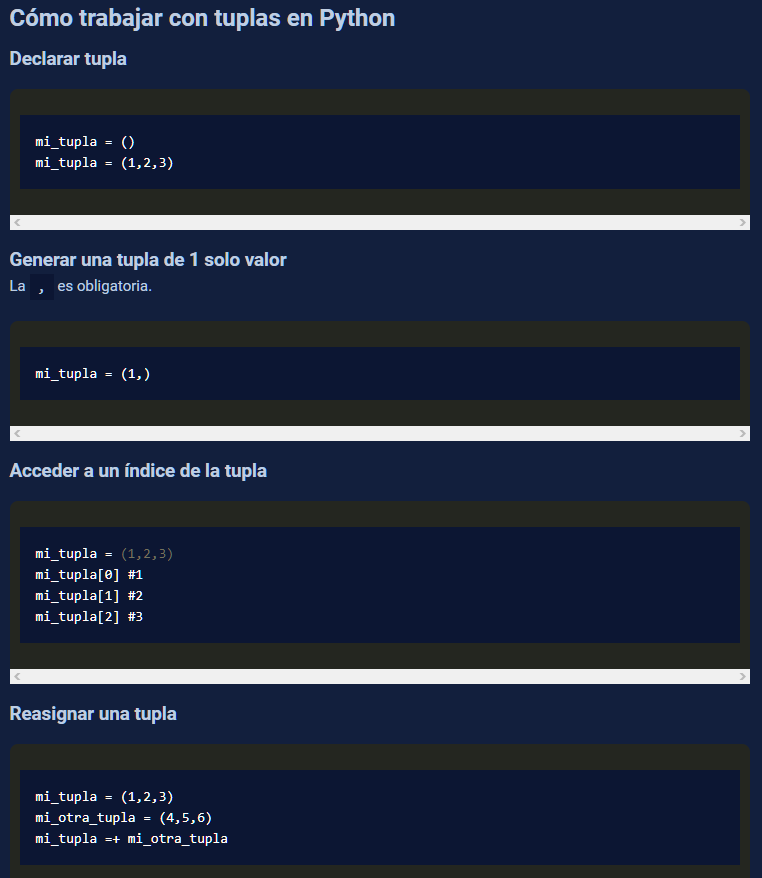
****

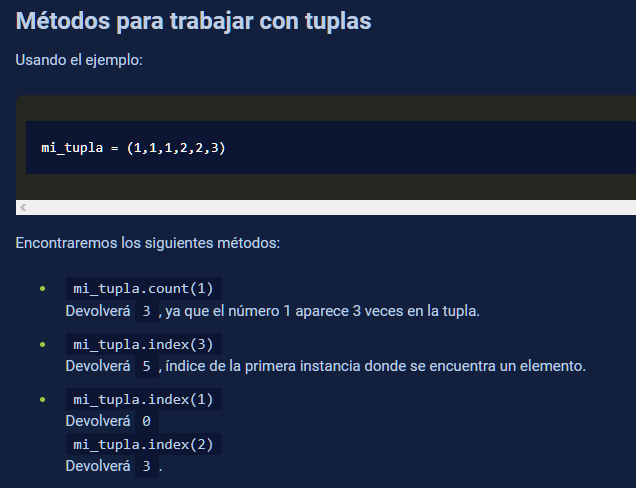
****

****

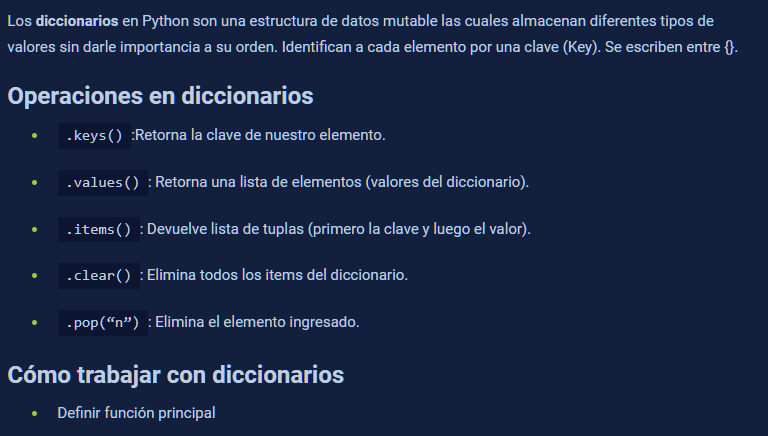
**TUPLAS**

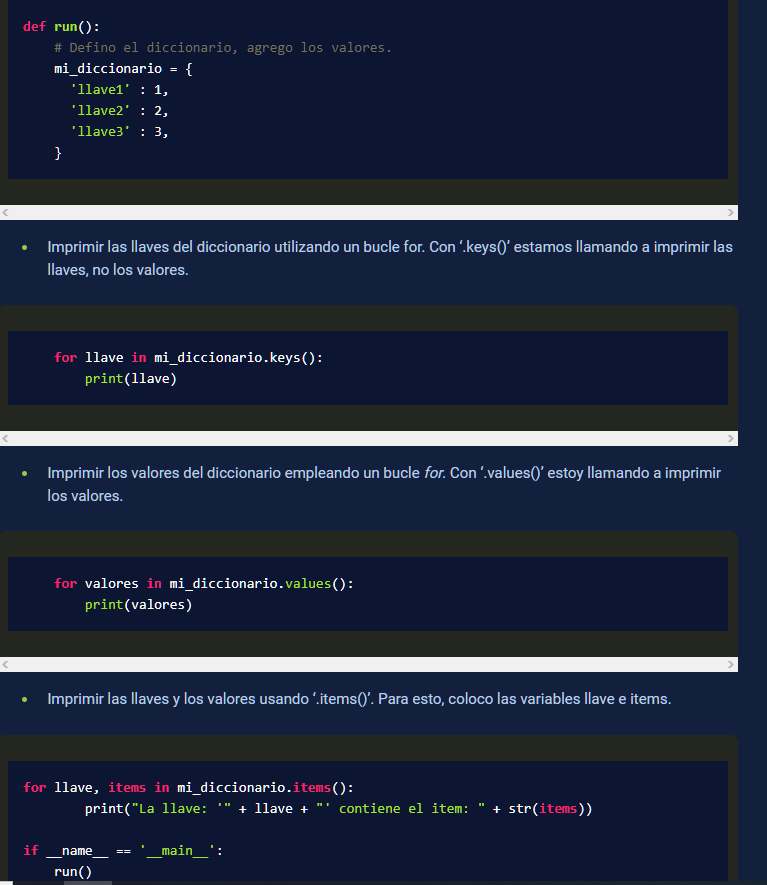
Las **tuplas** son estructuras de datos inmutables. Es decir, no puedes modificar una tupla a agregando o quitando un valor. Lo único que puedes hacer es definir de nuevo esa tupla a. Los objetos inmutables (como los strings) son tipos de datos para Python que apuntan a un lugar específico en memoria y que su contenido no puede ser cambiado. Al cambiar el contenido predefiniendo el contenido de la variable a, entonces cambiará su posición en memoria.

****

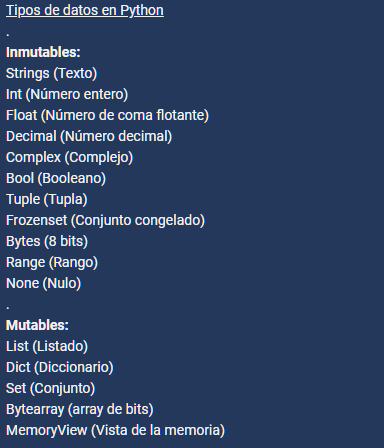
****

**Diccionarios**

****

****

**Datos mutables e inmutables en Python:**

****

**Mi proyecto (Generador de contraseña):**

**Mi código:**

import random

def generar\_contrasena():

    mayusc = ['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'Ñ', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'X', 'Y', 'Z']

    minusc = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'ñ', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'x', 'y', 'z']

    numeros = ['1', '2', '3', '4', '5', '6', '7', '8', '9', '0']

    simbolos = ['\*', '+', '-', '/', '@', '\_', '?', '!', '[', '{', '(', ')', '}', ']', ',', ';', '.', '>', '<', '~', '°', '^', '&', '$', '#', '"']

    caracteres = mayusc + minusc + simbolos + numeros

    contrasena = []

    for i in range(15):

        caracter\_random = random.choice(caracteres)

        contrasena.append(caracter\_random)

    contrasena = ''.join(contrasena)

    return contrasena

def run():

    contra = generar\_contrasena()

    print("Tu nueva contraseña es: " + contra)

if \_\_name\_\_ == '\_\_main\_\_':

    run()

Y HEMOS TERMINADO EL CURSO #1 de Python de Platzi