**BITÁCORA PYTHON**

**TERCER CORTE**

**ANALISIS Y DISEÑO DE ALGORITMOS**

**ARIEL MEJIA SUAREZ**

**FACULTAD DE INGERIERIA**

**INGENIERIA DE SISTEMAS**

**UNIVERSIDAD DE SAN BUENAVENTURA**

**BOGOTA DC**

**2020**

Contenido

[**HISTORIA E INTRODUCCIÓN A PYTHON** 4](#_Toc40486050)

[**INSTALACIÓN DE PYTHON** 4](#_Toc40486051)

[**VIDEO 1 - SINTAXIS DE PYTHON I** 7](#_Toc40486052)

[**VIDEO 2 - USO DE PYTHON POR MEDIO DE SUBLIME TEXT** 8](#_Toc40486053)

[**VIDEO 3 - OPERADORES EN PYTHON** 9](#_Toc40486054)

[**VIDEO 4 - SINTAXIS BÁSICA II** 10](#_Toc40486055)

[**VIDEO 5 - SINTAXIS BÁSICA DE PYTHON FUNCIONES III** 11](#_Toc40486056)

[**VIDEO 6 - SINTAXIS LISTAS** 13](#_Toc40486057)

[**VIDEO 8 TUPLAS** 14](#_Toc40486058)

[**VIDEO 9 - DICCIONARIOS** 15](#_Toc40486059)

[**VIDEO 10 – CONDICIONALES 1** 17](#_Toc40486060)

[**VIDEO 11- CONDICIONALES 2** 18](#_Toc40486061)

[**VIDEO 12 – SWITCH CASE** 19](#_Toc40486062)

[**VIDEO 13 – OPERADORES LÓGICOS** 20](#_Toc40486063)

[**VIDEO 14 – BUCLES 1** 22](#_Toc40486064)

[**VIDEO 15 – BUCLES 2** 23](#_Toc40486065)

[**VIDEO 16 – BUCLES 3** 24](#_Toc40486066)

[**VIDEO 17 – WHILE** 25](#_Toc40486067)

[**VIDEO 18 – CONTINUE, PASS Y ELSE** 26](#_Toc40486068)

[**VIDEO 19 – GENERADORES 1** 27](#_Toc40486069)

[**Video 20 – GENERADORES 2** 28](#_Toc40486070)

[**VIDEO 21 – EXCEPCIONES** 29](#_Toc40486071)

[**VIDEO 22 – EXCEPCIONES** 30](#_Toc40486072)

[**VIDEO 23 – EXCEPCIONES 3** 31](#_Toc40486073)

[**VIDEO 24 – PROGRAMACIÓN ORIENTADO A OBJETOS (POO)** 32](#_Toc40486074)

[**Video 25 - PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 2** 32](#_Toc40486075)

[**Video 26 - PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 3** 33](#_Toc40486076)

[**VIDEO 27 – PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 4** 34](#_Toc40486077)

[**VIDEO 28 – PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 5** 35](#_Toc40486078)

[**VIDEO 29 – HERENCIA 1** 36](#_Toc40486079)

[**VIDEO 30 – HERENCIA 2** 37](#_Toc40486080)

[**VIDEO 31 – HERENCIA 3** 38](#_Toc40486081)

[**VIDEO 32 – POLIMORFISMO** 39](#_Toc40486082)

[**VIDEO 33 – METODOS DE CADENAS** 40](#_Toc40486083)

[**VIDEO 34 – MODULOS** 41](#_Toc40486084)

[**VIDEO 35 – PAQUETES** 42](#_Toc40486085)

[**VIDEO 36 – PAQUETES DISTRIBUIBLES** 43](#_Toc40486086)

[**VIDEO 37 – ARCHIVOS EXTRENOS 1** 44](#_Toc40486087)

[**VIDEO 38 – ARCHIVOS EXTRENOS 2** 44](#_Toc40486088)

[**VIDEO 39 – SERIALIZACION** 45](#_Toc40486089)

[**VIDEO 40 – SERIALIZACION 2** 45](#_Toc40486090)

[**VIDEO 42 – INTERFASES GRAFICAS 1** 47](#_Toc40486091)

[**VIDEO 43 – INTERFASES GRAFICAS 2** 48](#_Toc40486092)

[48](#_Toc40486093)

[**VIDEO 44 – INTERFASES GRAFICAS 3** 49](#_Toc40486094)

[**VIDEO 45 – INTERFASES GRAFICAS 4** 50](#_Toc40486095)

[**VIDEO 46 – INTERFASES GRAFICAS 5** 51](#_Toc40486096)

[**VIDEO 47 – INTERFASES GRAFICAS 7** 53](#_Toc40486097)

[**VIDEO 48 – INTERFASES GRAFICAS 8** 54](#_Toc40486098)

[**VIDEO 49 – INTERFASES GRAFICAS 9** 55](#_Toc40486099)

[**VIDEO 50 – RADIOBOTON** 56](#_Toc40486100)

[**REFERENCIAS:** 57](#_Toc40486101)

# 

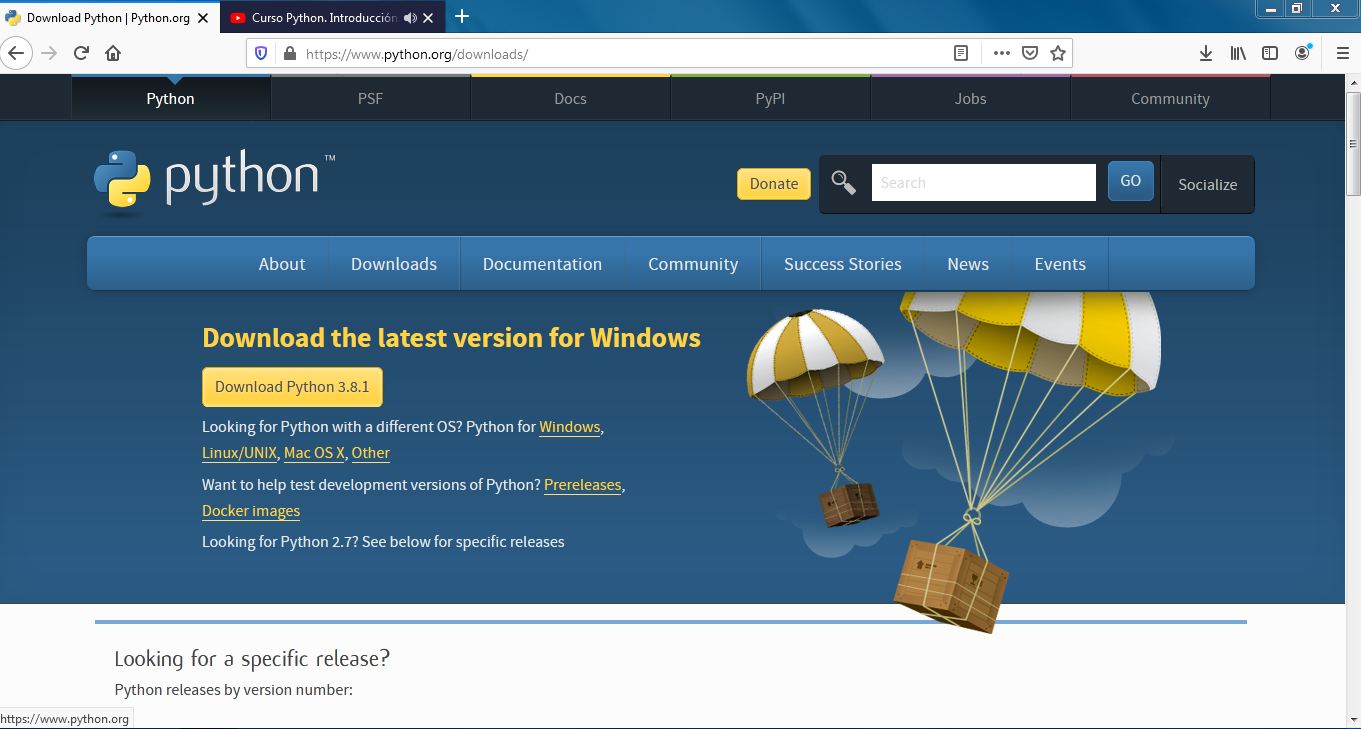
# **HISTORIA E INTRODUCCIÓN A PYTHON**

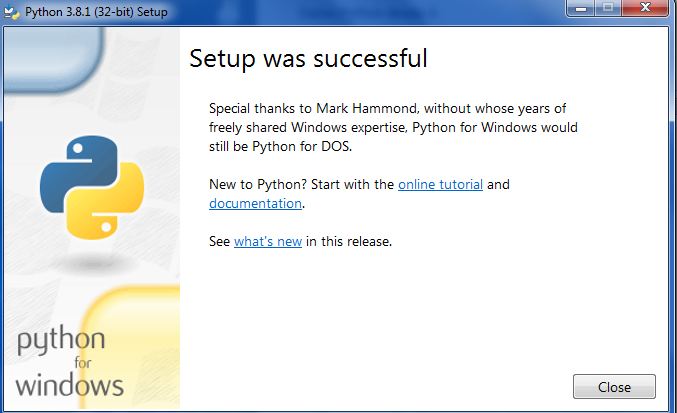
* Creado por Guido Van Rossum a comienzos de los 90.

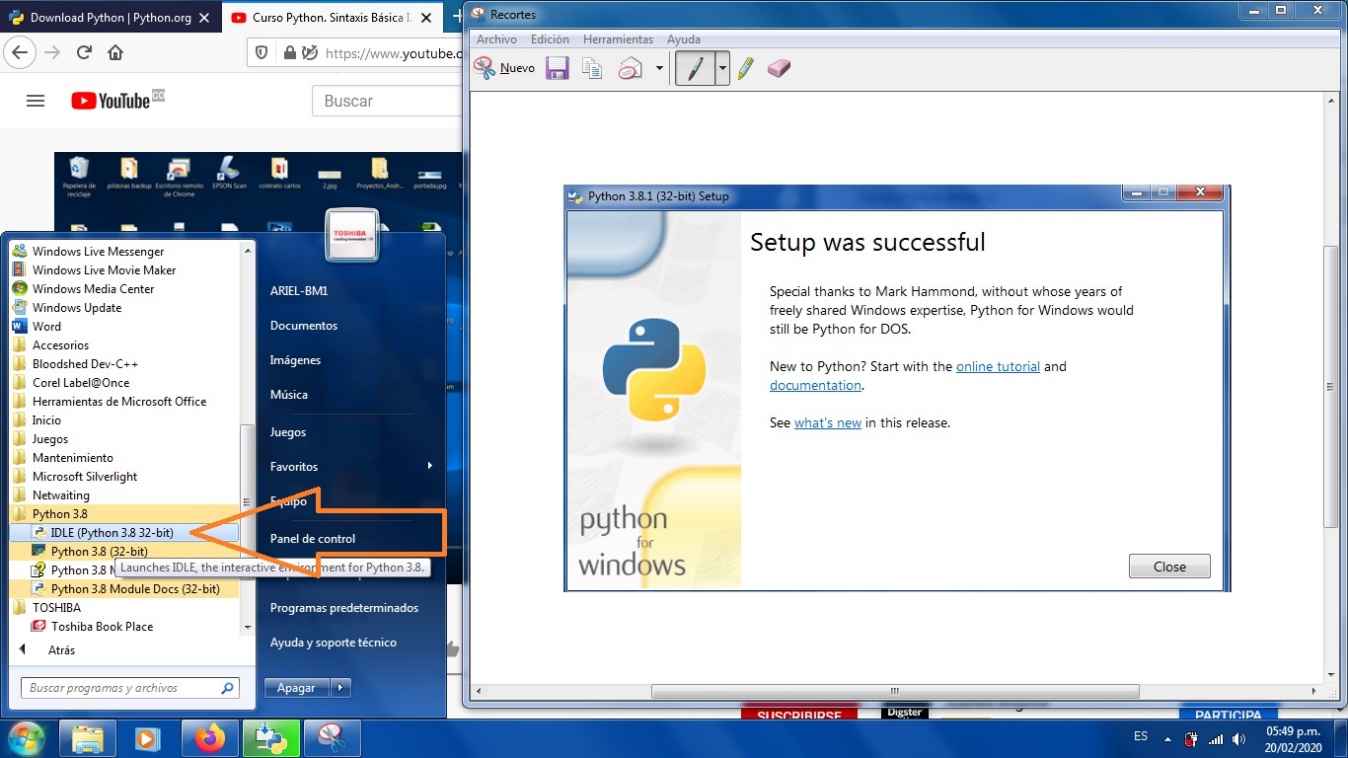
Es un lenguaje de altísimo nivel es decir de gramática sencilla y muy sencilla.

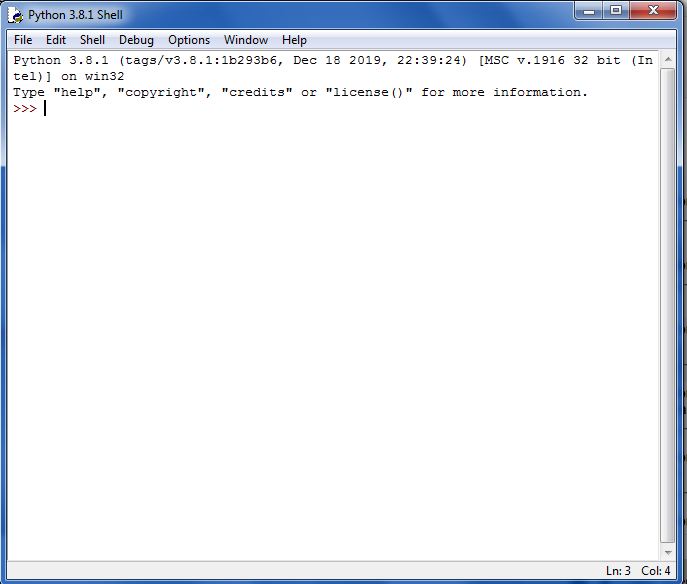
* Es orientado a objetos.
* Es código abierto.
* Fácil de aprender.
* No necesita de punto y coma (;).
* Es versátil, muy usado para app de escritorios, app web, app, servidor, app web.
* Sus declaraciones de variables son notable mente diferentes declarar.
* Cuenta con herencia múltiple.
* Es una herramienta muy sencilla a la hora de iniciar en el mundo de programación
* Es multiplataforma podemos usarlo en varios S.O.

# **INSTALACIÓN DE PYTHON**

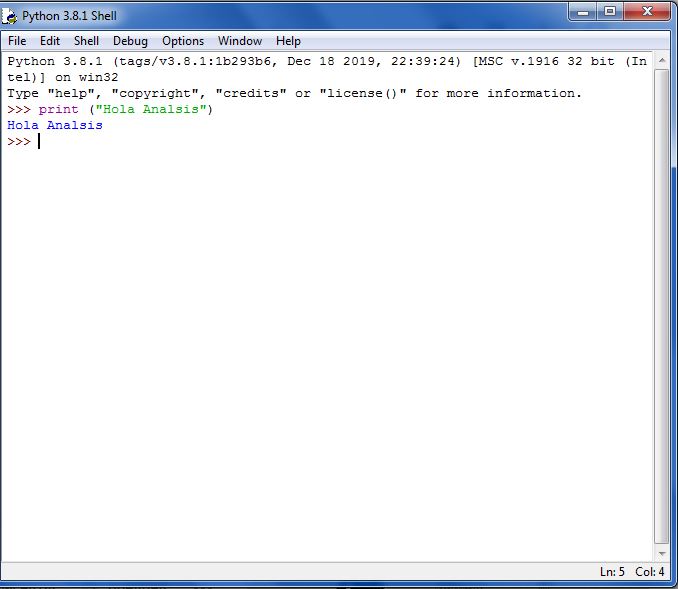
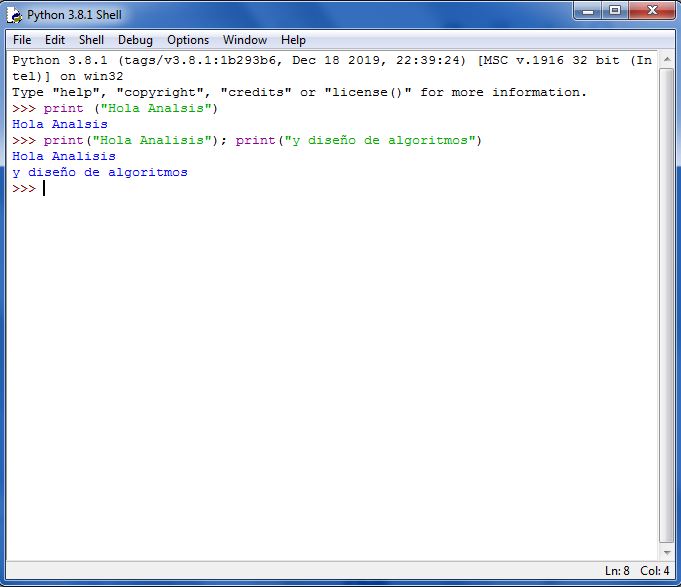
Nos dirigimos a la página oficial de Python [www.python.org](http://www.python.org) en la sección descargas:

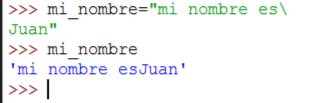




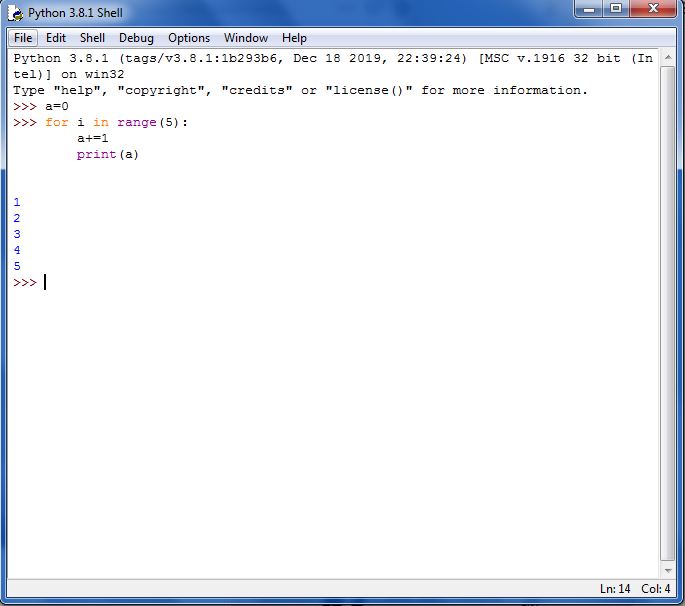


# **VIDEO 1 - SINTAXIS DE PYTHON I**

* Comando imprimir en Python:
* Comando para decalara variables:
* Comando para dar salto de linea:



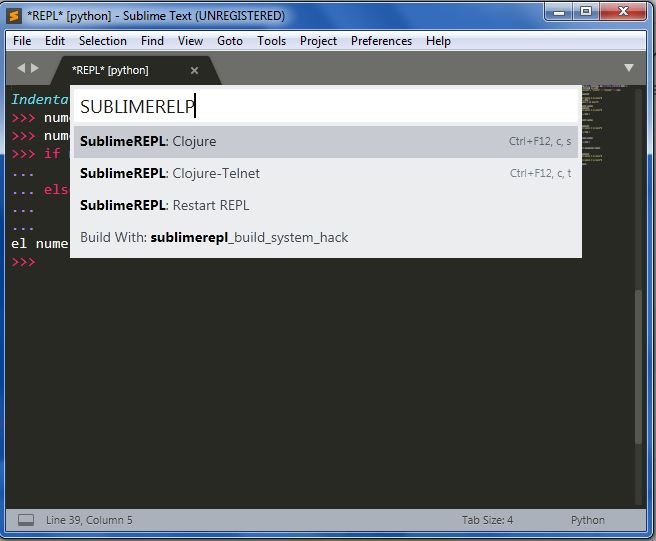
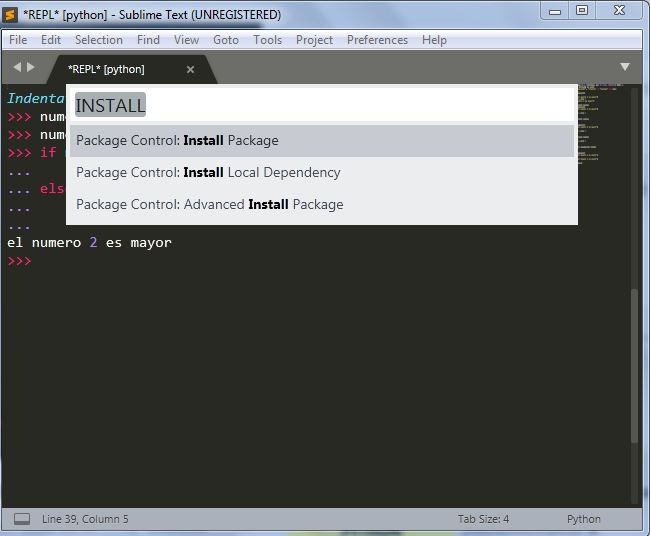
* Sintaxis para ciclo for:



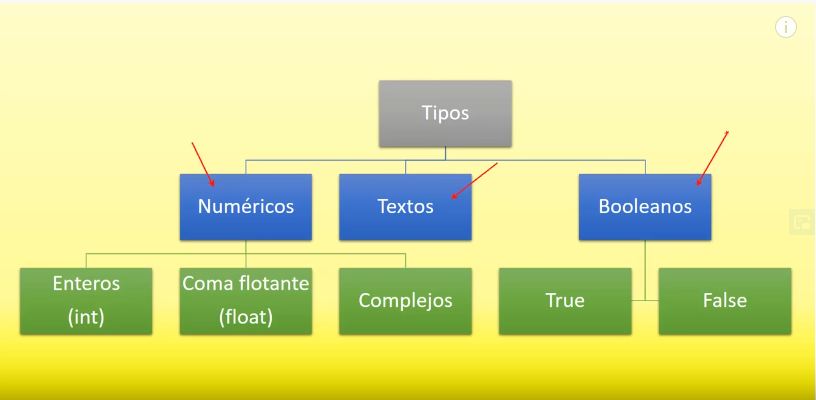
# **VIDEO 2 - USO DE PYTHON POR MEDIO DE SUBLIME TEXT**

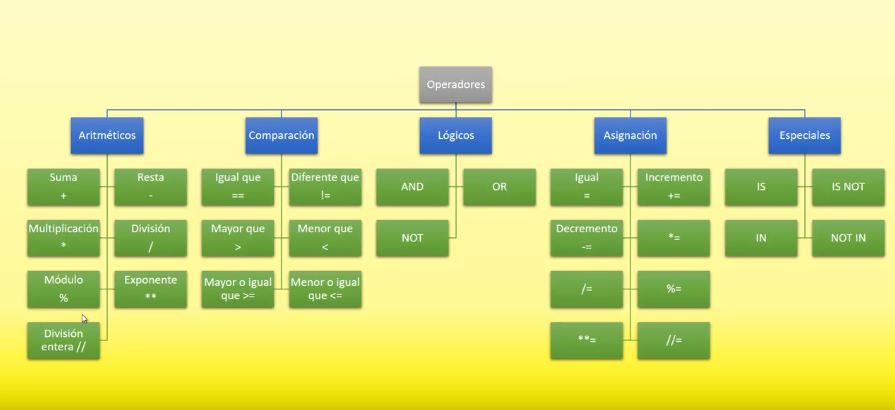
* Instalación de sublime text

Para instalar sublime text vamos <https://www.sublimetext.com/3> verificamos el tipo de S.O descargamos e instalamos, una vez instalamos vamos a descarga los paquetes que nos permitirán usar el lenguaje Python en sublime text:

* descargamos el paquete SUBLIMERELP solo tecleando y dando intro se descargará e instalará
* Luego descargamos el package control: Install package
* Una vez instalado los compleneto en sublime text ya podemos escribir en el lenguaje python en el editor de texto tal como los hacemos en el IDE de python.

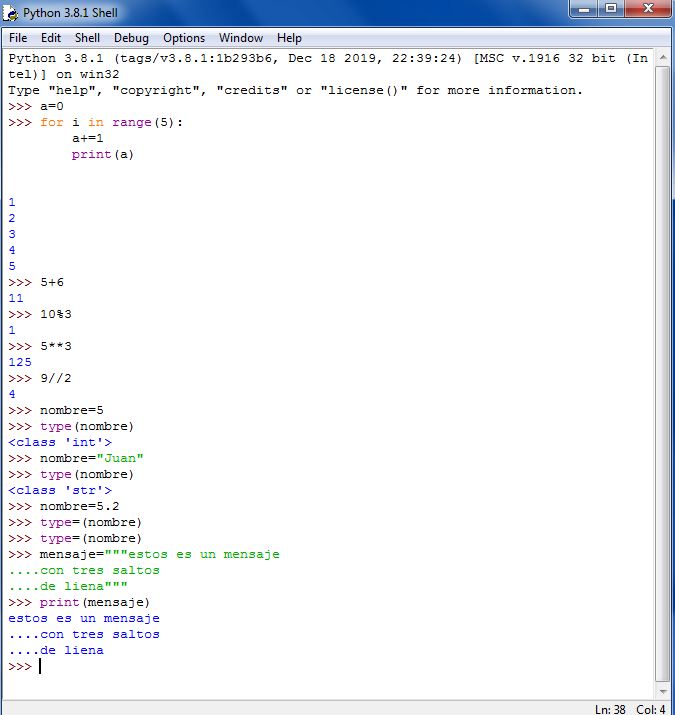
# **VIDEO 3 - OPERADORES EN PYTHON**

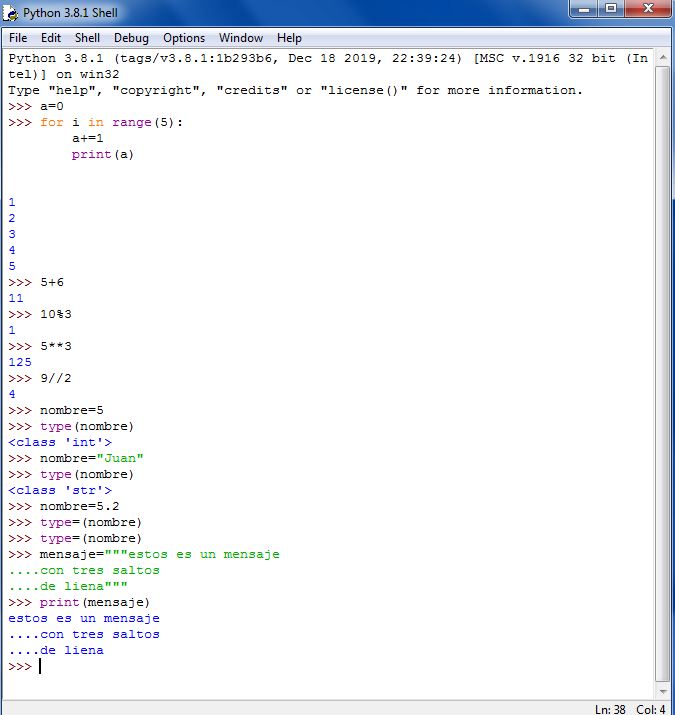
* Tipos de datos:
* Tipos de operadores:

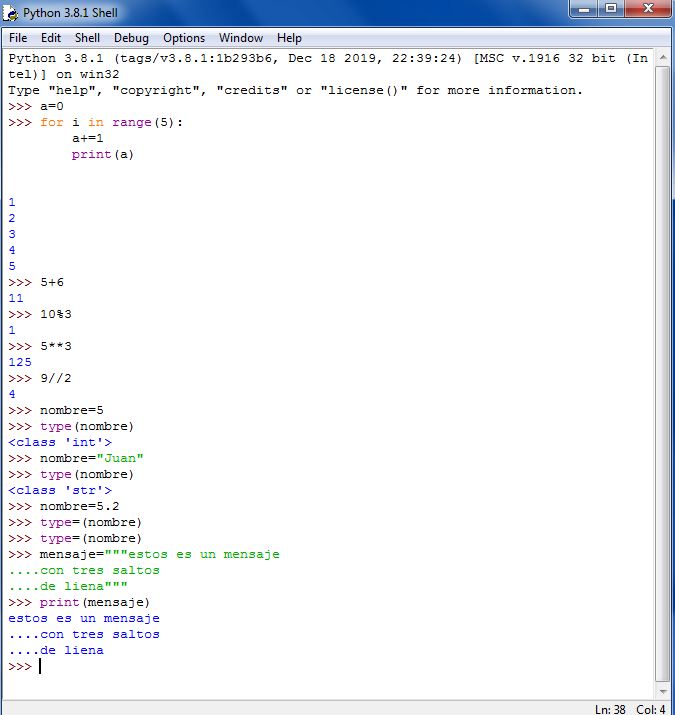


# **VIDEO 4 - SINTAXIS BÁSICA II**

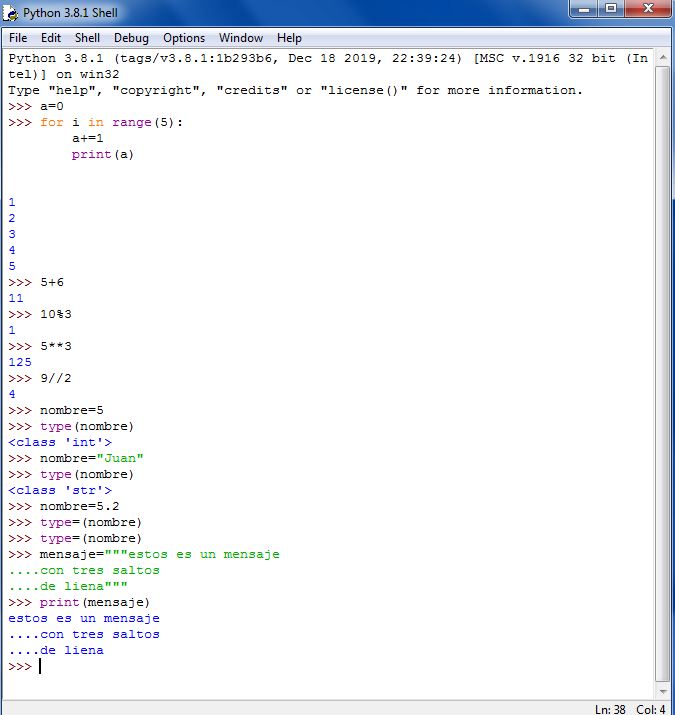
* Tipos de operadores:



* Detección de variables:



* Print con saltos de línea:



* Sintaxis if else



# **VIDEO 5 - SINTAXIS BÁSICA DE PYTHON FUNCIONES III**

La sintaxis en Python es muy sencilla tienen como características principales lo siguiente:

* Estas devuelven valores.
* Las funciones pueden tener parámetros y/o argumentos
* A las funciones también se les llaman métodos cuando se encuentran definidas en una clase

La función contiene tres partes:

* La declaración se identifica por la cabecera de la función **(def mensaje ():)**
* El cuerpo de la función se identifica con el proceso interno:

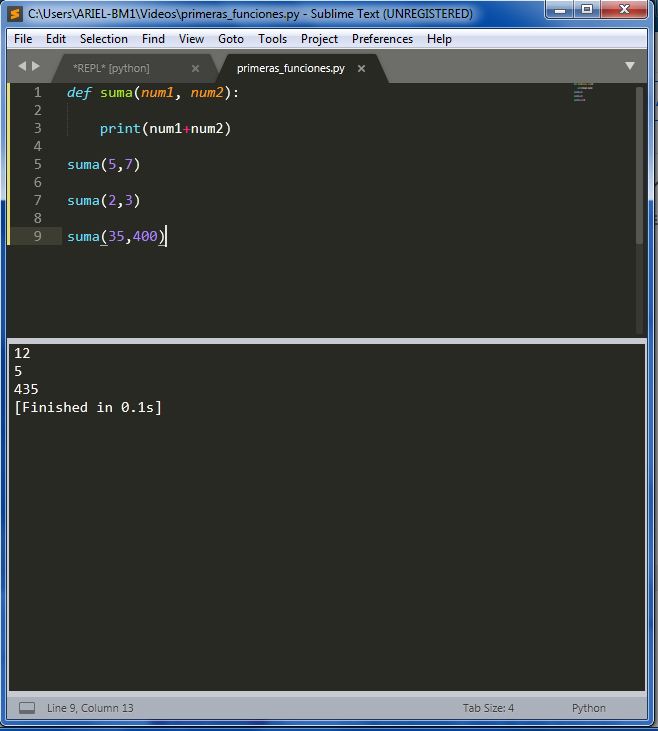
print("primera función en python")

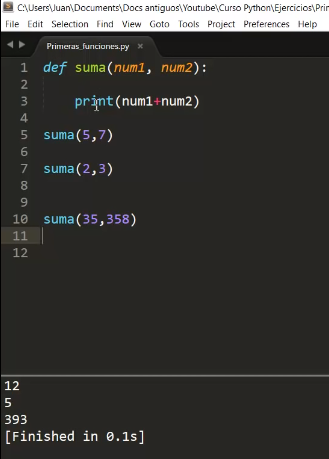
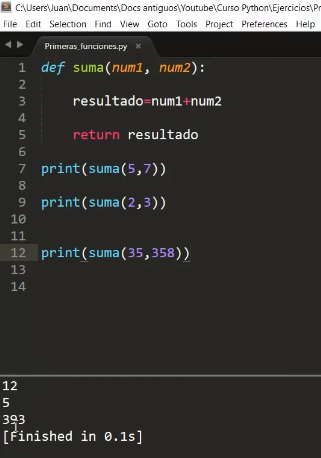
print("primeras instrucciones en python")

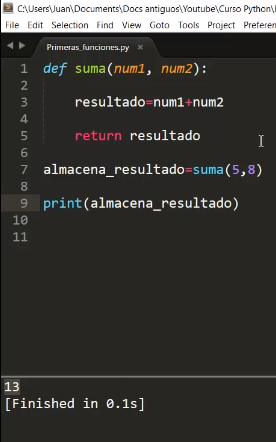
print("poco a poco iremos avanzando")

* Y la llamada de la función se identifica con **mensaje ()**

Ejemplo:



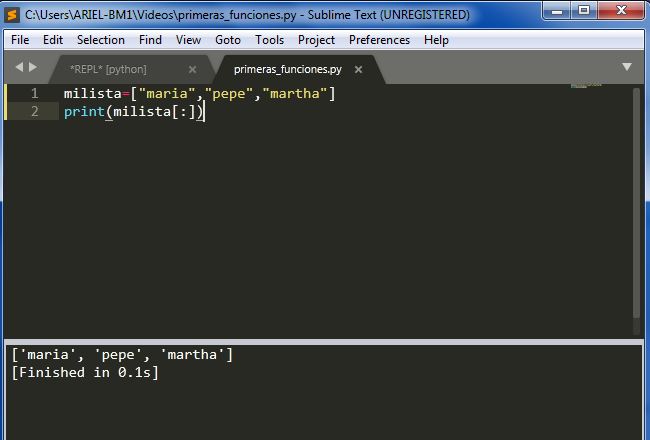
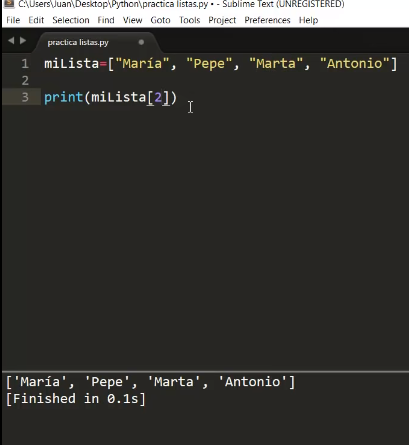


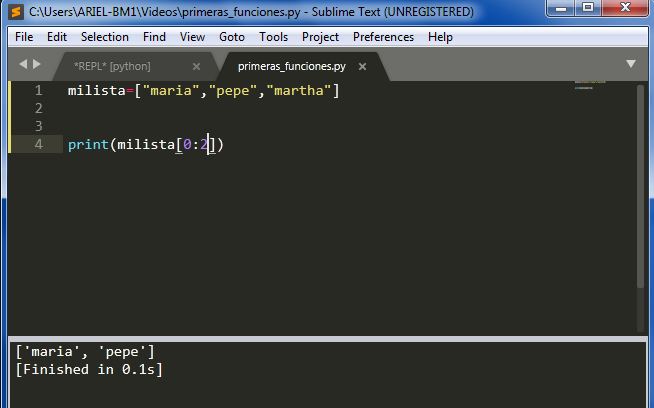


# **VIDEO 6 - SINTAXIS LISTAS**

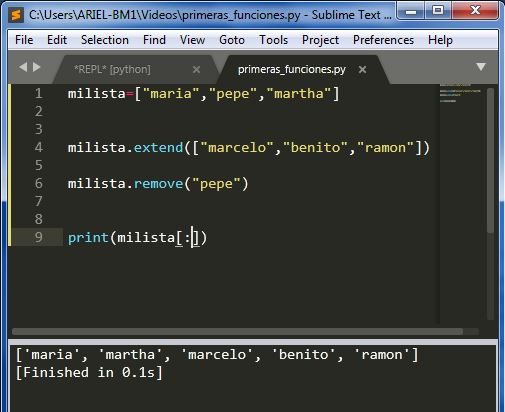
En el video 7 vamos a tratar de entender las listas como son y como fusionan en lenguaje Python como primera instancia es necesario conocer que las listas son los mismos arreglos (arrays)en otros lenguajes de programación tiene varias características:

* Las listas pueden guardar diferentes tipos de valores
* Se expanden dinámicamente añadiendo nuevos elementos





* Impresión de una lista por posición:
* Impresión excluyendo e incluyendo elementos de otras listas:

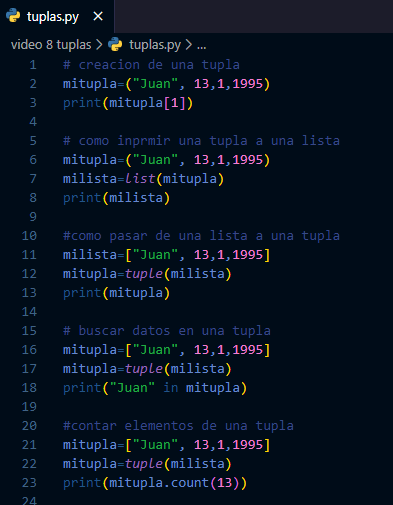


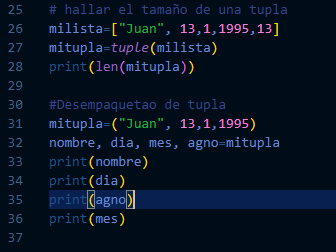
# **VIDEO 8 TUPLAS**

En el video 8 veremos las tuplas estas son listas inmutables es decir no se pueden modificar después de la creación ahí algunas condiciones que se deben cumplir:

* No se permiten añadir, eliminar, mover elementos (append, extend, remove)
* Si se puede extraer posiciones, pero el resultado, pero se está extracción es una tupla nueva
* No se pueden hacer búsquedas (no índex) en versiones anteriores a las 3.0
* Se verifica si un elemento se encuentra en la tupla

Ventajas:

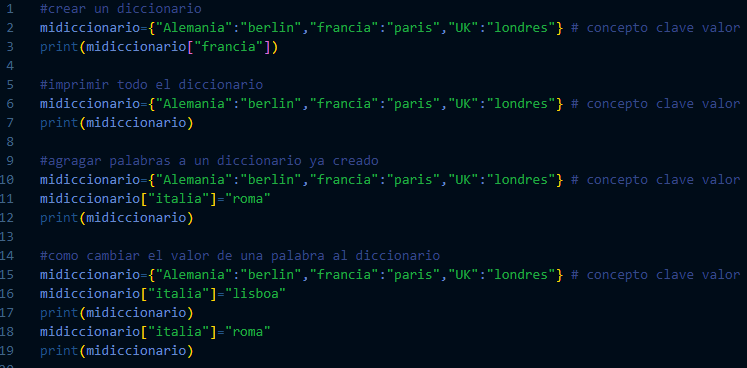
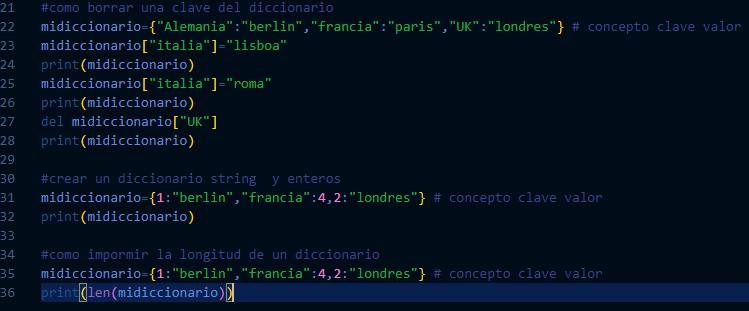
* Mas rápidas
* Menos espacio
* Mayor velocidad
* Formatean Strings
* Se pueden utilizarse como calves en un diccionario.
* Se diferencias de las listas pues las listas usan [] y las tuplas ()



# **VIDEO 9 - DICCIONARIOS**

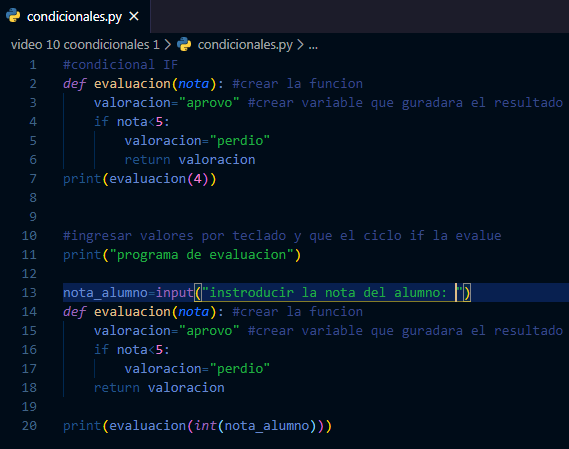
Los diccionarios son una estructura de datos que nos permite almacenar valores de diferentes tipos (enteros, cadenas de texto, decimales) e incluso listas y otros diccionarios.

Una de sus principales características es que los datos se almacenan siendo asociados a una clave de tal forma que se crea una asociación de tipo **clave: valor** para cada elemento almacenado.

Estos elementos que se almacenan no están ordenados al momento de guárdalos estos no tienen prioridad en el orden

# **VIDEO 10 – CONDICIONALES 1**

Condicional IF: sigue la lógica de otros lenguajes de programación donde ingresan una variable y las evalúa en el IF y luego imprime el resultado

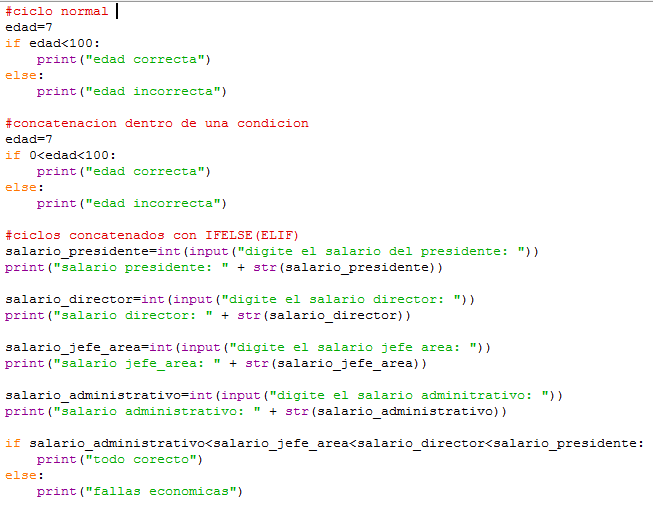


# **VIDEO 11- CONDICIONALES 2**

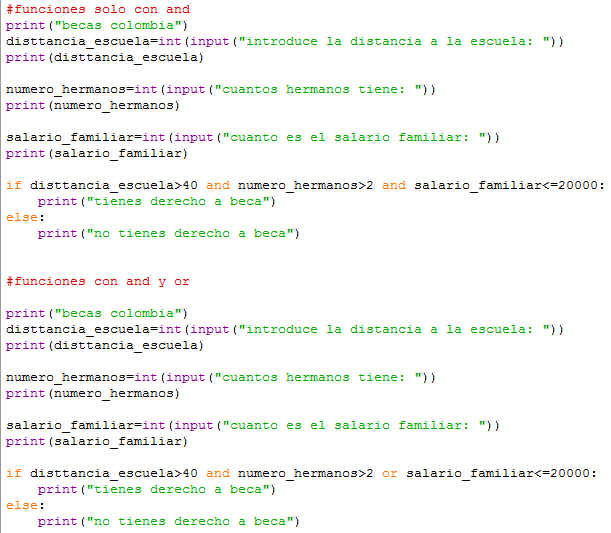
Construcción de ciclo IF:

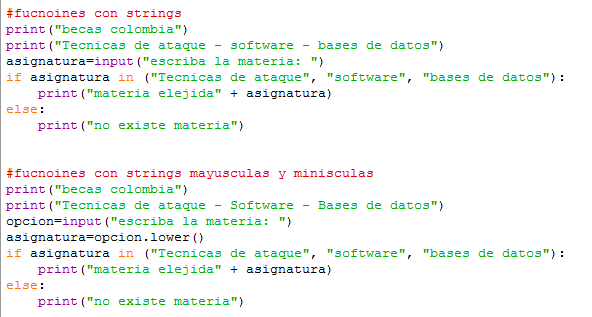
# **VIDEO 12 – SWITCH CASE**

Condicionales:

* Concatenación de operadores de comparación.
* Operadores lógicos “and”y “or”
* Operador “in”

# **VIDEO 13 – OPERADORES LÓGICOS**

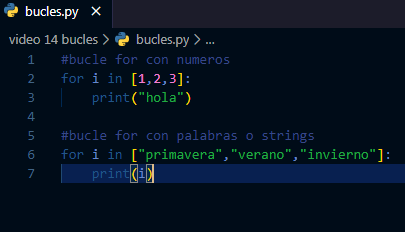
Operadores lógicos son muy útiles en la programación el operador lógico and se puede traducir como “y si además” y el operador lógico or se puede traducir como “o si no”.



# **VIDEO 14 – BUCLES 1**

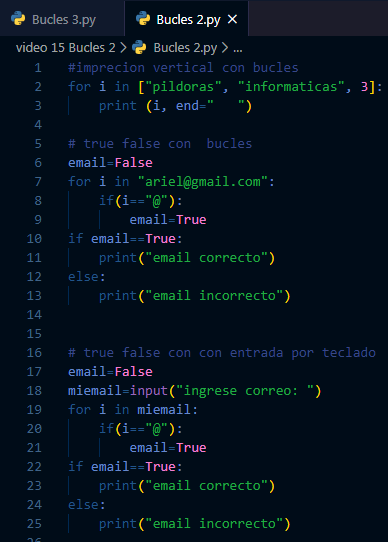
Tipos de bucles son usados para repetir código de ser necesario y son una estructura de control de flujo y los hay de dos tipos:

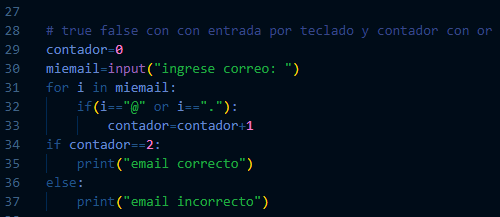
1. Determinados:
   1. Se ejecutan un número determinado de veces
   2. Se sabe a priori cuantas veces se va a ejecutar el código del interior del bucle
2. Indeterminados:
   1. Se ejecutan un nuero intermediando de veces
   2. No se sabe a prior cuantas veces se va a ejecutar el código del interior del bucle
   3. El numero de veces que se ejecutará dependerá de las circunstancias durante la ejecución del programa
3. Sintaxis de FOR
   1. Bucle for (determinado) ejemplo: **for variable in elemento a recorrer:** código a seguir



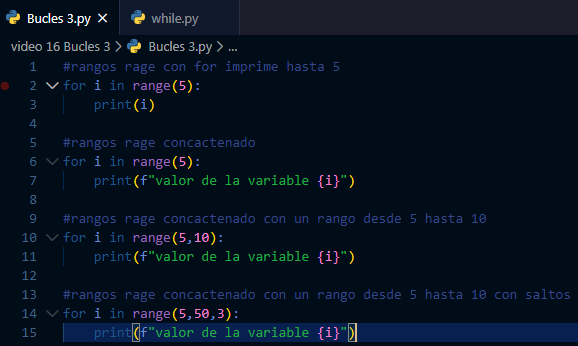
# **VIDEO 15 – BUCLES 2**

1. Bucle for:
   1. Recorrido Strings
   2. Tipo range o rango
   3. Notaciones especiales con print

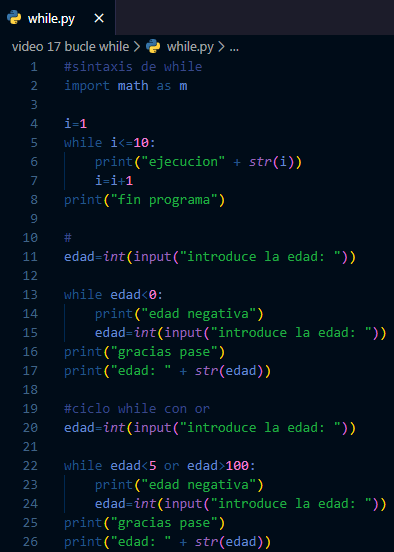


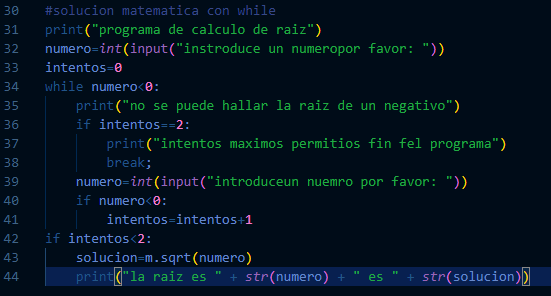


# **VIDEO 16 – BUCLES 3**

1. Bucle for
   1. Tipo range o rangos
   2. Notaciones especiales con print

# **VIDEO 17 – WHILE**

1. Bucle while es indeterminado
   1. Sintaxis



# **VIDEO 18 – CONTINUE, PASS Y ELSE**

# **VIDEO 19 – GENERADORES 1**

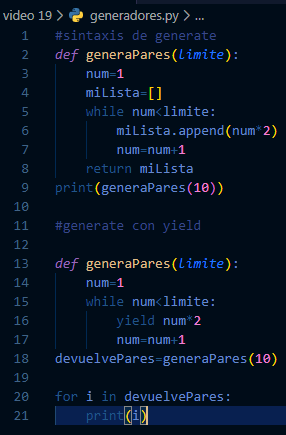
Son estructuras que extraen valores de una función y se almacenan en objetos iterables (que se pueden recorrer)

Estos valores se almacenan de uno en uno

Cada vez que un generador almacena un valor esta permaneces en un estado pausado hasta que se solicita el siguiente. Esta característica es conocida como “suspensión de estado”.

Funcionamiento:

1. Tradicional;
2. Generador:
3. Son más eficientes que las funciones tradicionales s
4. Muy útiles con listas e valores infinitos
5. Bajo determinados escenarios será más útil que un generador devuelva los valores de uno en uno

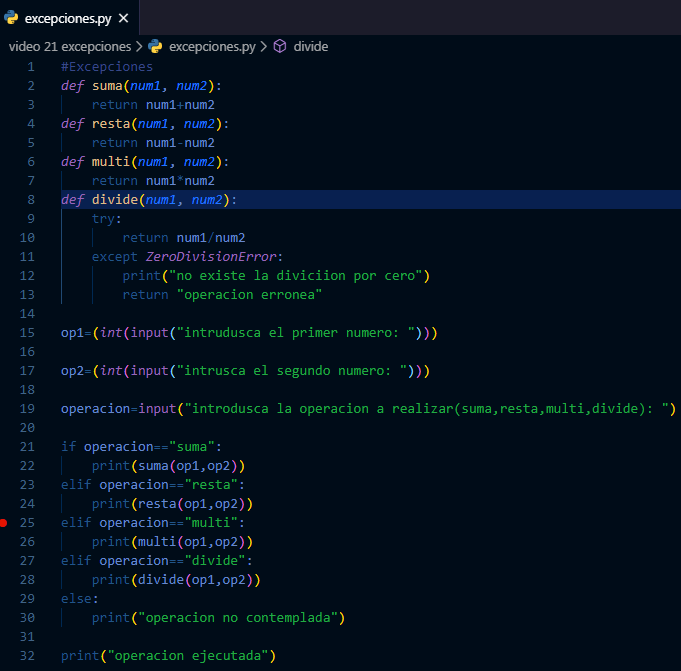


# **Video 20 – GENERADORES 2**

Generadores

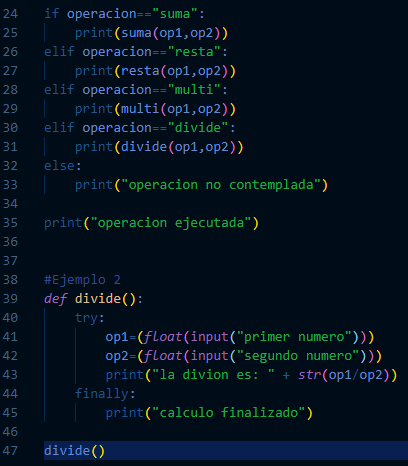
1. Nueva instrucción yield from comúnmente dentro de un generador con el fin de simplificar el código del generador si se usan bucles anidados como ejemplo un for dentro de otro for
2. El generador nos devuelve un objeto generar donde se alancenan elementos de uno por unos puede ser números, Strings, diccionarios, tuplas, etc.

# **VIDEO 21 – EXCEPCIONES**

Son errores que ocurren durante la ejecución el programa. La sintaxis del código es correcta, pero durante la ejecución ha ocurrido “algo inesperado”

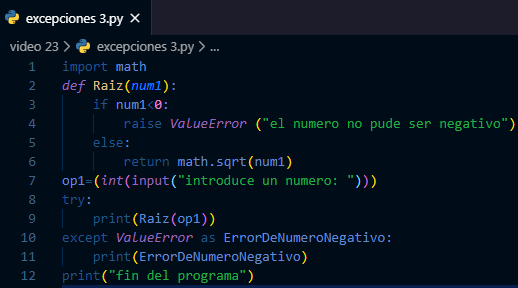
# **VIDEO 22 – EXCEPCIONES**

1. Capturas de varias excepciones
2. Clausula finally



# **VIDEO 23 – EXCEPCIONES 3**

1. Como lanzar excepciones
   1. Instrucción Raise
   2. Creación de excepciones propias



# **VIDEO 24 – PROGRAMACIÓN ORIENTADO A OBJETOS (POO)**

Hay dos tipos de paradigmas de programación:

1. Programación orientada a procedimientos
   1. Fortran
   2. Cobol
   3. Basic

* Desventajas:
  1. Mucho código en app complejas
  2. Difícil de descifrar
  3. Poco reutilizable
  4. Si hay fallos el programa cae
  5. Código espagueti
  6. Difícil de depurar

1. Programación orientada a objetos
   1. Pasar movimiento de objetos en la vida real y pasarlo a código
   2. C++
   3. Java
   4. Visual.NET

* Ventajas:
  + Programas divididos en trozos, partes, módulos, clases
  + Muy reutilizable herencia
  + Excepciones
  + Encapsulamientos
* Terminologías de la POO
  + Clase
  + Objeto
  + Instansiamientos de una clase
  + Modularizarían
  + Encapsulamiento
  + Herencia
  + Polimorfismo

# **Video 25 - PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 2**

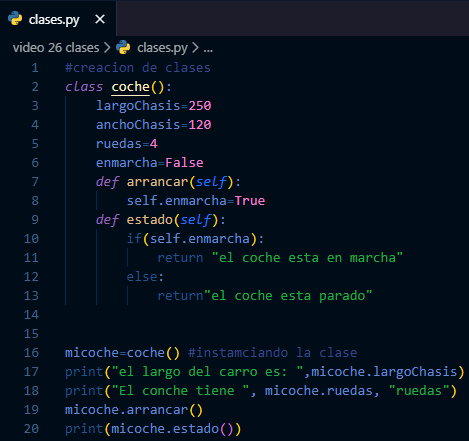
Clase: Es un modelo donde se redactan las características comunes de un grupo de objetos

Instancia: Es un ejemplo perteneciente a una clase

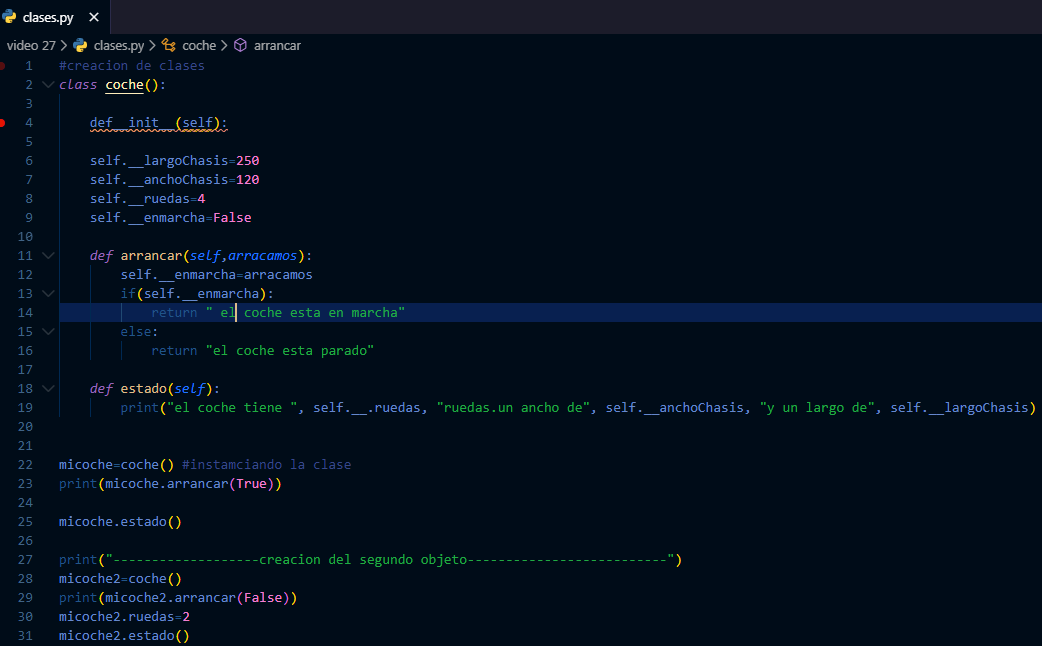
Modularizacion: hay varias clases en ella y es usado cuando hay varias clases

Objeto: tiene nombre y estos tiene atributos ejemplo: micoche.color=”rojo”

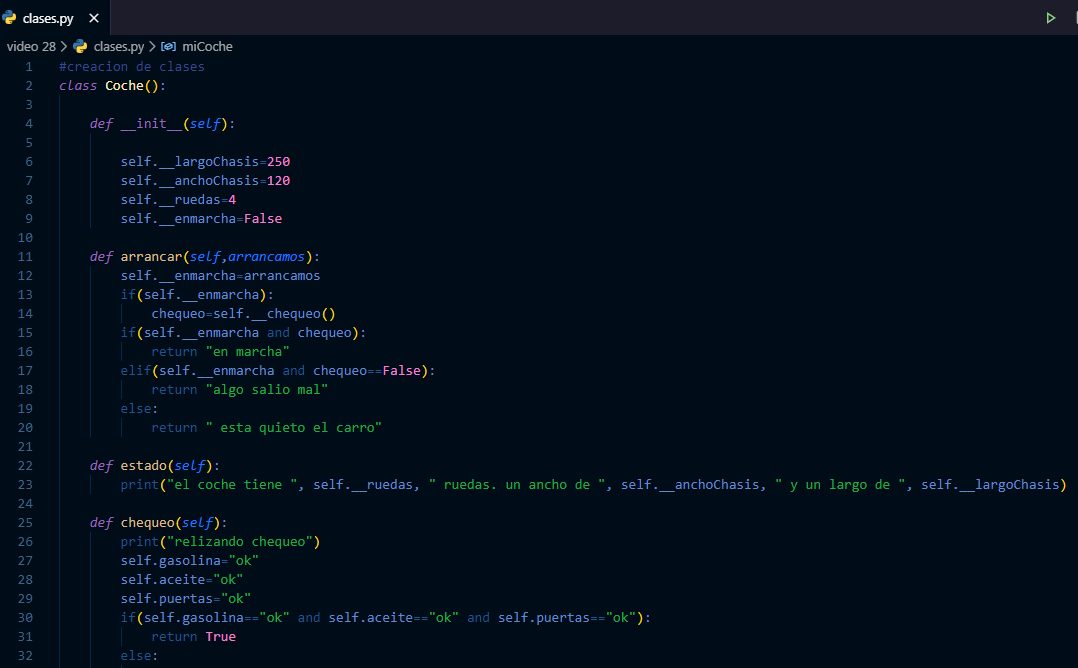
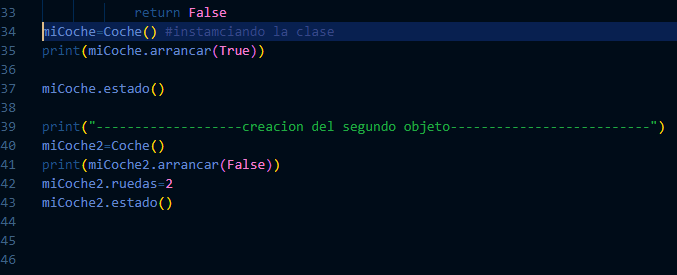
# **Video 26 - PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 3**

Creaciones e instanciación de una clase

# **VIDEO 27 – PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 4**



# **VIDEO 28 – PROGRAMACIÓN ORIENTADO A OBJETOS (POO) 5**



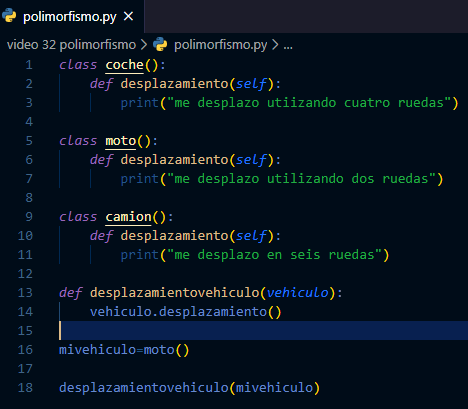
# **VIDEO 29 – HERENCIA 1**

La herencia bien de arriba hacia abajo donde hay una herencia padre y subclases hacia abajo es muy usado para reutilizar código creando objetos similares ye estas deben tener características similares para que la herencia se pueda dar.

# **VIDEO 30 – HERENCIA 2**

# **VIDEO 31 – HERENCIA 3**

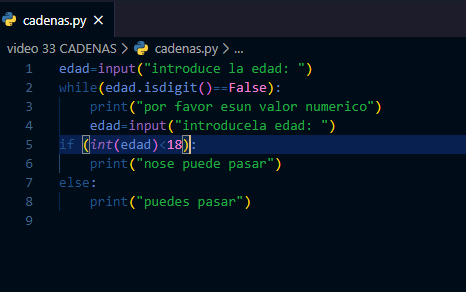
# **VIDEO 32 – POLIMORFISMO**

Es la manera de hacer una misma cosa de maneras diferentes es decir un objeto puede volverse otro objeto.

# **VIDEO 33 – METODOS DE CADENAS**

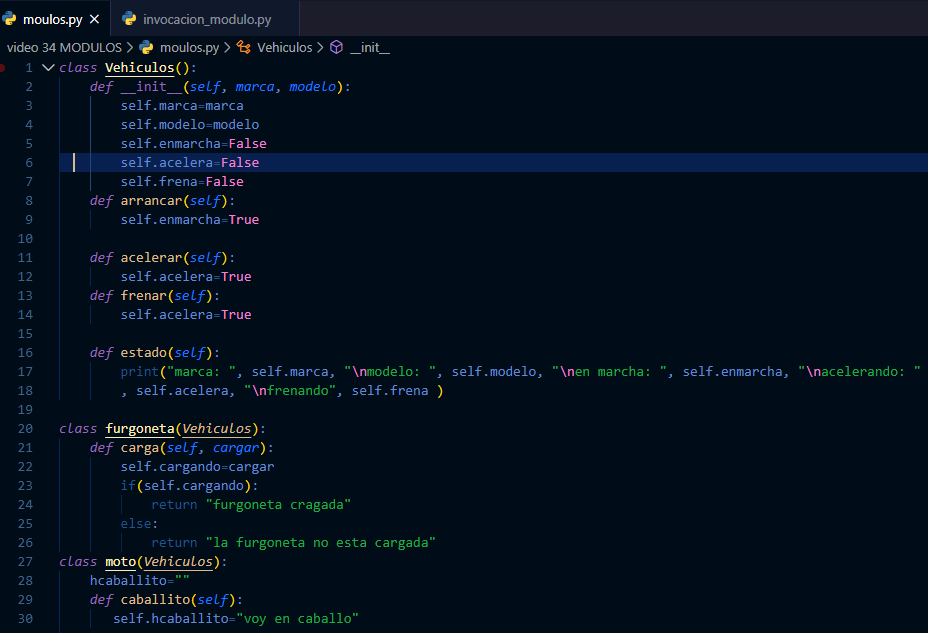
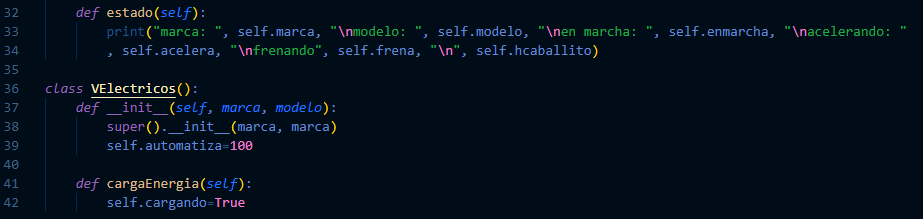
Hay varios métodos de cadenas y son:

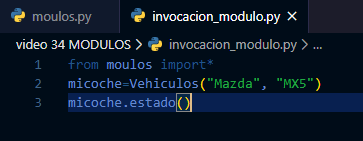
1. Upper (): convierte en mayúsculas un Strings
2. Lower (): convierte en minúsculas un Strings
3. Capitalize (): pone la primera letra en mayúscula
4. Count (): contar cuantas veces aparece una letra o una cadena
5. Find (): representa el índice en la cual aparece un carácter o un grupo de caracteres
6. Isdigit (): devuelve un booleano true o false
7. Isalum (): comprueba si son alfanuméricos
8. Isalpha (): comprueba si hay solo letras
9. Split (): separa las palabras con espacios
10. Strip (): borra los espacios al comienzo o al final
11. Replace (): cambia una palabra o letra por otra en un Strings
12. Rfind (): representa el índice de un carácter de atrás hacia adelante



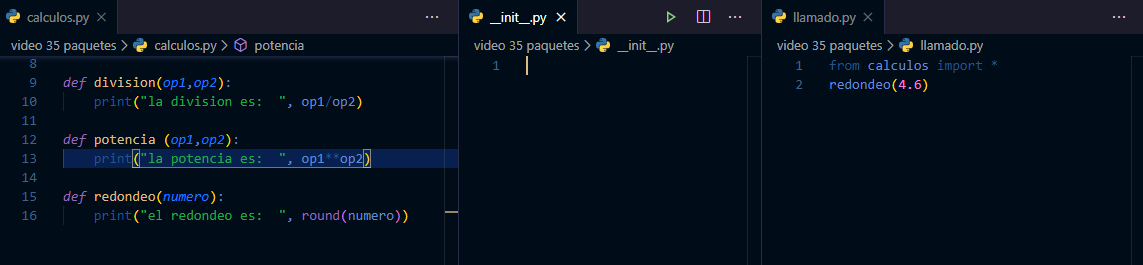
# **VIDEO 34 – MODULOS**

Son un archivo con extensión .py .pyc Python compilado o un escrito en C para CPython que posee su propio espacio de nombres y que puede contener variables funciones, clases e incluso otros módulos

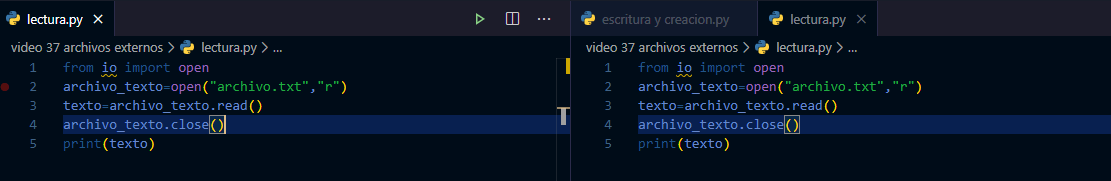
Organizan y reutilizan el código (Modularizacion y reutilización):



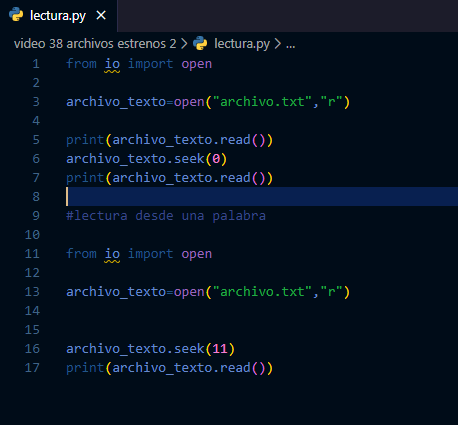
# **VIDEO 35 – PAQUETES**

Son directorios donde se almacenarán módulos relacionado entre si y sirven para organizar y reutilizar los módulos y se crea creando una carpeta y en el interior debe tener un archivo llamando \_\_init\_\_.py

# **VIDEO 37 – ARCHIVOS EXTRENOS 1**

Se usa para trabajar con archivos planos o de bases de datos.

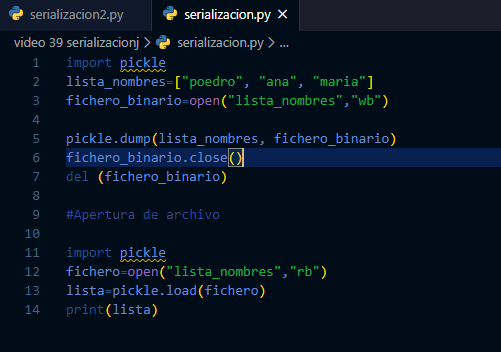
# **VIDEO 38 – ARCHIVOS EXTRENOS 2**



# **VIDEO 39 – SERIALIZACION**

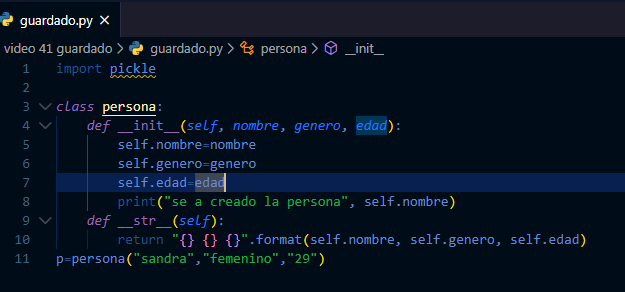
La serializa ion se puede hacer a las colecciones y objetos y se maneja con archivos externos y se usan bibliotecas como Pickle con métodos como:

1. Método dump (): volcado de datos el fichero binario externo
2. Método load (): carga de los datos del fichero binario externo



# **VIDEO 40 – SERIALIZACION 2**



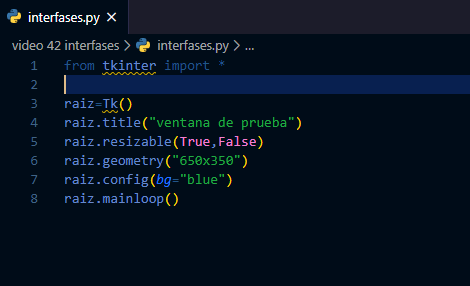
**VIDEO 41 – GUARDADO PERMANENTE**

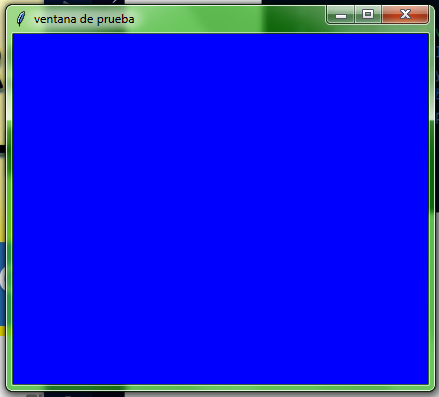
# **VIDEO 42 – INTERFASES GRAFICAS 1**

Son denominada GUI son intermediarios entre el programa y el usuario formadas por un conjunto de gráficos como ventanas botones menús casillas de verificación y para crearlas se usa librerías como:

1. Tkinter
2. WxPython
3. PyQT
4. PyGTK

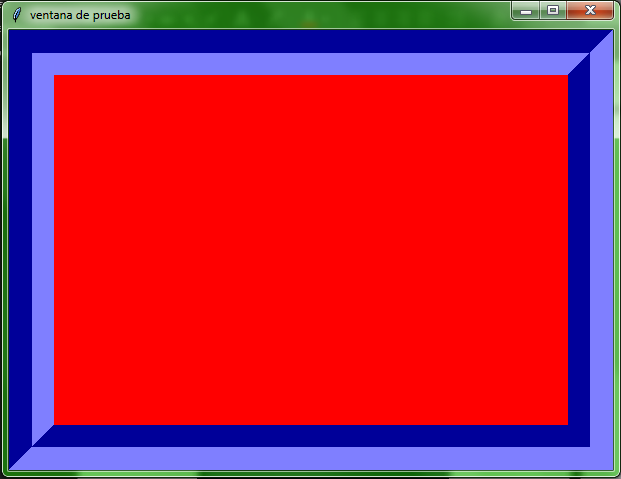
Tkinter es un puente entre Python y la librería TCL/TK y arrancamos con la raíz o ventana y dentro dela raíz se crea un frame q agrupara los elementos o widgets.





# **VIDEO 43 – INTERFASES GRAFICAS 2**

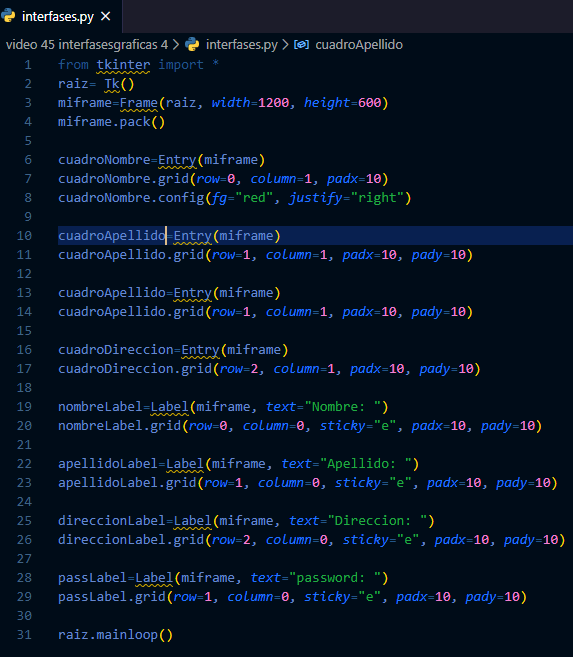
# 

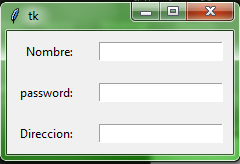


# **VIDEO 44 – INTERFASES GRAFICAS 3**

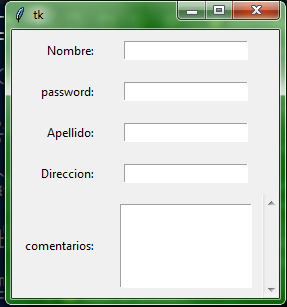


# **VIDEO 45 – INTERFASES GRAFICAS 4**

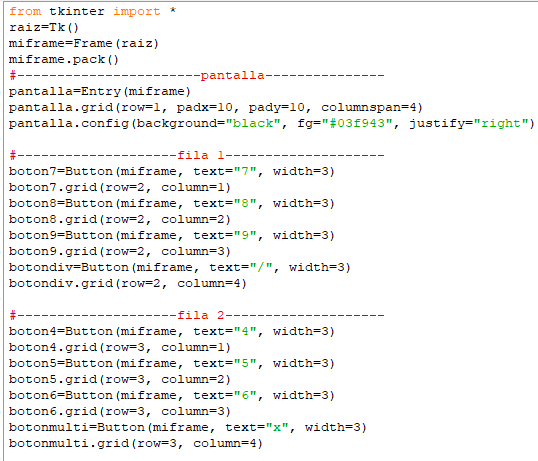


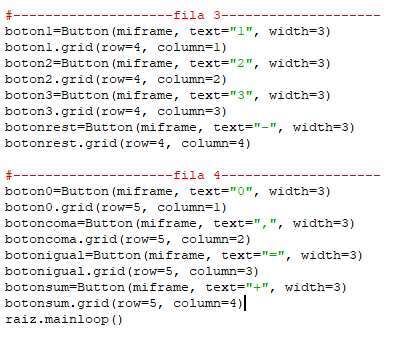


# **VIDEO 46 – INTERFASES GRAFICAS 5**



# **VIDEO 47 – INTERFASES GRAFICAS 7**



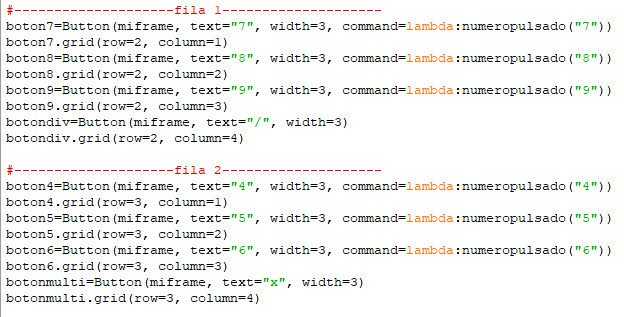


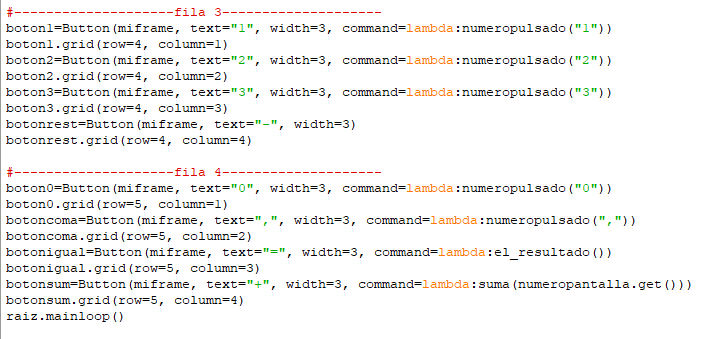
# **VIDEO 48 – INTERFASES GRAFICAS 8**



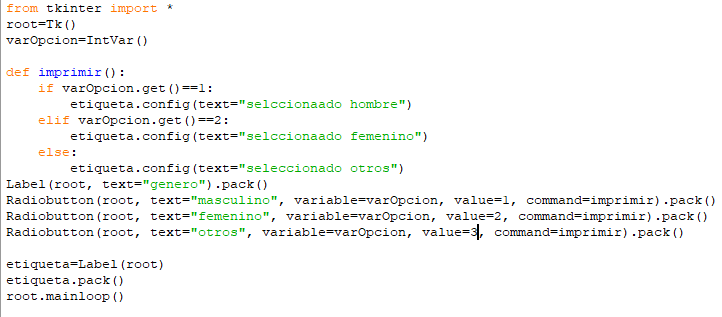
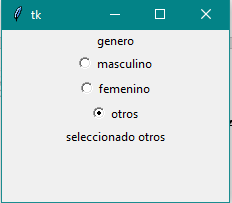
# **VIDEO 49 – INTERFASES GRAFICAS 9**







# **VIDEO 50 – RADIOBOTON**

# **REFERENCIAS:**

1. Píldoras Informáticas Canal YouTube:
   1. <https://www.youtube.com/watch?v=G2FCfQj-9ig&list=PLU8oAlHdN5BlvPxziopYZRd55pdqFwkeS>
2. Píldoras informáticas web:
   1. <https://www.pildorasinformaticas.es/course/curso-python/>
3. SublimeText:
   1. <https://www.sublimetext.com/3>
4. Visual Studio Code:
   1. <https://code.visualstudio.com/docs/?dv=win>