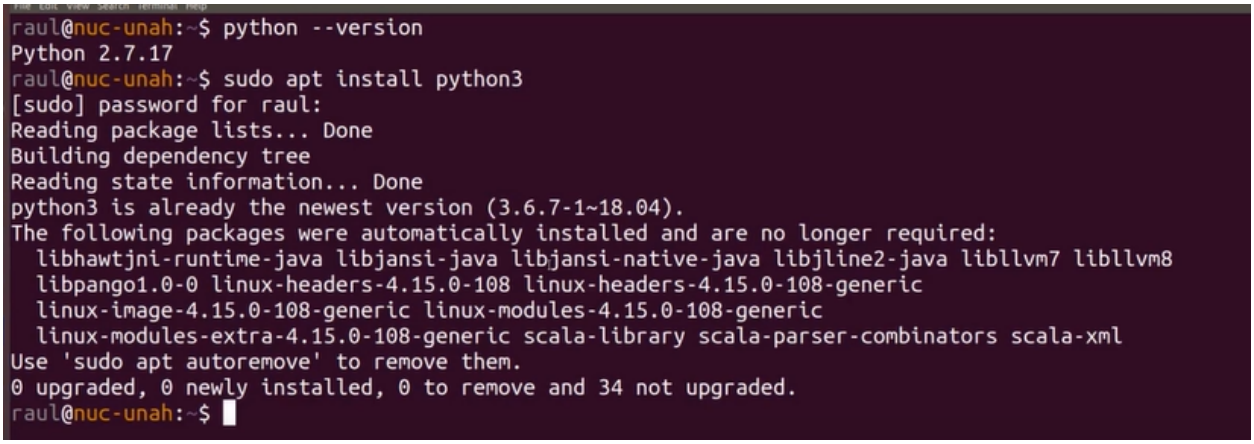
Semana 03

Instalación de Python en Ubuntu



Ejemplo:

Nuestro primer programa en python lo que hace es que lee una serie de notas que un estudiante obtuvo y imprime el **promedio.**

Entonces, para realizar este proceso primero se crea una variable puntos. Esta variable puntos o este objeto puntos mejor dicho es un objeto de tipo diccionario en python. Los objetos diccionarios son conjuntos de pares clave valor.

En este caso a b y c son las claves y 10, 8.0 y 5.0 son los valores. Los diccionarios lo que permiten es utilizar una clave para obtener el valor. De la clave a me devolvería 10, la clave b devuelve 8 y la clave c 5.

Luego tenemos dos objetos tipo enteros inicializados en cero y uno de tipo Booleano que tiene un valor de falso. Vemos que los tipos no se declaran en python simplemente el tipo se infiere o se obtiene a partir de el objeto que está a la derecha del igual.

Vemos que en python no es necesario crear una clase , crear un método, una función main para poder empezar a programar simplemente empezamos a escribir el código.

Ahora miramos acá que un ciclo while **la negación en python se hace utilizando** la palabra reservada a not. Not fin osea mientras no sea fin continúa haciendo lo siguiente, empezamos el ciclo con dos puntos

Si no se identa el código entonces no se considera que es forma parte del ciclo while.

input():

función que me sirve para leer texto desde la consola, leo una nota de la consola en este caso la nota se espera que sea a b o c.

Si la nota es algo vacío o sea no escribió nada solamente se escribió un enter entonces asignamos como fin = true. O sea ya no vamos a continuar el ciclo después. Sin embargo, si no es vacío entonces se busca si la nota está en puntos . **Not in** es un operador del diccionario que me permite verificar si está este texto que escribió el usuario pertenece o no pertenece mejor dicho del diccionario si no pertenece al diccionario entonces imprimo en la consola nota desconocida. Sin embargo si pertenece al diccionario entonces iría el else, al número de cursos le aumentaría en 1 y a total puntos y le aumentaría la cantidad de puntos que corresponde la nota que el usuario ingreso y que eso sí leen si ingresó a entonces se le sumaría 10 si ingreso b se le sumaría a 8 y se ingresó 5 se se le sumaría.

Este ciclo se va a repetir indefinidamente hasta que el usuario no ingrese nada, hasta que nota sea igual igual a vacío.

Al final entonces si el número de curso del mayor que 0 entonces se va a imprimir el promedio.

El promedio se imprime utilizando la función print y aquí pues interesante es la forma que sea imprime. Tenemos un string es indicado por comillas simples tiene un método que es el método format. El método format recibe un argumento en este caso lo que está imprimiendo en el promedio total puntos entre el número de cursos me imprimiria y el promediome calcularia un número real y este valor que se produce dentro del format va a ser mostrado en vez de la llave cero, es decir el string colocó este esta simbología llave cero para indicar que el primer argumento del format se va a poner dentro del las llaves, yo puedo utilizar llave 1 y mandar un segundo argumento a format para que se muestre otro valor y llave 2 y así sucesivamente para mostrar más valores dentro de un string la ventaja de esto es que de esta forma no tenemos que estar concatenando strings que es algo una práctica bastante incómoda y que también puede ser muy común.

```
puntos = {'A':10.0, 'B':8.0, 'C':5.0}
numeroCursos =0
totalPuntos =0
fin = False

print('Bienvenido, Ingrese sus notas')
while not fin:
  nota = input()
  if nota == '':
    fin = True
  elif nota not in puntos:
    print('Nota desconocida')
  else:
    numeroCursos += 1
    totalPuntos += puntos[nota]

if numeroCursos>0:
  print('Su promedio es {0}'.format(totalPuntos/numeroCursos))
```

```
puntos = {'A':10.0, 'B':8.0, 'C':5.0}
numeroCursos =0
totalPuntos =0
fin = False

print('Bienvenido, Ingrese sus notas')
while not fin:
    nota = input()
    if nota == '':
        fin = True
    elif nota not in puntos:
        print('Nota desconocida')
    else:
        numeroCursos += 1
        totalPuntos += puntos[nota]

if numeroCursos>0:
    print('Su promedio es {0}'.format(totalPuntos/numeroCursos))
```

```
Bienvenido, Ingrese sus notas
 A
 B
 C

Su promedio es 7.666666666666667
```

Objetos e Identificadores en Python

¿ Que son los objetos?

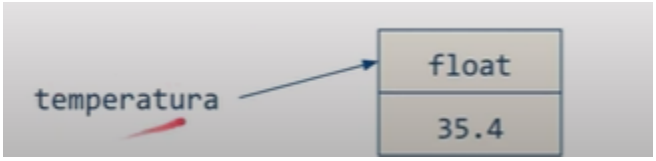
son la **forma en que python representa los** valores.

En python todos los valores son objetos. El comando más importante de python o el enunciado mas importante sería el de asignación. El enunciado de asignación pues tiene tres partes:

1. Un lado izquierdo
2. un igual y
3. El lado derecho

Ejemplo: temperatura = 35.4

El lado izquierdo vamos a tener un nombre o un identificador que se va a asociar mediante el igual a un objeto que produce el lado derecho, sabemos que el lado derecho puede haber una función una expresión que al final se evalúa a un objeto. En este caso tenemos un objeto literal el cual se asocia a al nombre temperatura, esto gráficamente conceptualmente podemos verlo asi:



vemos que tenemos el nombre asociado a un objeto en este caso este es un objeto de tipo float.

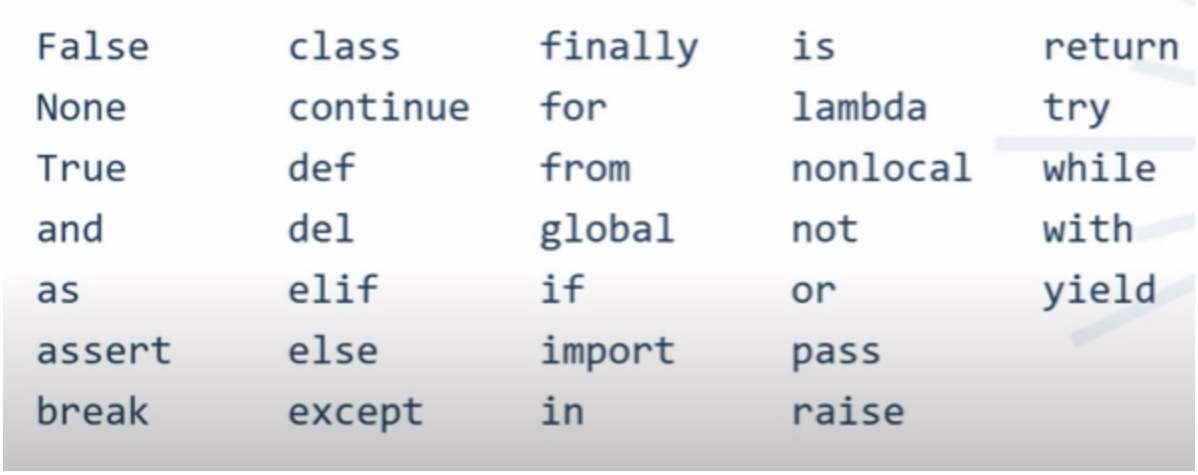
Los objetos en python tienen tipo lo que pasa que este tipo no está declarado de forma evidente sino que se conoce a partir en este caso del literal del objeto, en este caso literal porque tiene un punto entonces es un float .

Identificadores

Los identificadores son sensitivos a mayúsculas y minúsculas. Es decir no es lo mismo HOLA con mayúscula a hola con minúscula son identificadores diferentes.

También en python se suele utilizar el guión bajo para unir las palabras distintas digamos si tengo un identificador como producto punto se suele escribir producto que un bajo punto no producto mayúsculas p punto sino que usa para unir las palabras. También se pueden usar números y la restricción es que no se pueden utilizar o no se puede utilizar un número para iniciar el nombre de un identificador sino que tiene que empezar con una letra

Existe un grupo de palabra reservada que no se pueden utilizar son palabras que utiliza el lenguaje para fines especiales. Se puede esperar de que son utilizadas en otros lenguajes de programación también y por tanto estas palabras no se pueden usar como identificadores



False	class	finally	is	return
None	continue	for	lambda	try
True	def	from	nonlocal	while
and	del	global	not	with
as	elif	if	or	yield
assert	else	import	pass	
break	except	in	raise	

Tipado Dinamico

Python es un lenguaje de tipo dinámico qué significa esto que los identificadores pueden estar asociados a un objeto de un tipo en un momento y pueden estar asociado a un objeto de un tipo diferente en un momento posterior

vemos acá por ejemplo que a este identificador está asociado a un objeto de tipo string, yo imprimo a y pues imprimiria hola, luego yo puedo asociar a un objeto de tipo entero y se imprimiría el 2 luego yo puedo asociar a un tipo a un objeto de tipo booleano y pues podría usarlo dentro de un if para imprimir verdadero

```
a = 'hola'
print(a)
a = 2
print(2)
a = True
if (a):
  print('Verdadero')
```

A pesar de que python es un lenguaje de tipo dinámico y que el objetos identificadores pueden asociarse a objetos de diferentes tipos esta no es una práctica conveniente cuando se está desarrollando un programa.

El único fin de esto es que podamos utilizar esto debido a que python es un lenguaje también interactivo, es posible abrir la consola de python y utilizar y escribir instrucciones de manera interactiva y obtener respuesta inmediata.

Sin embargo cuando crea un programa en python desde el editor de texto lo conveniente es que cada identificador esté asociado a un objeto del mismo tipo siempre, sin embargo es el programador el que tiene la responsabilidad de hacer esto

¿Que son los Alias?

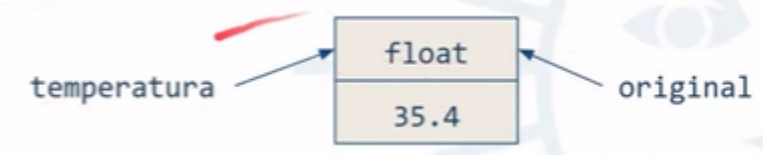
Los alias es una forma de tener varios nombres o varios identificadores para un mismo objeto

Es posible crear un alias asignando un identificador a un objeto existente por ejemplo:

yo tengo el objeto temperatura

temperatura = a 3.54 sin embargo yo puedo hacer original temperatura que es lo que hace esto? crear un alias, otro identificador para el mismo objeto.

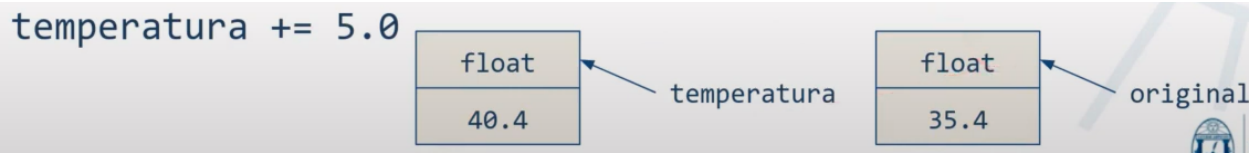
Entonces esto conceptualmente gráficamente lo podemos ver así :



dos identificadores para el mismo objeto.

Los alias también se pueden romper. si yo a uno de estos objetos en este caso temperatura le sumo cinco por ejemplo entonces estoy rompiendo el alias temperatura vale ahora 40.4 cree un nuevo objeto y original vale 35.4

vemos acá que este comportamiento parece un poco extraño no pareciera que fuera lo común en lenguajes de programación sin embargo este una práctica común en otros lenguaje de programación vamos a volver o vamos a aclarar este concepto un poquito más cuando hablemos acerca de la inmutabilidad de los objetos en este caso los objetos de tipo flow son inmutables por esta razón este flow 35.4 no se puede cambiar sino que al modificarlo se crea un objeto nuevo y se rompe la ley.



Clases Predefinidas Numéricas

Clases Predefinidas

Predefinidas: Que se pueden utilizar sin importar nada adicional

Algunas de las clases predefinidas mas usadas en Python son:

Class	Description	Immutable?
bool	Boolean value	✓
int	integer (arbitrary magnitude)	✓
float	floating-point number	✓
list	mutable sequence of objects	
tuple	immutable sequence of objects	✓
str	character string	✓
set	unordered set of distinct objects	
frozenset	immutable form of set class	✓
dict	associative mapping (aka dictionary)	

Clases numéricas : Bool, int, float

Secuencias de elementos : list, tuple, str

Representar conjuntos: set, Frozenset

Representar diccionarios: dict.

Las clases que están marcadas con un check **son clases inmutables y las clases que** no tienen el check son clases mutables

¿C**uál es la diferencia entre las clases mutables y las inmutables**?

Las clases mutables una vez que se ha creado un objeto de dicha clase este objeto puede cambiar su valor. Por el contrario en la clase inmutable es un objeto de esta clase una vez que ha sido creado ya no puede cambiar su valor tiene que ser sustituido por un objeto diferente.

Vemos entonces que las clases la mayoría son inmutables sin embargo tenemos las listas, los sets y los diccionarios como las tres excepciones a esta regla dentro de la clase predefinida mas comunes. Las listas, los diccionarios y los sets.

Clases numéricas

- Clase Bool
    - Nos sirve para representar valores Booleanos y sabemos que solo hay dos posibles instancias o valores que pueden haber True y False. Recordando que en python tenemos diferencia entre mayúsculas minúsculas, hay sensibilidad entonces true va con T mayúscula F maysucula. l

    - Las clases tienen un constructor con el mismo nombre de la clase. El constructor permite crear objetos de tipo bool a partir de otros objetos.
    - En el caso de que le enviemos números ya sea flotantes o números enteros se evaluarán a false se convertirán a false si son cero o si son diferentes de cero se convertirán en True.
    - En el caso de decir que le mandemos una secuencia el constructor de bool en la que se retornará un valor false si la secuencia está vacía de lo contrario se retornará un valor True.
- Clase Int
    - Con respecto a la clase y algo muy interesante es que no hay diferentes tipos de int, no tenemos int cortos ni largos. Simplemente tenemos una sola clase int que tiene magnitud arbitraria es decir el lenguaje que se cargará el motor del lenguaje se encargará de darle una capacidad almacenamiento suficiente para almacenar o para soportar el número que se esté manejando el número entero y éste solamente va a estar limitado por la memoria del computador. Por tanto podemos manejar entero muy grandes con python sin tener que hacer cambios. Obviamente esto tiene un impacto en la eficiencia del lenguaje, en python pues se prefiere o se da favor a la facilidad de programación y a la conveniencia más que a la eficiencia.
    - El constructor de como en la clase bool también tenemos un constructor de enteros. Este constructor nos permite construir un entero a partir de un número flotante truncando los por ejemplo 2.6 al mandarse el constructor de que retornaría un 2 lo trunca los corta no lo apróxima queda la parte entera,
    - También puede recibir cadenas válidas por ejemplo un string que contenga un número entero y si se hace la conversión automáticamente.

- Clase Float
    - Es la única clase que nos permite manejar números flotantes con el punto decimal
    - En el caso de Float a diferencia del int tenemos una apreciación fija es decir que los Floats tienen una limitación, esa limitación está está dada por la por la

implementación de python y habría que validarla en cada uno de los motores, un motor específico que se esté trabajando.

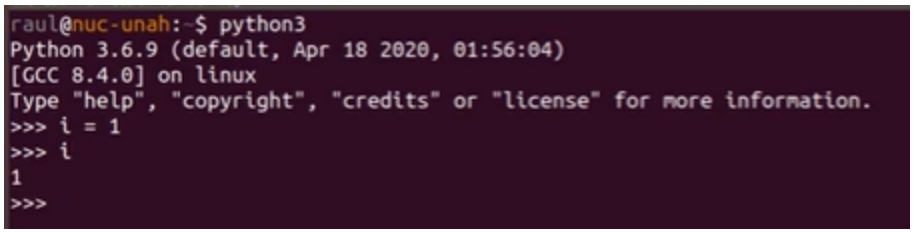
- En el caso de flow también hay también
- Hay un constructor como en el resto, puedo construir floats a partir de enteros y puedo construir floats a partir de cadenas
- Por otro lado también puedo hacerlo con un booleano, float(True) retorno 1.0 y float(False) me retorna 0.0

  Algo similar ocurrirá con el constructor de int int(True) me va a devolver 1 e int(False) me va a devolver 0

Ejemplo

Para ejecutar el intérprete de python simplemente tenemos que escribir el comando **python3 M**ediante el intérprete de python podemos ejecutar código en python y obtener una retroalimentación inmediata del resultado de ese código.

Por ejemplo podemos crear un entero, asignarle 1 y ya fue asignado puedo ver el valor de las variables del objeto que yo creo simplemente escribiendo el nombre del objeto y presionando enter.

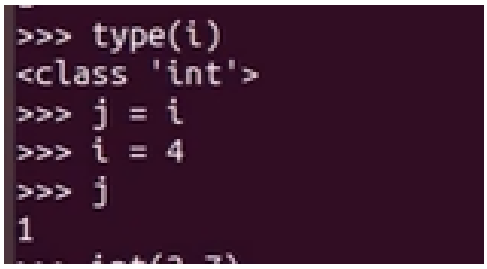


Por ejemplo el valor de i es 1. i es un número entero porque éste era un literal objeto entero por tanto y el tipo de i es entero. Podemos utilizar la función type para confirmar la clase

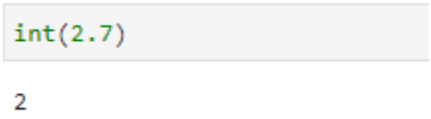
```
type(i)
##Clase int
```

ahora recordando los enteros son inmutables por esta razón si yo creo

un alias tenemos dos nombres para el mismo método pero si yo luego hago que i sea igual a cuatro por ejemplo, no estoy modificando el objeto uno simplemente estoy haciendo que el nombre i haga referencia esté haciendo referencia a un objeto nuevo al objeto j entonces si imprimo j muestra1 porque el uno no fue modificado simplemente y se hizo referencia a un momento nuevo pero j seguía haciendo referencia al mismo objeto que inmutable

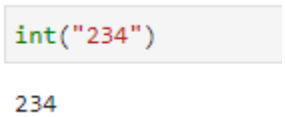


En el constructor de int tiene el mismo nombre de la clase y me permite construir enteros a partir de otros objetos por ejemplo a partir de un flotante le quitó la parte decimal

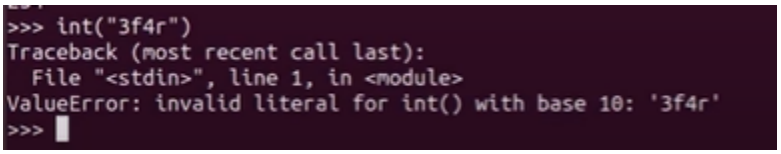


Obtener a partir de un booleano no como True puedo obtener el número 1

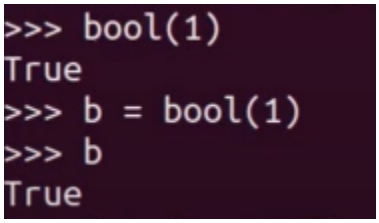
y a partir de una cadena por ejemplo puede obtener un número entero 234



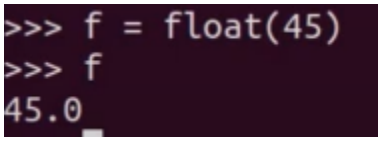
y obviamente si la cadena está más formada voy a obtener una excepción por ejemplo



también tenemos el constructor de bool que acabamos de ver por ejemplo bool si recibe el entero uno construye un True esto yo obviamente lo puedo tener una variable y tener allí el valor de True.



por ejemplo puedo construir un float a partir de un entero entero 45 yo tengo que ser igual 45 punto 0 entonces como podemos ver el intérprete de payton nos permite ejecutar código en python y dividió tener una retroalimentación inmediata de ese código esa es una de las ventajas de que python sea un lenguaje interpretado



un lenguaje interpretado pues usualmente va a tener un intérprete significa que su código es traducido a lenguaje de máquina línea a línea cada una de estas líneas se traduce a diferencia del lenguaje compilado que se toma todo el código se traduce al lenguaje de máquina python no esto le da a python versatilidad, flexibilidad pero también esto tiene un impacto en la eficiencia sin embargo hay librería de payton que nos permiten ayudar a mejorar eso.

Clases Predefinidas Secuencias

Qué son las secuencias?

las secuencias son clases que sirven como contenedores de otros objetos y que a esos objetos que contienen les ponen un orden definido y por eso se llaman secuencias porque
existen elementos existe elemento uno el dos y así sucesivamente hasta la cantidad de elementos que tengan.

Las tres clases secuencias mas importantes en Python, son:

- list
- tuple
- str

Clase List

La clase list es una estructura referencial, qué significa esto?, significa que lo que almacena en realidad **la secuencia** son las referencias o los apuntadores o las direcciones de memoria de los objetos reales o de los objetos que pertenecen a la secuencia. Entonces las listas podemos considerarlas como arreglos porque almacenan en forma continua en memoria pero no los objetos en sí sino las referencias o los apuntadores a los objetos que pertenecen a las listas.

Que cosas pueden pertenecer a un list?

cualquier cosa cualquier objeto de python puede pertenecer a un list .

cualquier cosa se refiere a que pueden haber cosas mezcladas puede haber un entero puede haber un número un string etcétera. Ejemplo:

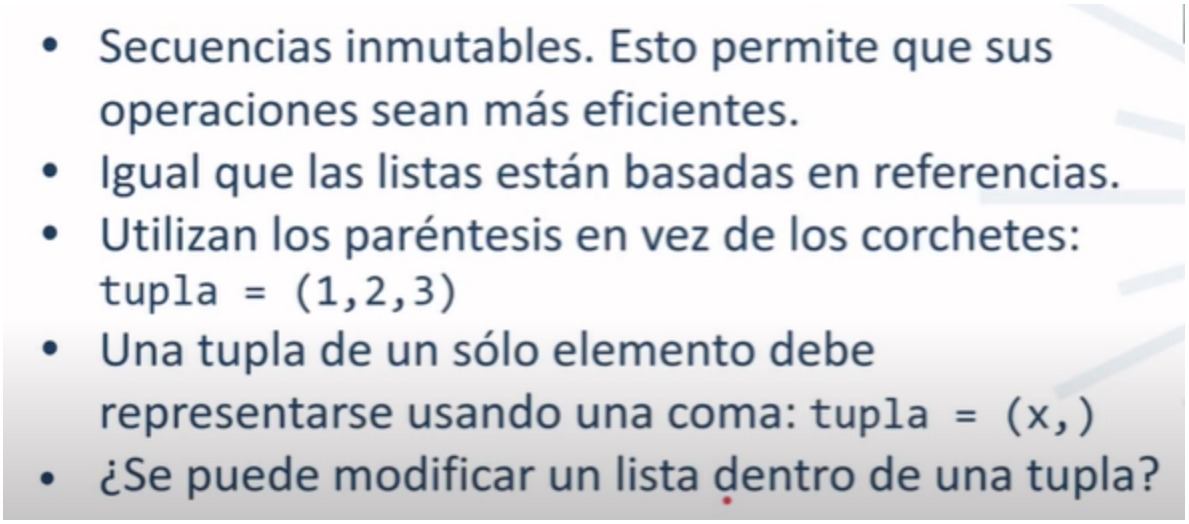
De manera continua en memorias estan guardado los apuntadores, la referencia pero el primer elemento es un entero el segundo en un string el tercero es un booleano

- las listas utilizan índices basados en cero. usan índice pasado en 0 eso quiere decir? que el primer elemento se tiene un índice 0 el segundo un segundo y así sucesivamente
- En python por defecto no existe el tipo a rey por lo que las listas son lo que utilice utilizaríamos en vez de un arreglo que se utilizaría en otro lenguaje. Ahora sí pueden haber arrays pero esto tiene que ser con una clase o con una librería que importe el tipo de arreglo.
- otra cosa importante de las listas que pueden aumentar su tamaño y disminuir su tamaño las listas son mutables recordando y pueden aumentar en su número de elementos y disminuir el tamaño máximo de una lista está definido o está limitado mejor dicho por el tamaño máximo la capacidad máxima de memoria en el computador.

Ejemplo de cómo definir una lista yo puedo hacerlo con el constructor de list pero esto es poco común le mandó al constructor un elemento. Lo más común utilizar los corchetes

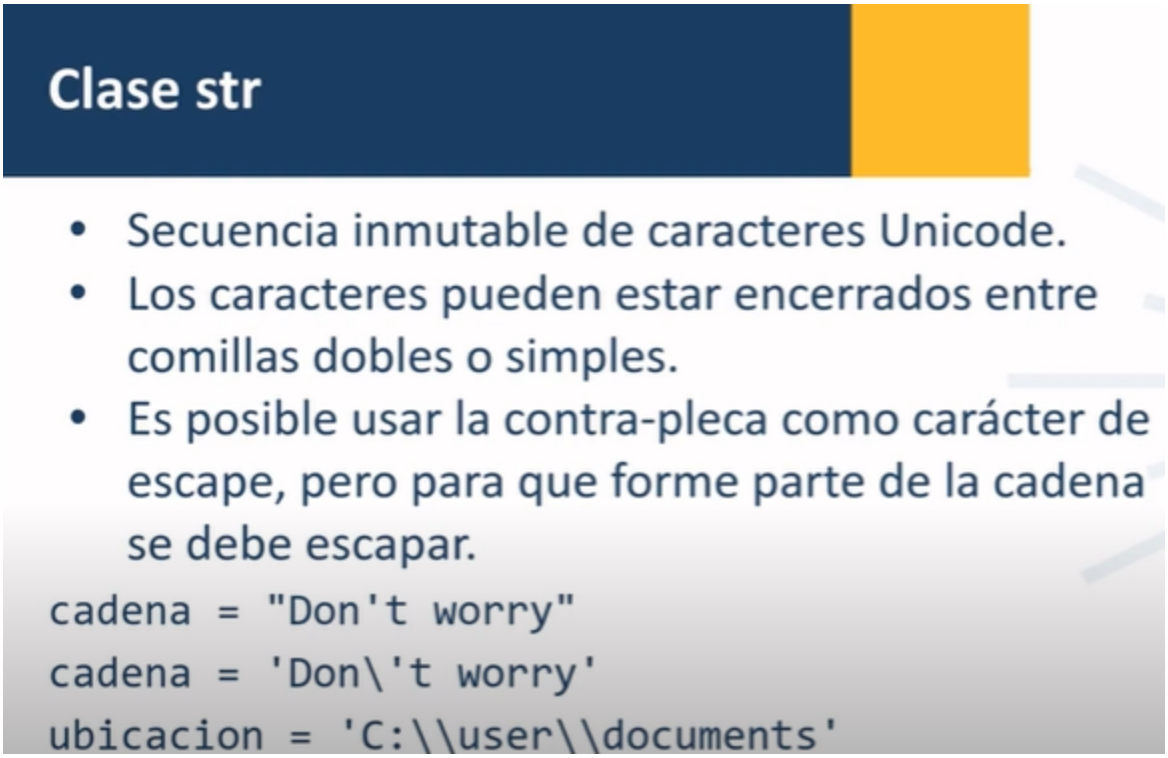
```
a = list(1, 'hola', True)
a = [1, 'hola', True]
```

Clase tuple



LA respuesta es si, porque la tupla lo que contienen son referencias, entonces al modificar una lista, puedo aumentarla de tamaño, pero su referencia sigue siendo la misma por tanto la tupla queda sin modificar pero la lista si puede modificarse.

Clase str



Unicode:

caracteres unicode lo que nos permite utilizar caracteres de distintos alfabetos para no solo el alfabeto latino ahora sino que el alfabeto **griego por ejemplo, chino.**

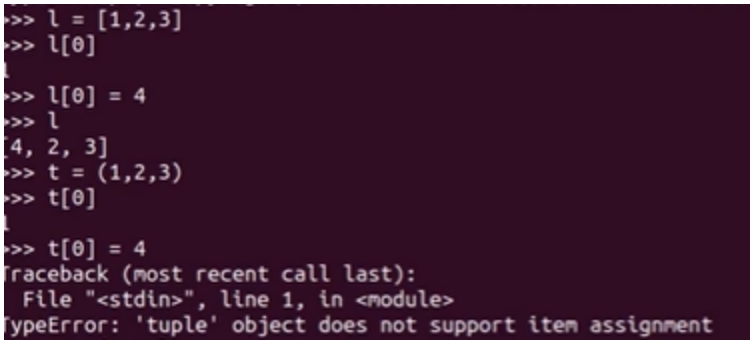
\sirve para \n, tabulador

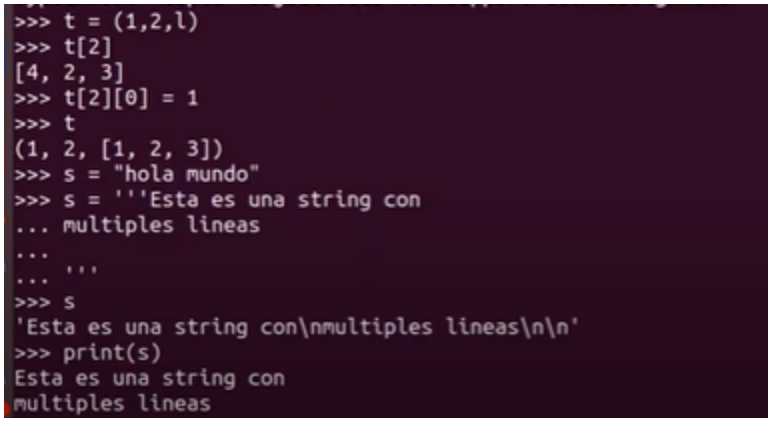
dos contraplecas en un string implica una contrapleca en la realidad

Clase str

- Además es posible usar la triple comilla doble """ o simple ''' para evitar escapar los saltos de línea.

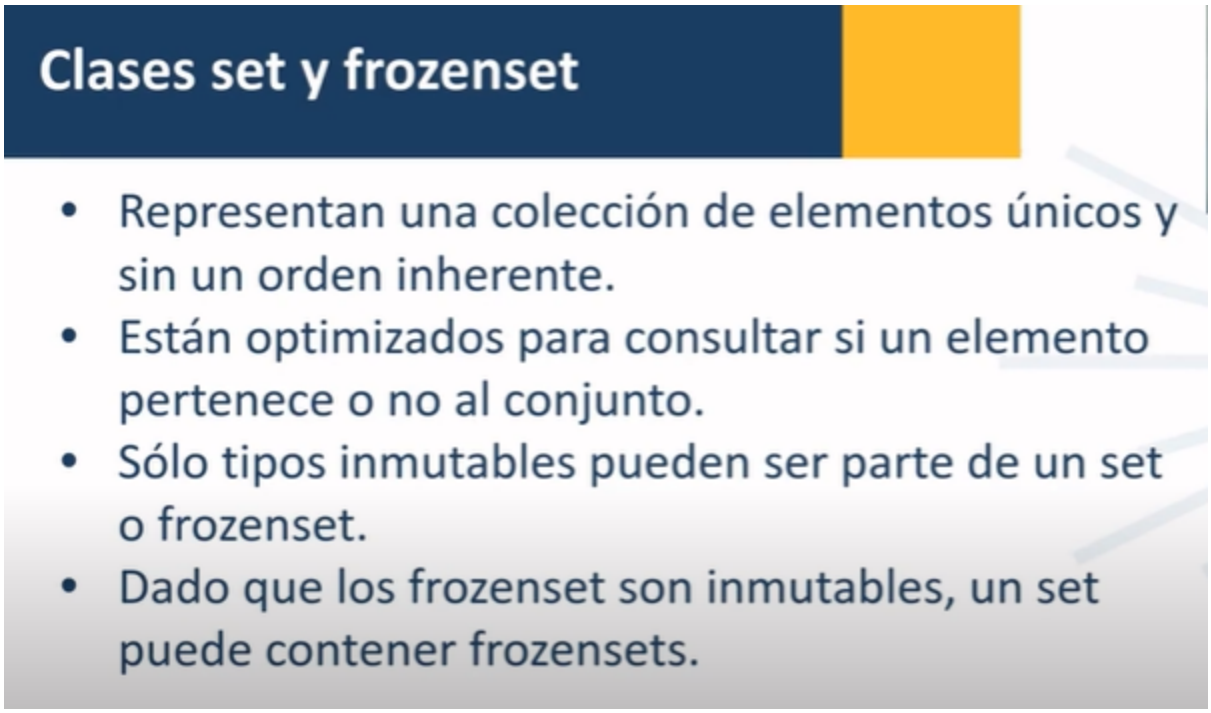
```
cadena = """Este es un ejemplo de una cadena
con múltiples saltos de línea que
no es necesario escapar con contra-pleca.

Fin de la cadena."""
```





Clases Predefinidas Conjuntos y Diccionarios

Clases para representar conjuntos

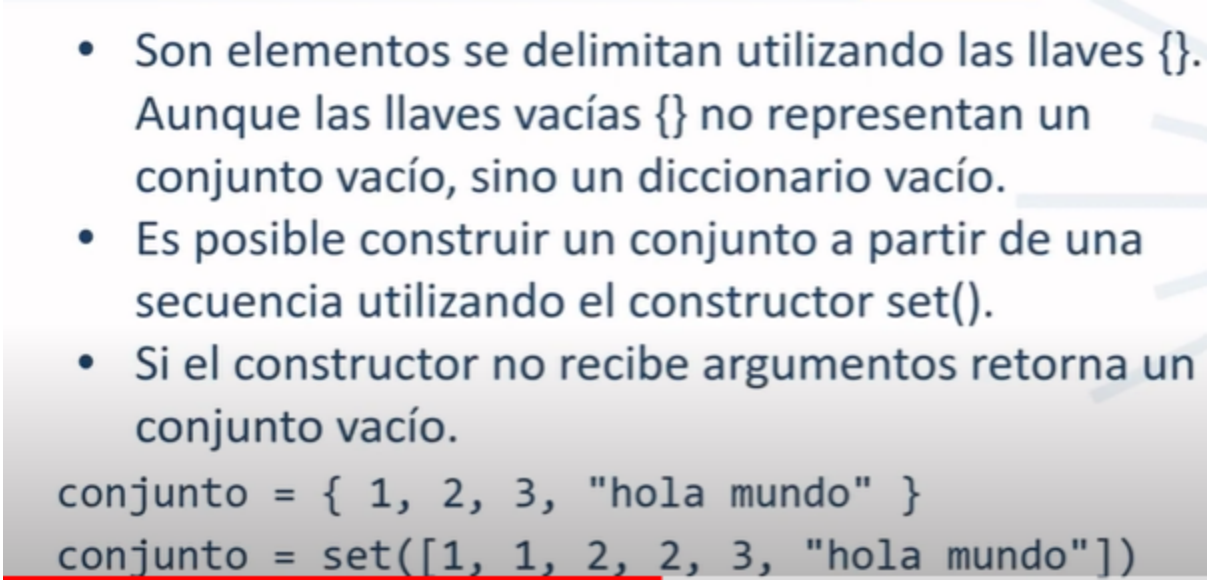


cuál es la función más importante

de los conjuntos su función más

importante es saber si un elemento

pertenece o no pertenece al conjunto



Los conjuntos no repiten los elementos

Clase Dict

Claves, valor.