



UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES

**Centro de
e-Learning**

UNIDAD DIDÁCTICA I

DIPLOMATURA EN PYTHON

Centro de e-Learning SCEU UTN - BA.

Medrano 951 2do piso (1179) // Tel. +54 11 4867 7589 / Fax +54 11 4032 0148

www.sceu.frba.utn.edu.ar/e-learning

Módulo I – Nivel Inicial I

Unidad I – Introducción.

Centro de e-Learning SCEU UTN - BA.

Medrano 951 2do piso (1179) // Tel. +54 11 4867 7589 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning



Bloques temáticos:

- 1.- Instalación.
- 2.- Como funciona Python.
- 3.- Variables y comentarios
- 4.- Asignación dinámica, garbage collection y referencias compartidas.

Glosario

GUI: Interfaz gráfica.

IDLE: Editor de texto que viene por defecto en cada distribución de python.

Path: ruta al ejecutable de python. En Windows la variable path contiene las rutas a los programas principales del sistema operativo.

Objeto: Esta palabra se utilizará con dos finalidades. En primer lugar llamaremos objeto a todo elemento que podemos manejar desde python, desde un archivo, una base de datos, o una variable en general. En segundo lugar en el nivel intermedio lo utilizaremos para definir la instancia de una clase, es decir la creación de un objeto de la clase. Estos términos escapan el nivel del curso inicial y lo veremos en el nivel intermedio.

Objeto inmutable: Es aquel que sí es modificado se convierte en otro objeto para python.

Objeto mutable: Es aquel que puede ser modificado parcialmente y python lo sigue considerando el mismo objeto.

PVM (Python Virtual Machin): Programa que realiza el papel de Intérprete del lenguaje.

Variable: Una variable se puede considerar como un símbolo que puede ser reemplazado o que toma un valor determinado, como puede ser un valor numérico en una ecuación o expresión matemática en general. Las variables se pueden utilizar para guardar datos de diferentes tipos, por ejemplo: enteros, caracteres, listas, arrays, diccionarios u objetos en general.

Variables de entorno: variables del sistema operativo Windows que deben ser modificadas durante la instalación para el correcto funcionamiento de python.

1. Instalación

La versión de Python que vamos a instalar es la 3.7 lanzada el 27 de junio del 2018, esta versión posee una performance mucho mayor a las anteriores y a la fecha es la última versión estable.

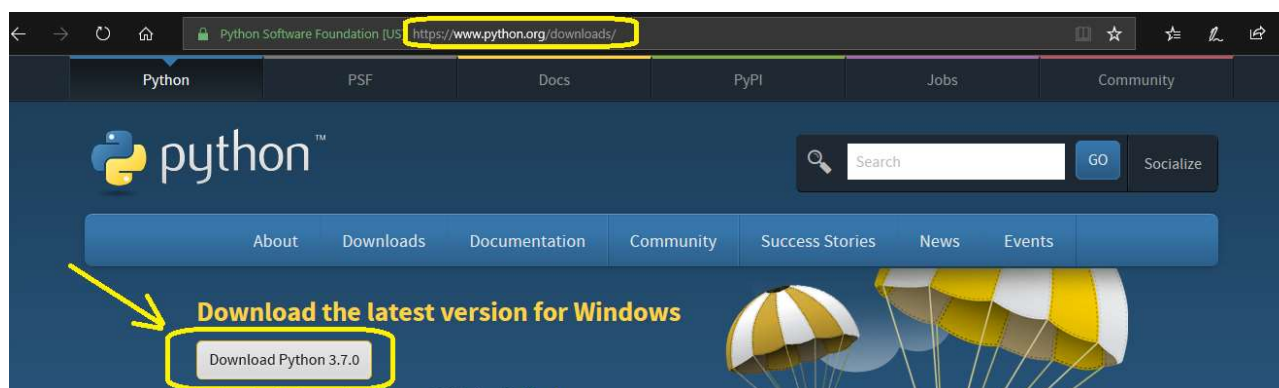
Nota: Si en el momento de leer el material el alumno encuentra una actualización, puede descargar la nueva versión y seguir los mismos pasos.

1.1. Descargar

Accedemos al sitio:

<https://www.python.org/downloads/>

Y descargamos el paquete correspondiente al sistema operativo que tenemos, en este caso para trabajar en facultad, lo haremos sobre windows de 64bit el cual es un ejecutable.

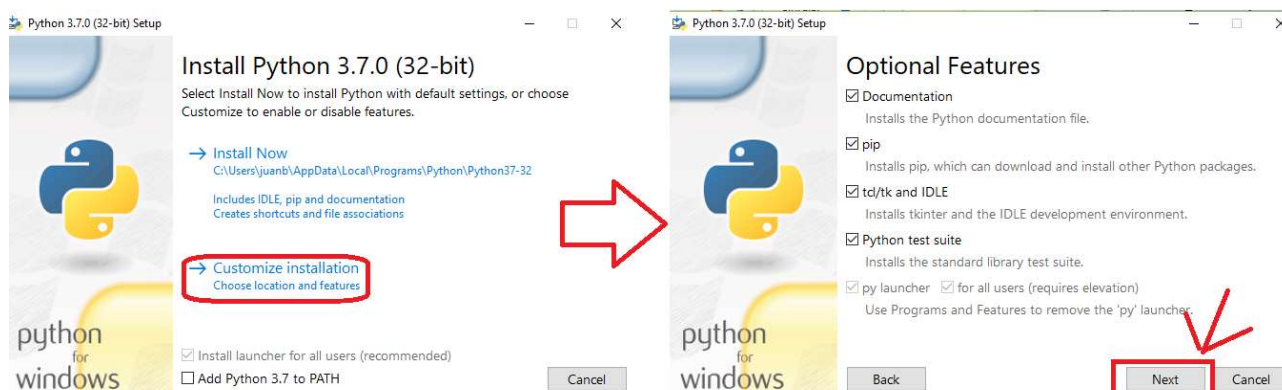


1.2. Instalación

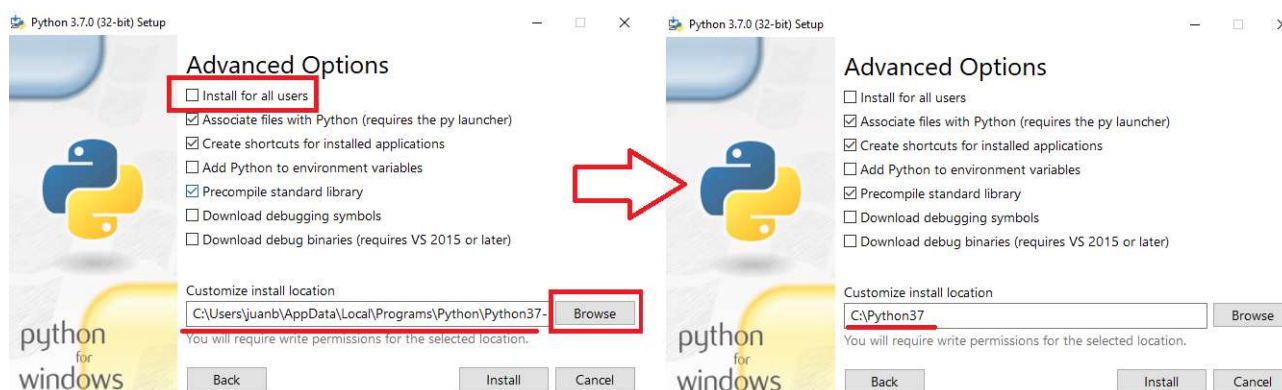
A partir de la versión 3.5, la instalación solo requiere que presionemos en “Install Now” y que tildemos el campo de “Add Python 3.X to PATH”, aquí seleccionaremos la opción que nos permite personalizar la instalación para poder seleccionar el directorio de instalación y comprender cómo realizar la instalación en versiones previas. No tildamos el campo para agregar la ruta al path.

Centro de e-Learning SCEU UTN - BA.

Medrano 951 2do piso (1179) // Tel. +54 11 4867 7589 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning



Por defecto la opción de instalación lo instala para el usuario actual a no ser que seleccionemos que lo instale para todos los usuarios o introduzcamos la ruta donde queremos instalarlo buscándola con el botón “Browse”.

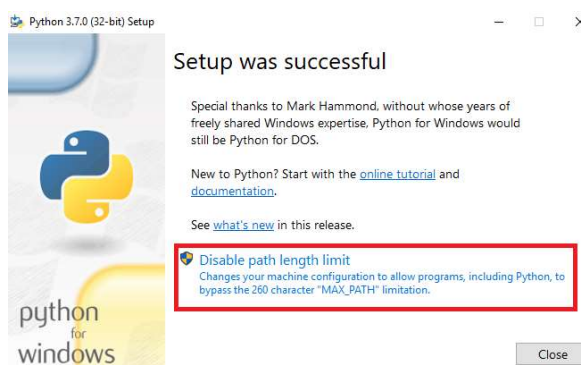


Debemos tener cuidado en el caso de seleccionar una carpeta a partir de Browse ya que si seleccionamos por ejemplo el disco “C”, agregaría todos los archivos sueltos dentro del disco. Para evitar esto indicamos el nombre de la carpeta que queremos que cree en el momento de la instalación dentro del disco para que todos los archivos se agreguen ahí dentro. Como ejemplo podríamos poner:

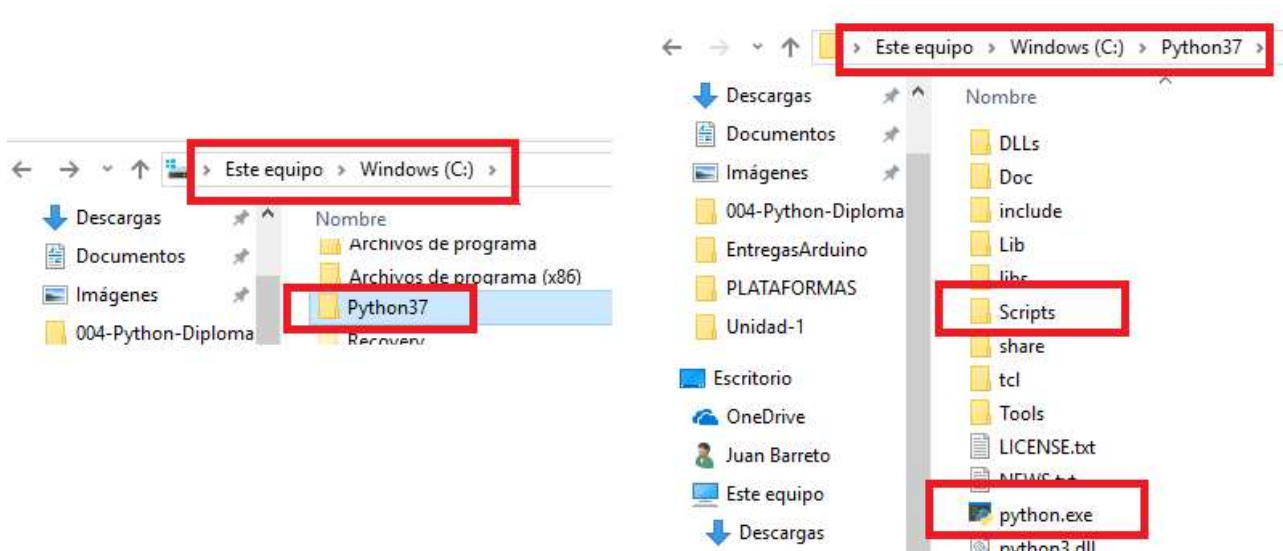
C:\Python37

Y se crearía la carpeta “Python37” con la finalidad de contener todos los archivos a instalar.

Finalmente presionamos “next” para comenzar la instalación, y deshabilitamos la limitación del path para finalizar.



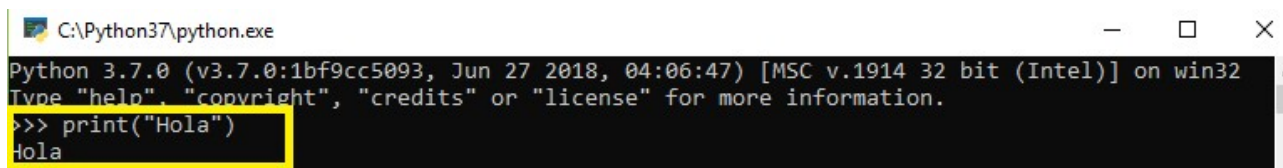
Ya podemos ingresar a la carpeta y observar que dentro se encuentra el ejecutable de python “python.exe” y un directorio muy importante “Script” al cual haremos referencia un poco más adelante en esta unidad.



Si damos doble click sobre el ejecutable se nos abrirá una terminal desde la cual ya podemos utilizar python, a modo de ejemplo escribiremos:

Print (“Hola”)

Podremos observar cómo nos retorna “Hola”.





Configuración.

En nuestro sistema operativo, podríamos tener instalada cualquier versión de python, incluso todas ellas al mismo tiempo, sin embargo la versión que estaría ejecutándose en el sistema sería aquella que se encontrara agregada en las variables de entorno (VER pág 13) del usuario que está logueado. Si abrimos el cmd del sistema y escribimos:

```
python
```

Nos saldría un mensaje diciendo que **“python no se reconoce como un comando interno o externo, programa o archivo por lotes ejecutable”** ya que no reconoce que python esté instalado.

Sin embargo si escribimos:

```
set path=%path%;C:\python37
```

Vemos como ahora podemos escribir python e ingresaríamos a la versión de Python que se encuentra indicada en la ruta resaltada en rojo en la línea de código anterior, pudiendo ahora repetir la línea “print(“Hola”) y se comportaría exactamente igual que al haber abierto el ejecutable que se encuentra dentro de la carpeta de instalación.



```
Simbolo del sistema - python
Microsoft Windows [Versión 10.0.17134.285]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

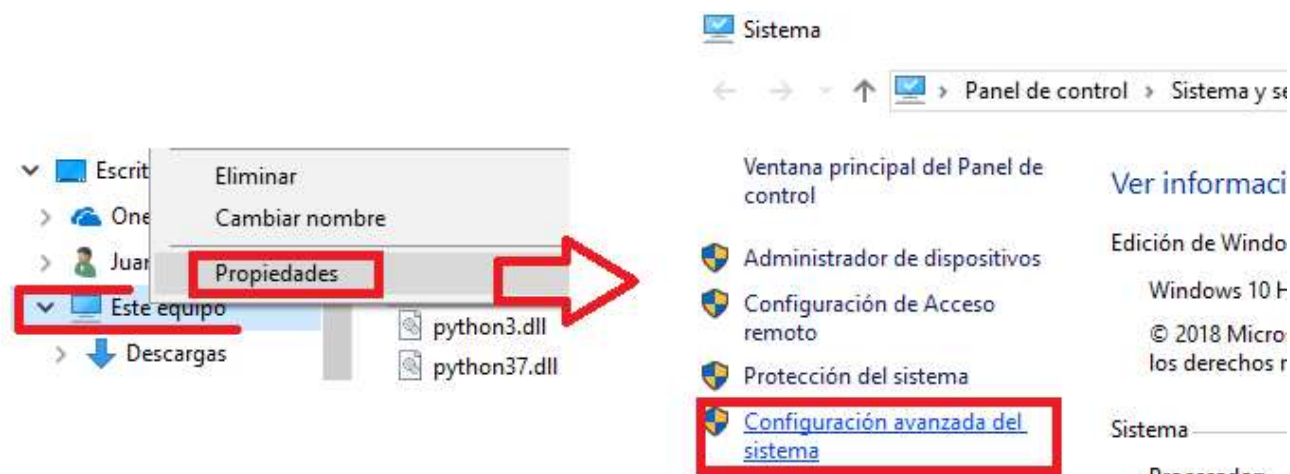
C:\Users\juanb>set path=%path%;C:\python37

C:\Users\juanb>python
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hola")
Hola
```

Lo que hemos realizado fue agregar dinámicamente la versión de python 3.7 a las variables de entorno del sistema operativo para el usuario actual de forma que mientras esta ventana del cmd se encuentre abierta podamos trabajar con ella con python. Si abrimos otra ventana vemos como nuevamente el comando “python” no se reconoce.

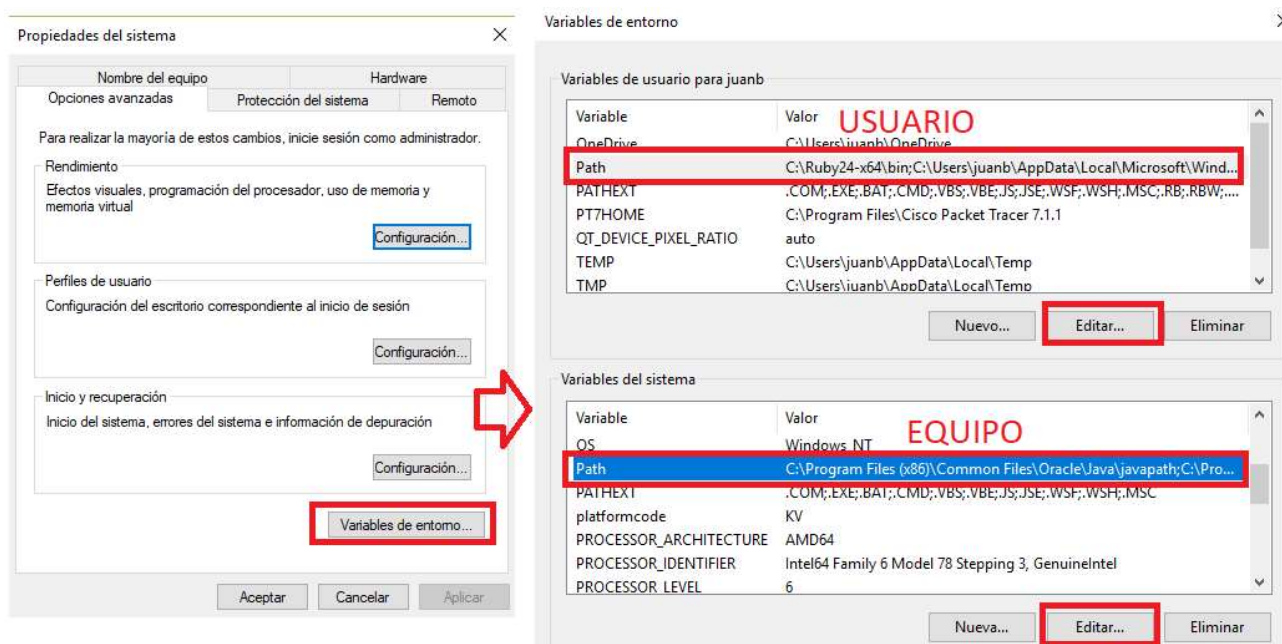
¿Cómo se agrega de forma permanente una versión de python?

Cómo comentamos un poco más arriba, es posible tener varias versiones de python instaladas pero solo hacer accesible una versión desde nuestro sistema operativo, para ello abrimos una carpeta cualquiera en Windows y hacemos click derecho sobre el ícono de “Equipo” y luego seleccionamos “Propiedades”. En la nueva ventana seleccionamos “Configuración avanzada del sistema”





Accedemos a una ventana emergente desde la cual debemos ir a “Variables de entorno” para acceder a las secciones desde las cuales poder agregar a la ruta “Path” mediante el botón “Editar”, la ruta a la versión de python que queremos utilizar ya sea para el usuario actual o para todos los usuarios del equipo.



Nota: Solo la versión de python agregada en la ruta “Path” es utilizada en el sistema operativo, por lo que cuando descargemos paquetes de terceros para extender nuestras funcionalidades de python, estos paquetes se instalarán en la versión que se encuentre indicada en la variable “Path”.

Las rutas que debemos agregar son:

C:\Python37
C:\Python37\Scripts

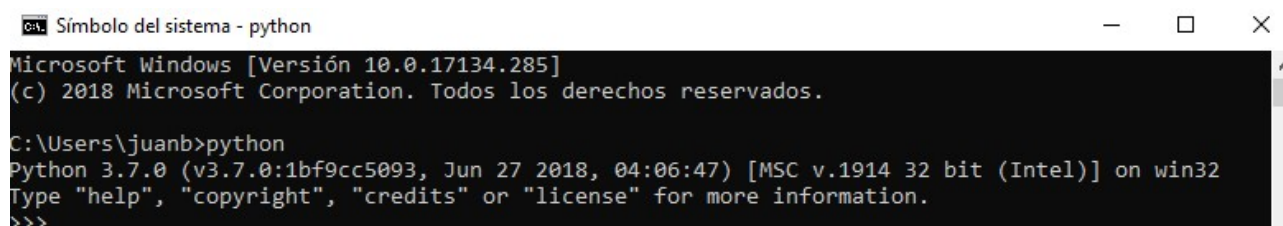
En Windows 10 es más fácil porque podemos agregarlas desde la ventana emergente de a una, sin embargo en versiones previas de Windows todas las rutas venían en una sola línea separadas por punto y coma (;) y debíamos agregarlas al final de la línea de esta forma:

Contenido-del-path;C:\Python37;C:\Python37\Scripts;

Nota: Debemos tener mucho cuidado de no borrar ninguna de las rutas de la variable “Path” porque podríamos generar grandes problemas en la ejecución de nuestro sistema operativo, si creemos que por accidente modificamos algo, debemos cancelar la operación.

1.3. Evaluar si la instalación se realizó correctamente.

Para verificar que todo esté funcionando correctamente, abrimos el cmd y tipeamos “python”. Si todo está bien debemos ver un mensaje como el siguiente:



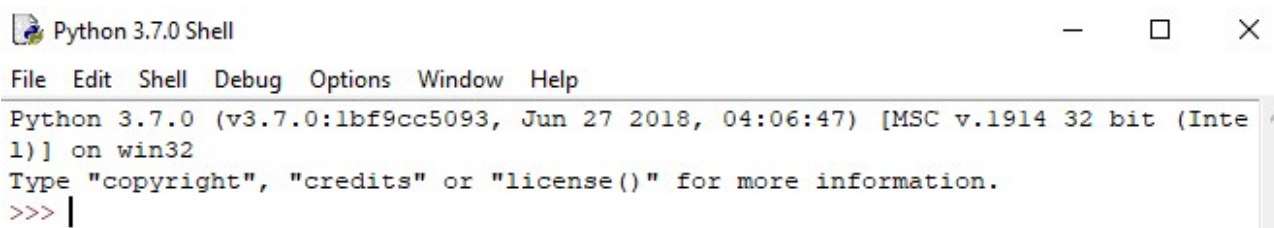
```
Símbolo del sistema - python
Microsoft Windows [Versión 10.0.17134.285]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\juanb>python
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:06:47) [MSC v.1914 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

2. Herramientas Útiles

2.1. Uso del IDLE.

Dentro de las cosas que se han instalado en nuestra máquina, se encuentra el IDLE (Editor de texto que viene por defecto con cada distribución de python), lo abrimos y debería de presentar una apariencia como la de la siguiente imagen:



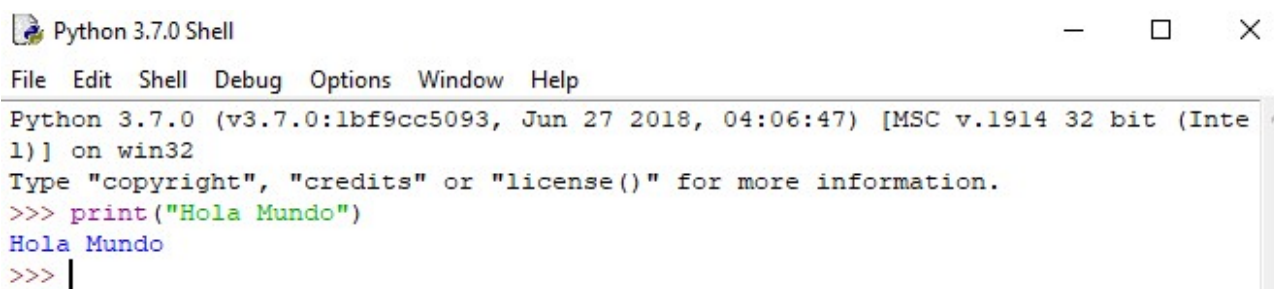
Mediante el IDLE de python podremos evaluar el correcto funcionamiento de nuestros scripts y obtener un detalle de los posibles errores.

Como ejemplo podemos realizar nuestro primer “Hola Mundo” escribiendo:

```
print(“Hola Mundo”)
```

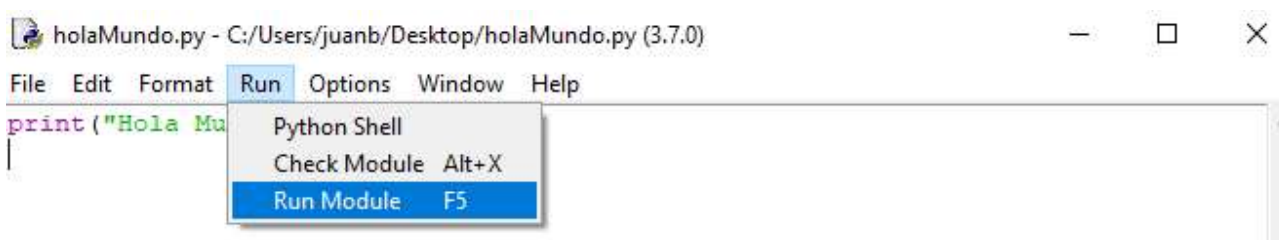
Notar que:

- 1.- En el código, no se agrega punto y como al final de print, como en muchos de los programas actuales (php, javascript,)
- 2.- El IDLE asigna un código de colores, el cual puede ser editable.

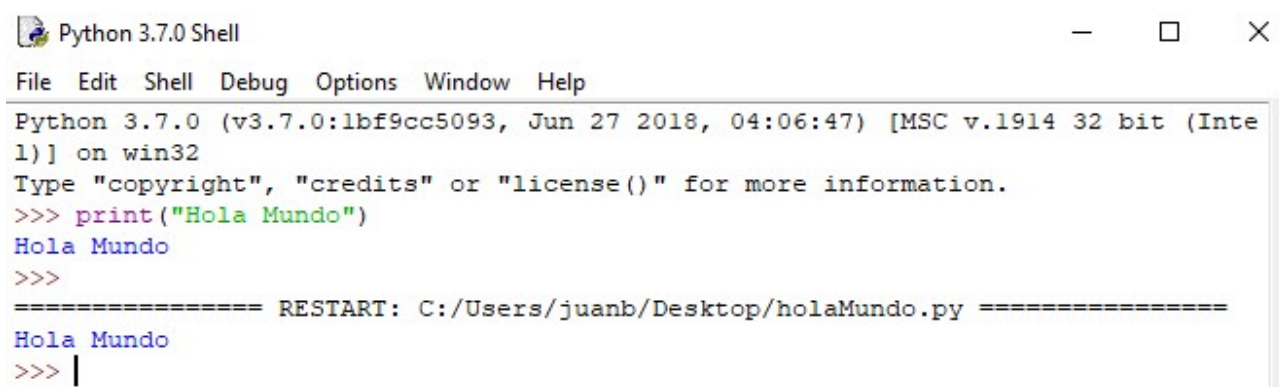


2.1.1. Crear y ejecutar script con el IDLE

Si vamos a File en el margen superior izquierdo del IDLE podemos crear un documento nuevo, al cual podemos salvar con un nombre. Si ahora dentro del archivo generado copiamos nuestro `print("Hola Mundo")`, guardamos el archivo y vamos a Run > Run Module F5. Se abre otra ventana del IDLE que nos muestra el resultado de la ejecución y de existir errores un mensaje con el detalle de los mismos.



Visualización en la nueva ventana:



2.1.2. Recobrar código en el IDLE

En Windows y linux

Alt-P for Previous y Alt-N for Next

En Mac

Ctrl-P y and Ctrl-N.

Centro de e-Learning SCEU UTN - BA.

Medrano 951 2do piso (1179) // Tel. +54 11 4867 7589 / Fax +54 11 4032 0148
www.sceu.frba.utn.edu.ar/e-learning



2.1.3. Ejecutar script en Windows desde el cmd.

Cuando queremos ejecutar un script en windows desde el cmd, debemos pararnos en el directorio en el cual se encuentra y escribir “python nombreArchivo” y este se ejecutará.

Probemos con el ejemplo anterior, yo he guardado en el escritorio el archivo “holaMundo.py” el cual contiene la línea de código:

```
holaMundo.py
```

```
print(“Hola Mundo”)
```

Para ejecutarlo primero abro el cmd, sabiendo que para ingresar a un directorio lo hago con la palabra “cd” seguida del nombre del directorio y que para ir hacia atrás se utiliza “..”. Dado que en mi caso el cmd se abre desde el usuario actual y que el archivo se encuentra en el escritorio, escribo:

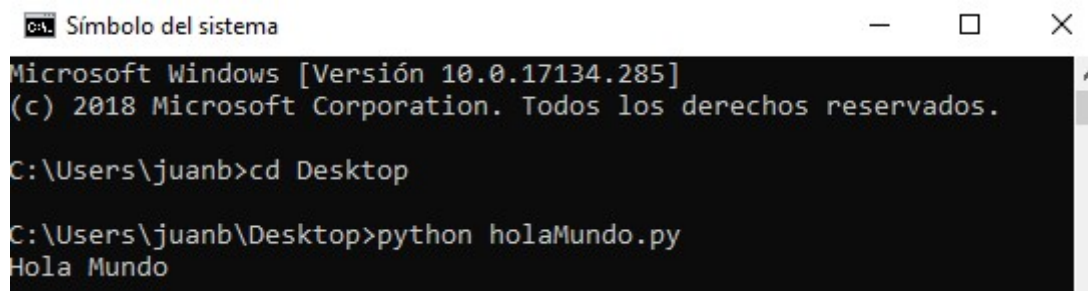
```
cd Desktop
```

```
Símbolo del sistema
Microsoft Windows [Versión 10.0.17134.285]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.
C:\Users\juanb>cd Desktop
C:\Users\juanb\Desktop>
```

Con la instrucción anterior ingreso al escritorio y ahora ejecuto:

```
python holaMundo.py
```

Podemos ver cómo nos aparece en pantalla el resultado de la ejecución.



```
Microsoft Windows [Versión 10.0.17134.285]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\Users\juanb>cd Desktop

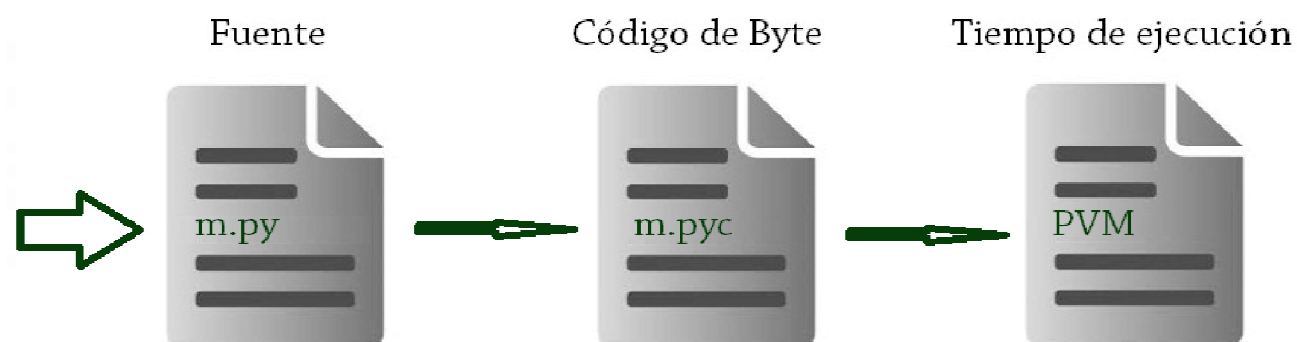
C:\Users\juanb\Desktop>python holaMundo.py
Hola Mundo
```



3. Como funciona Python

Python es tanto un lenguaje compilado como interpretado, en el cual:

- 1.- Python compila el código, o sea lo pasa a código de máquina, permitiendo que se ejecute más rápido.
- 2.- El código compilado se presenta con extensión .pyc con la estructura `_pycache_subdirectorio`. Una vez que se genere el archivo .pyc si no existe modificaciones se ejecuta este archivo en lugar del .py
- 3.- El intérprete luego interpreta el código de byte línea por línea.



Programming Python 5th Edition – Mark Lutz – O'Reilly 2013

NOTA 1: A diferencia de C o C++ el código corre inmediatamente luego de que es escrito, no existe la etapa de construcción.

NOTA 2: En Python la PVM (Python Virtual Machin), no el chip del CPU es el que realiza el papel de Intérprete por lo que Python no funciona tan rápido como C o C++. Por otro lado Python no re analiza cada línea de código, el efecto es que corre a una velocidad intermedia entre los lenguajes compilados y los interpretados.



4. Variables y comentarios.

Variable

Una variable se puede considerar como un símbolo que puede ser reemplazado o que toma un valor determinado, como puede ser un valor numérico en una ecuación o expresión matemática en general.

Las variables se pueden utilizar para guardar datos de diferentes tipos, por ejemplo: enteros, caracteres, listas, arrays, diccionarios, objetos, etc. Todos estos términos seguramente son desconocidos a esta altura, e iremos hablando de ellos en el transcurso de las unidades. De esta forma si queremos hacer que una variable se encuentre relacionada con un valor entero podríamos escribir algo como:

```
variable1 = 7
```

En el ejemplo anterior el nombre de la variable es “variable1” y el valor que toma la variable es el entero = 7.

Pero si el 7 lo ponemos entre comillas simples o dobles en este caso la variable1 estaría relacionada con el carácter “7” o como lo trabajaremos en python el string (alfanuméricos) “7”.

```
variable1 = “7”
```

Comentarios

En python podemos adicionar a nuestro código comentarios que podemos escribir en una línea anteponiendo el símbolo de numeral “#” o en varias líneas entre comillas triples como se muestra a continuación.

```
# Esto es un comentario
"""
Hola curso,
Esto también es un comentario pero multilínea.
"""
```

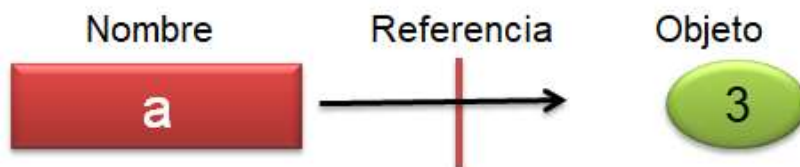


5. Asignación dinámica, garbage collection y referencias compartidas.

5.1. Asignación dinámica.

Cada vez que creamos una variable:

- 1 – Se genera un registro en una tabla
- 2 – Se crea un objeto
- 3 – Se establece la ruta desde la variable al objeto



El objeto creado posee:

- 1 – Un indicador de tipo de objeto.
- 2 – Un contador de referencia para establecer cuándo se puede reclamar el objeto.

Ejemplo:

La variable "a" cambia de tipo, de entero a cadena de caracteres y luego a un número flotante.

```
a = 3          # Es un entero
a = 'Manzana' # Ahora es un string
a = 1.23       # Ahora es un número flotante
```

En este caso la variable no tiene tipo, ya que este es un dato asociado al objeto , “a” simplemente está referenciando a diferentes objetos. Lo único que podemos decir de una variable es que ésta hace referencia a un determinado objeto en un determinado espacio de tiempo.



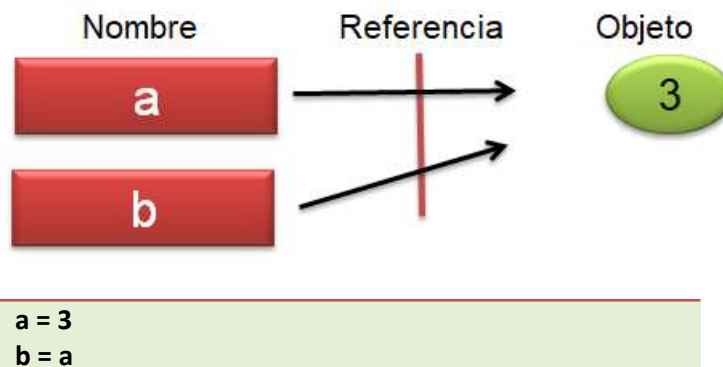
5.2. Garbage collection (Cuando se destruye un objeto)

En Python en el momento que un nombre (variable) es asignado a un nuevo objeto, el espacio de memoria ocupado por el nuevo objeto es reclamado si éste no es referenciado por ningún otro nombre u objeto.

```
x = 42
x = 'juan'      # Reclama 42
x = 3.1415      # Reclama juan
x = [1, 2, 3]   # Reclama 3.1415
```

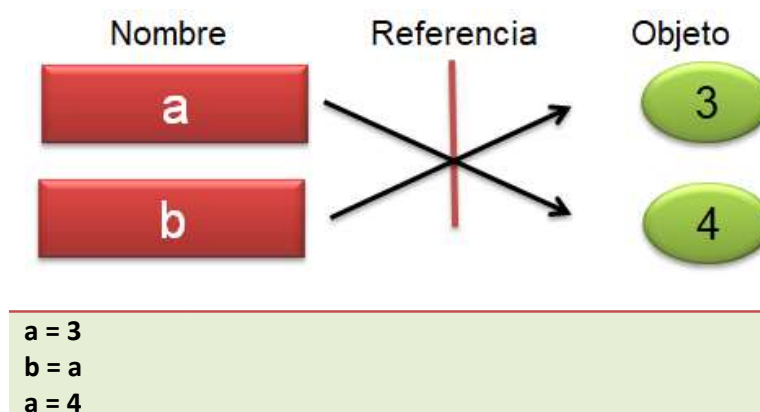
5.3. Referencias compartidas

En este caso ambos nombres de variables, tanto “a” como “b” se encuentran asociados al mismo espacio de memoria



En Python, no hay forma de hacer que una variable haga referencia a otra variable, en lugar de esto la nueva variable, hace referencia al mismo objeto.

Ojo, pues según esto al modificar el valor de a, b sigue haciendo referencia al objeto asignado antes



A diferencia de otros lenguajes, en **Python las variables señalan siempre a objetos**, no a espacios de memoria. Al setear una variable con un nuevo valor el objeto original no es alterado, sino que hace referencia a un nuevo objeto.

Con este punto debemos tener especial cuidado y lo desarrollaremos un poco más en la siguiente unidad ya que existen situaciones que nos pueden conducir a un error de interpretación, al trabajar con los tipos de datos “Listas” y “Diccionarios”.

La siguiente tabla presenta algunos de los tipos de objetos que comenzaremos a analizar, debemos prestar especial atención a la primera columna en donde se indica si el objeto es inmutable o no. En la siguiente unidad desarrollaremos este tema, hasta aquí solo diremos que lo dicho antes se cumple para objetos del tipo inmutables.

Inmutalbe/N o Inmutable	Tipos de Objetos	Ejemplo
Inmutable	Números	1234, 3.14, 3+4j, 0b111, Decimal()
Inmutable	Strings	'Pera', b'a\x01c'
No inmutable	Listas	[1, ['tres', 'dos']], list(range(10))
No inmutable	Diccionarios	{'nombre': 'Pedro', 'edad':23, }, dict(horas=10)
Inmutable	Tuplas	(1, 'pera', 4), tuple('pera')
	Archivos	open('archivo.txt')
No inmutable	Sets	set('abc'), {'a', 'b', 'c'}
	Otros	Booleans, types, None