

Métodos de minería de datos en
Python

Programación básica en Python

Contenido

1

¿Qué es Python?

2

Tipos de datos

3

Operadores básicos

4

Type casting

5

Variables

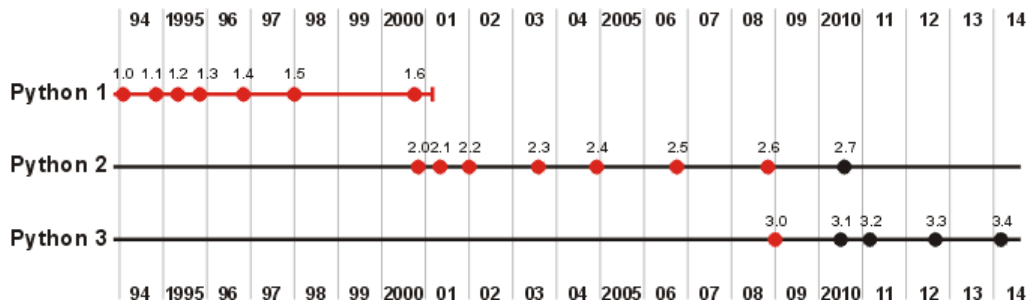
¿Qué es Python?

Python es un lenguaje de programación interpretado cuya principal filosofía es que sea legible por cualquier persona con conocimientos básicos de programación.



De Doc Searls - 2006oscon_203.JPG, CC
BY-SA 2.0,
<https://commons.wikimedia.org/w/index.php?curid=4974869>

Guido Van Rossum, el creador de Python 1991



Filosofía de Python

- Bello es mejor que feo.
- Explícito es mejor que implícito.
- Simple es mejor que complejo.
- Complejo es mejor que complicado.
- Plano es mejor que anidado.
- Disperso es mejor que denso.
- La legibilidad cuenta.
- Los casos especiales no son tan especiales como para quebrantar las reglas.
- Lo práctico gana a lo puro.
- Los errores nunca deberían dejarse pasar silenciosamente.
- A menos que hayan sido silenciados explícitamente.

Filosofía de Python

- Frente a la ambigüedad, rechaza la tentación de adivinar.
- Debería haber una —y preferiblemente solo una— manera obvia de hacerlo.
- Aunque esa manera puede no ser obvia al principio a menos que usted sea holandés.²³
- Ahora es mejor que nunca.
- Aunque *nunca* es a menudo mejor que *ya mismo*.
- Si la implementación es difícil de explicar, es una mala idea.
- Si la implementación es fácil de explicar, puede que sea una buena idea.
- Los espacios de nombres (*namespaces*) son una gran idea

¡Hagamos más de esas cosas!

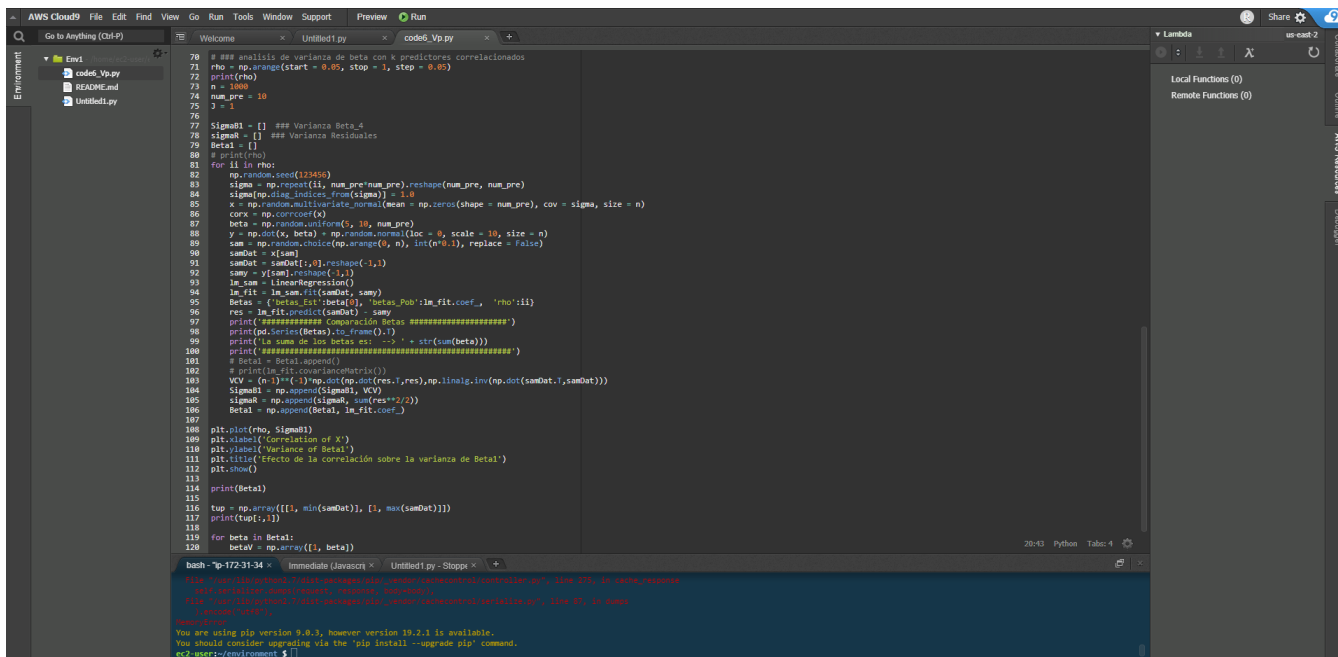
IDE's

IDE es un paquete de software que utilizan los desarrolladores al crear programas.

Integra múltiples componentes altamente relacionados a través de una interfaz de usuario simple para maximizar la productividad del programador.

Básicamente, un IDE es una herramienta que mejora el proceso de creación, prueba y depuración de código, y facilita estas tareas.



[illegible]

IDE's

```
C:\Users\robert.santoso\Documents\Robert\Santoso\Mineral\code\l_Vip.py - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

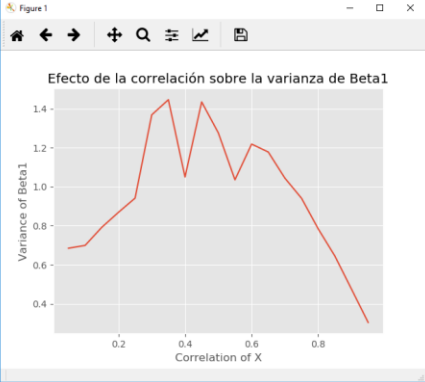
n = 1000
num_pre = 10
J = 1

SigmaB1 = [] ### Varianza Beta_4
sigmaR = []  ## Varianza Residuales
Beta1 = []
# print(rho)
for ii in rho:
    np.random.seed(123456)
    sigma = np.repeat(ii, num_pre*num
    sigma[np.diag_indices_from(sigma)
    x = np.random.multivariate_normal
    corx = np.corrcoef(x)
    beta = np.random.uniform(5, 10, n
    y = np.dot(x, beta) + np.random.n
    sam = np.random.choice(np.arange(
    samDat = x[sam]
    samDat = samDat[:,0].reshape(-1,1
    samy = y[sam].reshape(-1,1)
    lm_sam = LinearRegression()
    lm_fit = lm_sam.fit(samDat, samy)
    Betas = {'betas_Est':beta[0], 'betas_POD':lm_fit.coef_, 'rho':ii}
    res = lm_fit.predict(samDat) - samy
    print('##### Comparación Betas #####')
    print(pd.Series(Betas).to_frame().T)

La suma de los betas es: --> 70.912088202
#####
```

Figure 1

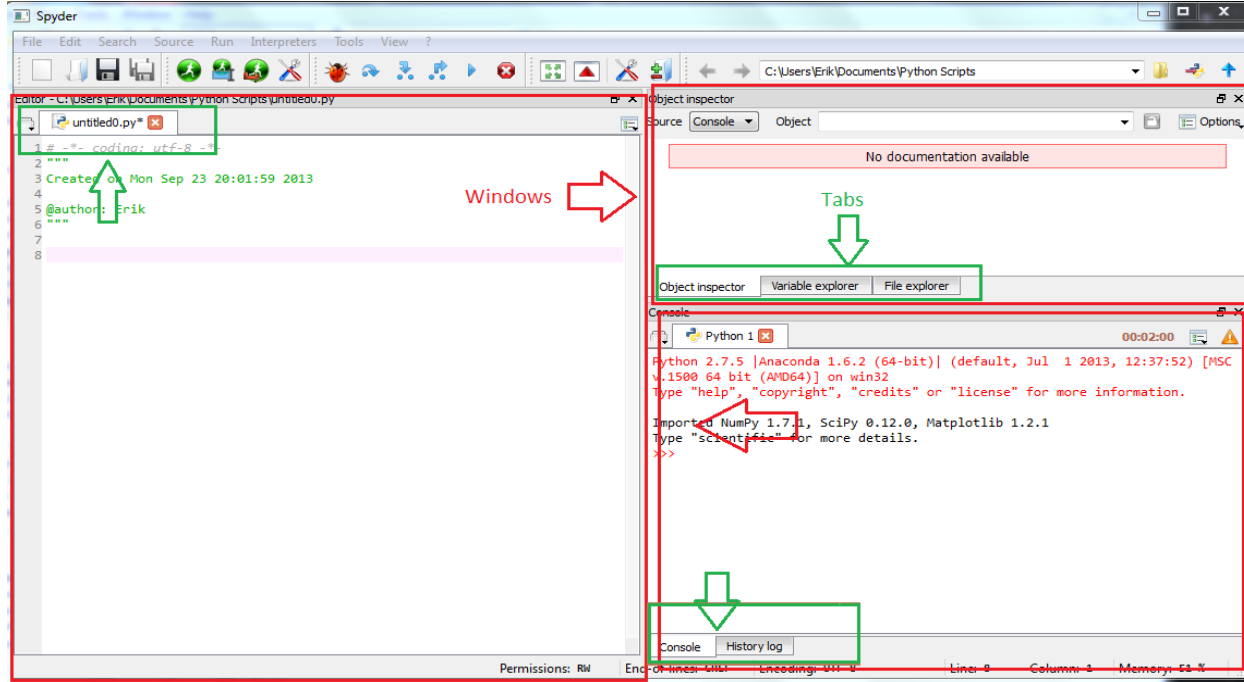
Efecto de la correlación sobre la varianza de Beta1



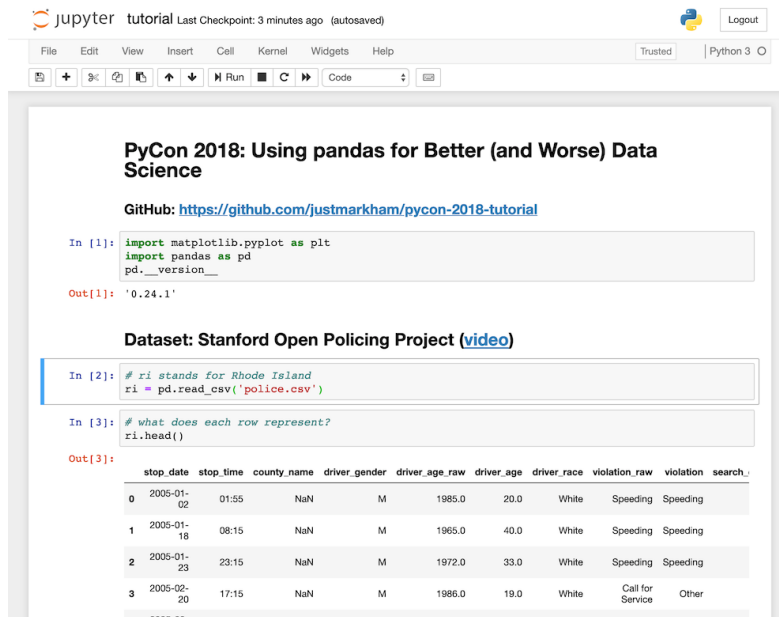
sigma, size = n)


Line 91, Column 39 Tab Size: 4 Python

IDE's



IDE's



jupyter tutorial Last Checkpoint: 3 minutes ago (autosaved)  Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

PyCon 2018: Using pandas for Better (and Worse) Data Science

GitHub: <https://github.com/justmarkham/pycon-2018-tutorial>

```
In [1]: import matplotlib.pyplot as plt
import pandas as pd
pd.__version__
```

```
Out[1]: '0.24.1'
```

Dataset: Stanford Open Policing Project ([video](#))

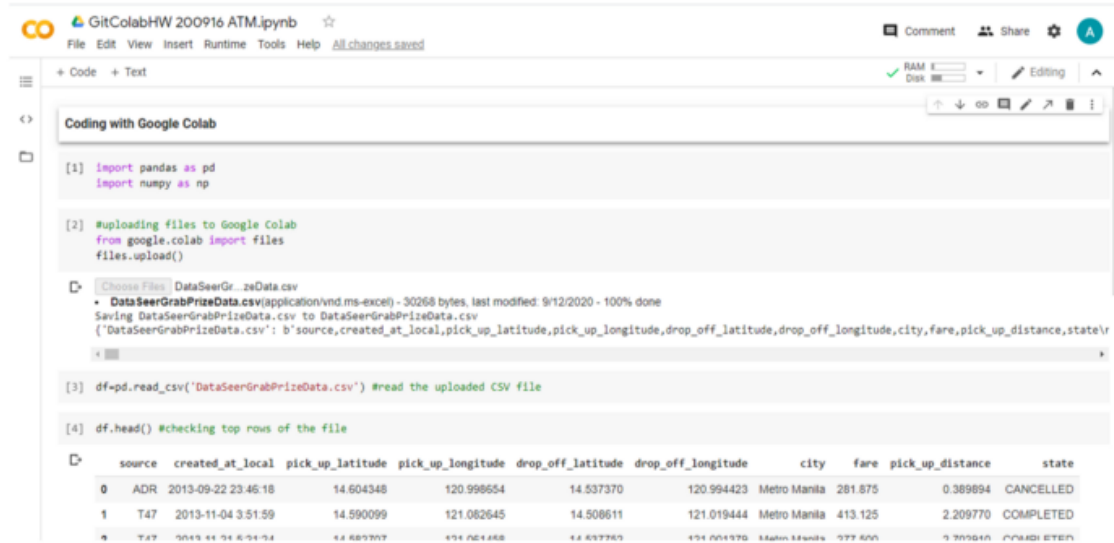
```
In [2]: # ri stands for Rhode Island
ri = pd.read_csv('police.csv')
```

```
In [3]: # what does each row represent?
ri.head()
```

```
Out[3]:
```

	stop_date	stop_time	county_name	driver_gender	driver_age_raw	driver_age	driver_race	violation_raw	violation	search...
0	2005-01-02	01:55	NaN	M	1985.0	20.0	White	Speeding	Speeding	
1	2005-01-18	08:15	NaN	M	1965.0	40.0	White	Speeding	Speeding	
2	2005-01-23	23:15	NaN	M	1972.0	33.0	White	Speeding	Speeding	
3	2005-02-20	17:15	NaN	M	1986.0	19.0	White	Call for Service	Other	

IDE's



The screenshot displays a Google Colab interface with a Jupyter Notebook titled "GitColabHW 200916 ATM.ipynb". The notebook contains four code cells. The first cell imports pandas and numpy. The second cell uploads a file named "DataSeerGrabPrizeData.csv". The third cell reads the CSV file into a DataFrame. The fourth cell displays the first few rows of the DataFrame using the head() method.

```
[1] import pandas as pd
import numpy as np

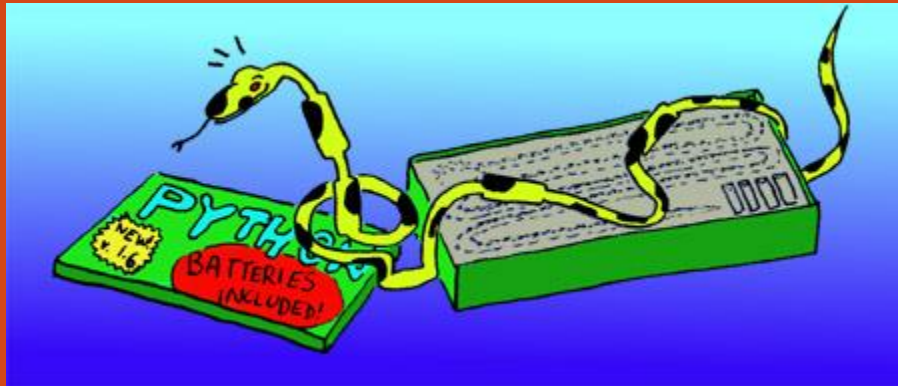
[2] #uploading files to Google Colab
from google.colab import files
files.upload()

[3] df=pd.read_csv('DataSeerGrabPrizeData.csv') #read the uploaded CSV file

[4] df.head() #checking top rows of the file
```

	source	created_at_local	pick_up_latitude	pick_up_longitude	drop_off_latitude	drop_off_longitude	city	fare	pick_up_distance	state
0	ADR	2013-09-22 23:46:18	14.604348	120.998654	14.537370	120.994423	Metro Manila	281.875	0.389894	CANCELLED
1	T47	2013-11-04 3:51:59	14.590099	121.082645	14.508611	121.019444	Metro Manila	413.125	2.209770	COMPLETED
2	T47	2013-11-04 3:51:59	14.604348	120.998654	14.537370	120.994423	Metro Manila	281.875	0.389894	CANCELLED

Manos a la obra





Tipos de datos

Tipos de datos



- El lenguaje provee **representaciones** para **conjuntos de valores**
- Les llamamos **tipos de datos**

Tipos básicos o primitivos

- Tipos numéricos: **int**, **float**
- Tipos de texto: **str**
- Tipos lógicos (booleanos): **bool**

Tipos de datos

- `int` representa números enteros

2 33 0 42
2017 -139 1 -89 17948141

- El año actual
- La edad exacta
- El número de hijos de una persona
- El estato socioeconómico
- ...



`type(42)`



`int`



`type(-5)`



`int`



`type(179000)`



`int`



`type(0)`



`int`

Tipos de datos

- **float** representa números con punto decimal

0.5 1.25 12.
3.933333333 5.0 23.01
-42.3899

- Promedio acumulado de notas
- Saldo de la cuenta
- Tiempo de vuelta rápida en el circuito de Monza
- Temperatura promedio de la tierra el último año
- ...



type(1.25)



float



type(-5.0)



float



type(12.)



float



type(0.5)



float

Tipos de datos

- **str** representa secuencias o cadenas (*string*) de caracteres.

```
"J"    "4"    "¿2+2 = 5?"  
"Cristian" 'Tengo 10 amigos' '-3.0'  
"Aprendo a programar" "98394255"
```

- Nombre y apellidos
- Nombre de la materia
- Número de cédula
- Número telefónico
- ...



type("a")



str



type("Minería")



str



type("155")



str



type(178)



int

Tipos de datos

- **bool** representa valores *booleanos* o de lógica binaria: **Verdadero** o **Falso**

True False



George Boole
1815 - 1864

https://es.wikipedia.org/wiki/George_Boole

- Hoy es lunes?
- Tengo \$ 20.000 en el bolsillo?
- El valor pedido es negativo?
- El valor es mayor a 100?
- ...



type(True)



bool



type(False)



bool



type("True")



str



type(false)



NameError



Operadores básicos

Operadores básicos



Objetivo: Efectuar operaciones con los datos

- Necesitamos expresar **operaciones**
- Utilizamos símbolos: **operadores**
- Expresamos cálculos: **expresiones**

Operadores básicos



Objetivo: Efectuar operaciones con los datos

- Necesitamos expresar **operaciones**
- Utilizamos símbolos: **operadores**
- Expresamos cálculos: **expresiones**

Operadores básicos

- Operadores sobre **int** y **float**

+

Suma

-

Resta

*

Multiplicación

/

División



7+5



12



7-5



2



7*5



35



7/5



1.4

Operadores básicos

- Operadores sobre **int** y **float**

-	**	//	%
Inverso aditivo	Exponenciación	División entera	Módulo



-5



-5



7**5



16807



7//5



1



7%5



2

Operadores básicos

$(3+5//4-2)-2**4+3*(7-2)?$

$2**3**2?$

- Expresiones con más de un operador se evalúan por **precedencia**
- Operaciones con igual precedencia se resuelven por orden de **asociatividad**

Operador	Preced.	Asociatividad	Ejemplo	Resultado
**	1	Derecha a izquierda	$2**3**2$	512
+, - (unarios)	2		$-2**2$	-4
*, /, //, %	3	Izquierda a derecha	$15/3*2$	10
+, - (binarios)	4	Izquierda a derecha	$3-4+5$	4

Operadores para tipos lógicos

- Se aplican a **int** o **float**
 < <= > >= != ==
- Siempre entregan un tipo **bool**



7 < 4.5



False



8 >= 4



True



8 != 6



True



6 == 7



False

Operadores para tipos booleanos

- Se aplican a **bool**
- Siempre entregan un tipo **bool**



not 7 < 4.5



True



3 > 5 **and** 2 < 6



False



3 > 5 **or** 2 < 6



True



5 > 3 **and** 2 < 6



True

Operadores para tipos booleanos

$5//4 > 3$ **or** $2 < 5**2$?
 $3 < 4 \leq 4 < 5$?

Operador	Asociatividad	Ejemplo	Resultado
**	Derecha a izquierda	$2**3**2$	512
+, - (unarios)		$-2**2$	-4
*, /, //, %	Izquierda a derecha	$15/3*2$	10
+, - (binarios)	Izquierda a derecha	$3-4+5$	4
<, <=, >, >=, !=, ==	Izquierda a derecha	$3<4\leq4<5$	True
not		not not 5>2	True
and	Izquierda a derecha	not True and False	False
or	Izquierda a derecha	True or True and False	True

Operadores para tipos de texto

- Operadores para **str**

+
Concatenación

Repetición



"está chévere"
+ "la clase"



"está chévere
la clase"



"Ja, " * 4



"Ja, Ja, Ja,
Ja,"



"Echaron " +
"a" + "Messi"



"Echaron a
Messi"



"JA" * 3 + "porque"
+ "se" + "ríen"



"JAJAJA porque
se ríen"

Taller: Operaciones básicas



`100 + 25`



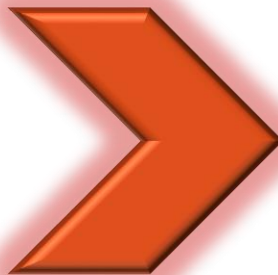
`13 * 13`



`2 ** 12`



`"Hola a todos"`



`125`



`169`



`1024`



`'Hola a todos'`

Taller: Operaciones básicas



"Py" + "thon"



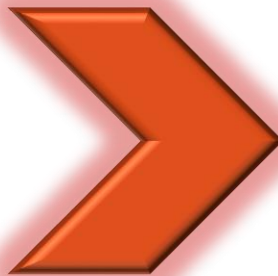
'Py'*2+"thon"



"Py" - "Py"



"ABC" - "AD"



'Python'



'PyPython'



TypeError



TypeError

Taller: Operaciones básicas



`25 + "25"`



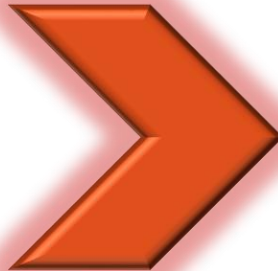
`5 * "5"`



`abc`



`abc = 123`



unsupported
operand type



`'55555'`



`'abc'` is not
defined



Taller: Operaciones básicas



`1 == 1`



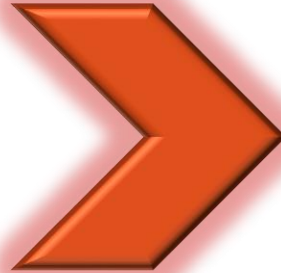
`1 == True`



`0 == False`



`True != False`



`True`



`True`



`True`



`True`

Taller: Operaciones básicas



`100 == 10*10`



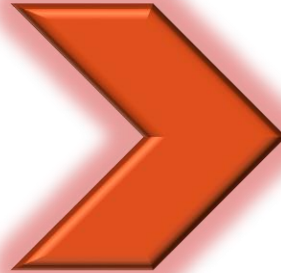
`(100 == 10)*10`



`(5==5)*4+5==1`



`2**5>=4**4`



`True`



`0`



`False`



`False`

Taller: Operaciones básicas



`5 / 2`



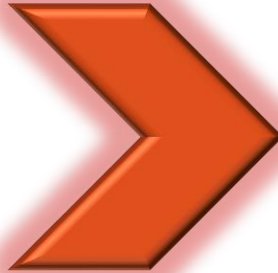
`10 % 3`



`5 // 2`



`5.0 // 2`



`2.5`



`1`



`2`



`2.0`

Taller: Operaciones básicas



```
2000**200
```



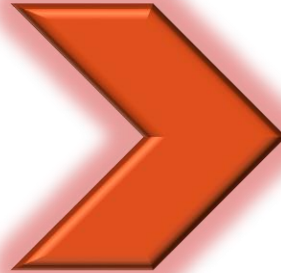
```
2000.5**200
```



```
1.0 + 1.0 - 1.0
```



```
1.0+1.0e20-  
0e20
```



```
1606938044...
```



```
OverflowError
```



```
1.0
```



```
0.0
```

Taller: Operaciones básicas



$3 * 8.73$



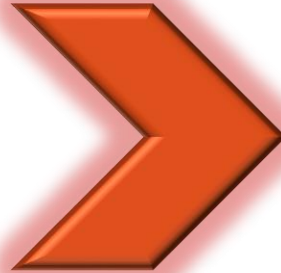
$3 / 8.73$



$3 + 8.73$



"Son las " + (2 + 13) + "pm."



26.19

0.3436426...

11.73

TypeError: must be str, not int

Taller: Operaciones básicas



^{int} 3 * ^{float} 8.73



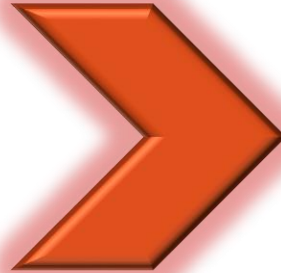
^{int} 3 / ^{float} 8.73



^{int} 3 + ^{float} 8.73



^{str} "Son las " + (^{int} 2 +
13) + ^{str} "pm."



26.19 ^{float}

0.3436426... ^{float}

11.73 ^{float}

TypeError: must be
str, not int

Taller: Operaciones básicas



float float
 $3.0 * 8.73$



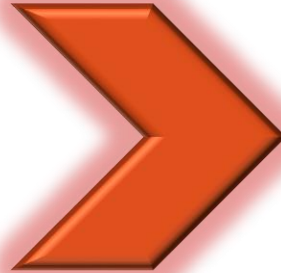
float float
 $3.0 / 8.73$



float float
 $3.0 + 8.73$



str int
"Son las " + (2 +
13) + "pm." str



26.19 float

0.3436426... float

11.73 float

TypeError: must be
str, not int



Type casting

Type Casting

Qué sucede cuando ingresa lo siguiente:



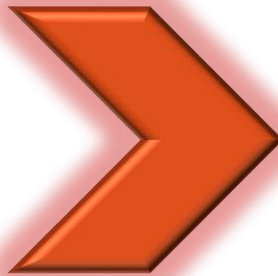
```
str(2 + 13)
```



```
type(str(2 + 13))
```



```
"Son lasstr  
+str(2+13)+str"pm."str
```



"15"

str

"Son las 15pm."

Type Casting

Conversiones a **int**:



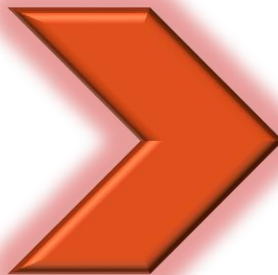
```
int(12.679545)
```



```
int("3") + 12
```



```
int("El 3")
```



12

15

ValueError: invalid literal
for int() with base 10: 'El
3'

Type Casting

Conversiones a **float**:



```
float(3)
```

3.0



```
float("4.5") + 8
```

12.5



```
float("4.5sg")
```

ValueError: could not
convert string to float:
'4.5sg'

Type Casting

Conversiones a **bool**:



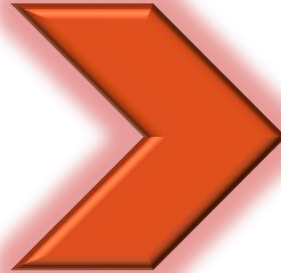
```
bool(0)
```



```
bool("")
```



```
bool("true")
```



False

False

True

Type Casting

Conversiones a **str**:



```
str(5.6)
```

'5.6'



```
str(3 + 1.76)+"  
segundos"
```

'4.76 segundos'



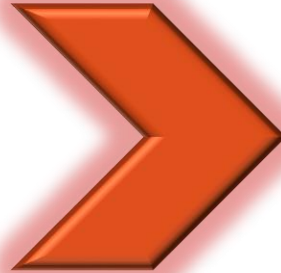


```
str(3<5 and  
3.76 < 10)
```

'True'

Type Casting

Ejercicios:

 `float("123")` `int("123")` `str(123)` `bool("a")` `123.0` `123` `'123'` `True`



Variables

Variables

Las variables son ubicaciones de memoria reservadas para almacenar valores.

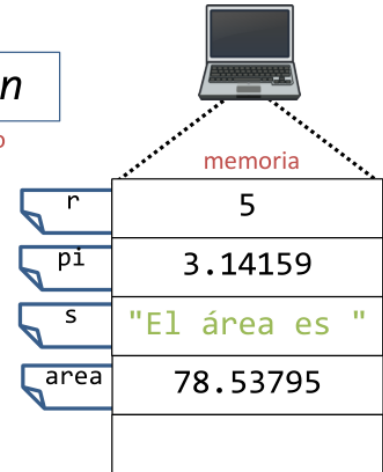
asignación

$nombre = expresión$

lado izquierdo lado derecho

```
>>> r = 5
>>> pi = 3.14159
>>> s = "El área es "
>>> area = pi * r**2
>>> area
```

78.53795



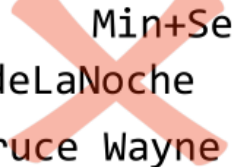
Variables

- **Deben** empezar con una **letra** o '_'
- Puede seguir con letras, números, '_'

pesos_por_hora i km20s
Clark vAlErIa
_CRISTIAN j0rg3



Min+Seg
12deLaNoche
Bruce Wayne

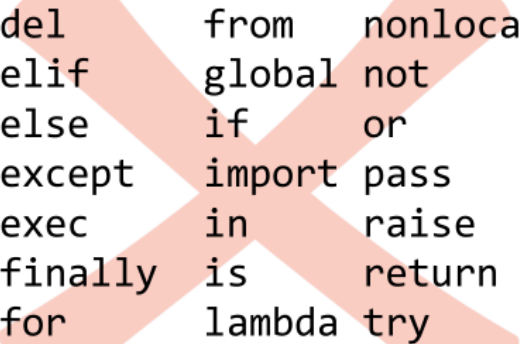


- Las mayúsculas / minúsculas **importan**

Vivaldi ≠ vivaldi ≠ viValdi ≠ VIVALDI

Variables

- Estas palabras **NUNCA** pueden ser usadas



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

Asignación de valores a variables



```
var = 10.0
```



```
a = b = c = 1
```



```
a, b, c = 1, True, "Juan"
```

En Acción!!!

- ❑ **Ejemplo: Crear programa que salude al usuario**

```
[1] nombre = "Tatiana"  
saludo = "Hola,"  
pregunta = "¿Qué tal la clase?"  
print(saludo, nombre, pregunta)
```

Hola, Tatiana ¿Qué tal la clase?

En Acción!!!

- ❑ **Reto:** Crear programa que lo salude a Ud

```
[2] nombre = "Srta. María Paula"  
saludo = "Hola,"  
pregunta = "¿Qué tal la clase?"  
print(saludo, nombre, pregunta)
```

Hola, Srta. María Paula ¿Qué tal la clase?

En Acción!!!



En Acción!!!



obtener datos de entrada

variable = `input(texto)`

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
pregunta = "¿Qué tal la clase?"
print(saludo, nombre, pregunta)
```

¿Cuál es tu nombre?

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
pregunta = "¿Qué tal la clase?"
print(saludo, nombre, pregunta)
```

... ¿Cuál es tu nombre?

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
pregunta = "¿Qué tal la clase?"
print(saludo, nombre, pregunta)
```

¿Cuál es tu nombre?Gabriela
Hola, Gabriela ¿Qué tal la clase?

En Acción!!!

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
saludo2 = ". Espero estés disfrutando la clase"
print(saludo, nombre, saludo2)

lec = input("¿Cuántas semanas llevas de clase?: ")
total = 16
faltan = total - lec
print("Te faltan ", faltan, "semanadas para vacaciones ¡Ánimo!")
```


En Acción!!!

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
saludo2 = ". Espero estés disfrutando la clase"
print(saludo, nombre, saludo2)
```

```
lec = input("¿Cuántas semanas llevas de clase?: ")

¿Cuál es tu nombre?Juan
Hola, Juan . Espero estés disfrutando la clase
¿Cuántas semanas llevas de clase1
```

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
saludo2 = ". Espero estés disfrutando la clase"
print(saludo, nombre, saludo2)
```

```
lec = input("¿Cuántas semanas llevas de clase?: ")
total = 16
faltan = total - lec
print("Te faltan ", faltan, "semanadas para vacaciones ¡Ánimo!")
```

```
¿Cuál es tu nombre?Juan
Hola, Juan . Espero estés disfrutando la clase
¿Cuántas semanas llevas de clase1
```

```
TypeError                                Traceback (most recent call last)
<ipython-input-7-2e98e3958cd0> in <module>()
      6 lec = input("¿Cuántas semanas llevas de clase")
      7 total = 16
----> 8 faltan = total - lec
      9 print("Te faltan ", faltan, "semanadas para vacaciones ¡Ánimo!")
```

```
TypeError: unsupported operand type(s) for -: 'int' and 'str'
```

En Acción!!!

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
saludo2 = ". Espero estés disfrutando la clase"
print(saludo, nombre, saludo2)

lec = input("¿Cuántas semanas llevas de clase?: ")
total = 16
faltan = total - lec
print("Te faltan ", faltan, "semanadas para vacaciones ¡Ánimo!")

¿Cuál es tu nombre?Juan
Hola, Juan . Espero estés disfrutando la clase
¿Cuántas semanas llevas de clase?1
-----
TypeError                                Traceback (most recent call last)
<ipython-input-7-2e98e3958cd0> in <module>()
      6 lec = input("¿Cuántas semanas llevas de clase")
      7 total = 16
----> 8 faltan = total - lec
      9 print("Te faltan ", faltan, "semanadas para vacaciones ¡Ánimo!")

TypeError: unsupported operand type(s) for -: 'int' and 'str'
```

```
print(type(nombre))
```

```
<class 'str'>
```

```
print(type(lec))
```

```
<class 'str'>
```

En Acción!!!

```
nombre = input("¿Cuál es tu nombre?")
saludo = "Hola,"
saludo2 = ". Espero estés disfrutando la clase"
print(saludo, nombre, saludo2)

lec = int(input("¿Cuántas semanas llevas de clase?: "))
total = 16
faltan = total - lec
print("Te faltan ", faltan, "semanadas para vacaciones ¡Ánimo!")
```

```
¿Cuál es tu nombre?Juan
Hola, Juan . Espero estés disfrutando la clase
¿Cuántas semanas llevas de clase?: 1
Te faltan 15 semanas para vacaciones ¡Ánimo!
```

¡Gracias?

¿Preguntas?