



# CURSO PRE QUANT UCEMA

PYTHON PARA FINANZAS QUANT

# INTRODUCCION: POR QUE PYTHON

## Simplificado y rápido

Es un gran lenguaje para scripting, con unas cuantas líneas ya está resuelto.

## Portable, multiplataforma

Es un lenguaje muy portable (Mac, Linux, Windows)

## Open Source

## Comunidad

Googleen una misma pregunta con Python o con R por comparar con cualquiera y vean la diferencia en cantidad de recursos entre ambos

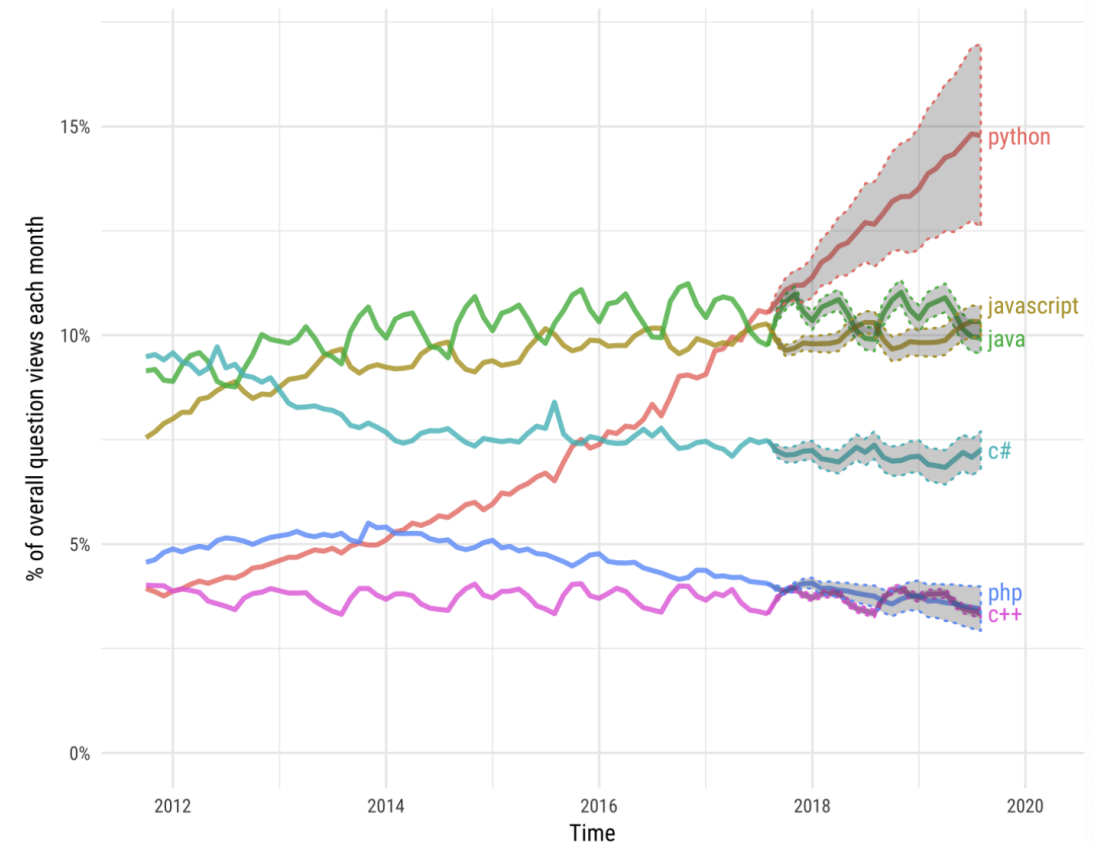
## Librerías

## Curva de Aprendizaje

## Hosting

### Projections of future traffic for major programming languages

Future traffic is predicted with an STL model, along with an 80% prediction interval.



# USOS PRINCIPALES DE PYTHON Y OTROS LENGUAJES

Ejemplo, el lenguaje C es un lenguaje de bajo nivel, es decir, un lenguaje más cercano al hardware, por eso se los llama lenguaje de máquina, por lo tanto es utilizado en proyectos en los que se requiera más cercanía entre el código y el hardware, generalmente proyectos en donde la performance es vital (Por ejemplo HFT)

Ojo que C++ no es lo mismo que C, es una evolución, a su vez C# es una evolución de C++ desarrollada por Microsoft, y que compite con Java

Otro ejemplo de lenguaje muy utilizado es Java que es compilado, es el referente en programación orientada a objetos y muy utilizado para interactuar con todo tipo de dispositivos, es muy usado para apps nativas para mobile por ejemplo o para aplicaciones de escritorio.

En cambio, PHP por ejemplo es el lenguaje del lado del servidor por excelencia para páginas y aplicaciones web, hoy en día se dice que el 80% de la web está hecha en PHP

Sin embargo, Javascript está ganando terreno gracias a NodeJs que permite utilizar este lenguaje para el backend (del lado del servidor)

Python, es un lenguaje de muy alto nivel, es decir muy fácil para escribir código por la gran cantidad de librerías y código prescrito que tienen, su uso más típico es para Inteligencia artificial, machine learning, data science (en donde compite con R) pero también se usa para aplicaciones web en menor medida y es un lenguaje muy versátil y con una curva de aprendizaje muy amigable con lo cual es genial para aprender


## WORLDWIDE PROGRAMMING LANGUAGE STATISTICS



(\*) JavaScript includes CoffeeScript, TypeScript

(\*) The 'least popular' column only includes sectors for which we have data on the language in question.

# ANTE LA DUDA SIEMPRE A LAS FUENTES

 Why GitHub? Team Enterprise Explore Marketplace Pricing  Sign in Sign up

Repositories33K

Code?

Commits158K

Issues55K

DiscussionsBeta0

Packages8

Marketplace0

Topics160

Wikis6K

Users5K

Languages

Python4,711

JavaScript4,193

Java3,313

Ruby2,412

Jupyter Notebook2,293

HTML1,685


PHP1,212


C#1,109


TypeScript950


CSS702


33,245 repository resultsSort: Best match


 7kfpun/FinanceReactNative  
[Deprecated] iOS's Stocks App clone written in React Native for demo purpose (available both iOS and Android).  
android yahoo-finance ios react-native stock  
★ 1.9k JavaScript MIT license Updated on 9 Dec 2019

 yhilpisch/py4fi  
Python for Finance (O'Reilly)  
★ 1.1k Jupyter Notebook Updated on 10 Jul 2018


 lukaszbanasiak/yahoo-finance  
Python module to get stock data from Yahoo! Finance  
python finance yahoo-finance-api  
★ 853 Python Updated on 7 Oct 2018 10 issues need help

 marksweston/finance  
A library for financial calculations in Ruby  
★ 186 Ruby Updated on 25 Mar

 HyperledgerHandsOn/trade-finance-logistics  
Trade Finance and Logistics based on Letter of Credit and Proof of Shipment  
★ 107 JavaScript Apache-2.0 license Updated on 9 Feb

 Hvass-Labs/FinanceOps  
Research in investment finance with Python Notebooks  
finance investing stocks simfin  
★ 197 Jupyter Notebook MIT license Updated 23 days ago

Advanced search Cheat sheet

 Why GitHub? Team Enterprise Explore Marketplace Pricing  Sign in Sign up

Repositories1M

Code?

Commits54M

Issues3M

DiscussionsBeta14

Packages376

Marketplace37

Topics3K

Wikis141K

Users51K

Languages

Java620,803

Kotlin64,637

C31,062

JavaScript19,395

C++19,222

C#12,176


Makefile11,011


Shell8,666


HTML8,121


Python7,983


1,014,815 repository resultsSort: Best match


 Android  
Android is an operating system built by Google designed for mobile devices.  
See topic

 open-android/Android  
GitHub上最火的Android开源项目,所有开源项目都有详细资料和配套视频  
android java  
★ 9.7k Updated on 8 Feb

 hmrcode/Android  
Android related examples  
★ 3k Java Updated on 4 Mar

 owncloud/android  
The ownCloud Android App  
★ 2.9k Java GPL-2.0 license Updated 2 days ago

 cSploit/android  
cSploit - The most complete and advanced IT security professional toolkit on Android.  
★ 2.4k Java GPL-3.0 license Updated on 16 Feb 5 issues need help

 itheima1/Android  
收集Android方方面面的经典知识, 最新技术.  
android java itheima  
★ 2.5k Updated on 30 Apr 2017

Advanced search Cheat sheet

# DOS SITIOS QUE NO PUEDEN DESCONOCER

## ■ GITHUB

Es un sitio de repositorios, es decir como un Google drive específico de archivos de programación, en el fondo funciona medio como una especie de red social, pero la realidad es que no pasa por ahí su utilidad.

Tiene la opción de repositorios públicos y privados, y en ambos es un ambiente de trabajo colaborativo

## ■ STACKOVERFLOW

Es el Google de los programados, tiene un formato espectacular, ya lo van a notar, no hay mejor amigo del programador que stackoverflow

### How do I parse a string to a float or int?

Asked 11 years, 4 months ago · Active 3 months ago · Viewed 3.9m times

2219 In Python, how can I parse a numeric string like "545.2222" to its corresponding float value, 545.2222 ? Or parse the string "31" to an integer, 31 ?

I just want to know how to parse a `float` `str` to a `float`, and (separately) an `int` `str` to an `int`.

python parsing floating-point type-conversion integer

266 share improve this question follow

edited Jun 5 '19 at 15:20  
Georgy  
3,834 ● 5 ● 29 ● 40

asked Dec 19 '08 at 1:52  
Tristan Havelick  
53k ● 18 ● 51 ● 64

6 As a general rule, if you have an object in Python, and want to convert to that type of object, call `type(my_object)` on it. The result can usually be called as a function to do the conversion. For instance `type(100)` results in `int`, so you can call `int(my_object)` to try convert `my_object` to an integer. This doesn't always work, but is a good "first guess" when coding. – robertlayton Jul 5 '18 at 1:18

`int(x) if int(x) == float(x) else float(x)` – tcpaiva Apr 23 at 0:31

add a comment

26 Answers

Active Oldest Votes

2588  

```
>>> a = "545.2222"
>>> float(a)
545.22220000000004
>>> int(float(a))
545
```

share improve this answer follow

edited Feb 5 at 15:38

answered Dec 19 '08 at 1:54  
Harley Holcombe  
139k ● 15 ● 65 ● 62

# PREMISAS BÁSICAS DE LA PROGRAMACIÓN

- 1.Confiabilidad: ¿Resuelve siempre sin fallas? ¿De qué depende? ¿Cuál es el % de fallas?
- 2.Testeabilidad: ¿Hay maneras de probar el programa antes de sacarlo a producción?
- 3.Performance: ¿Se podría hacer que sea más rápido ejecutándose o que use menos recursos de memoria/procesador?
- 4.Usabilidad: ¿Es fácil de usar para un nuevo usuario o para mi mismo?
- 5.Mantenibilidad: ¿Que tan sencillo es realizarle cambios? ¿qué tan prolijo y legible queda el código?
- 6.Escalabilidad: ¿Como se comportaría si crece en cantidad de datos, de usuarios, de instrumentos?
- 7.Portabilidad: ¿Me sirve para otro mercado, instrumento, horario, etc?
- 8.Seguridad: ¿Está protegido de ataques?



# INSTALACIÓN DE ANACONDA

Entren en

<https://www.anaconda.com/products/individual>

Seleccionen la versión acorde a su sistema operativo

Denle a todo que si, no cambien nada, no cambie la ruta de instalación por default

Si tienen un firewall muy estricto apáguenlo un toque para instalar esto, es seguro



The screenshot shows the 'Anaconda Installers' page with three columns for different operating systems. Each column lists installers for Python 3.7 and Python 2.7, with 64-bit and 32-bit graphical installers, and 64-bit command line installers for Linux.

Operating System	Python Version	Installer Type	Size
Windows	Python 3.7	64-Bit Graphical Installer	466 MB
		32-Bit Graphical Installer	423 MB
	Python 2.7	64-Bit Graphical Installer	413 MB
		32-Bit Graphical Installer	356 MB
	Python 3.7	64-Bit Command Line Installer	442 MB
		64-Bit Command Line Installer	430 MB
MacOS	Python 3.7	64-Bit Graphical Installer	442 MB
		64-Bit Command Line Installer	430 MB
Python 2.7	64-Bit Graphical Installer	637 MB	
	64-Bit Command Line Installer	409 MB	
Linux	Python 3.7	64-Bit (x86) Installer	522 MB
		64-Bit (Power8 and Power9) Installer	276 MB
	Python 2.7	64-Bit (x86) Installer	477 MB
		64-Bit (Power8 and Power9) Installer	295 MB
	Python 3.7	64-Bit (x86) Installer	522 MB
		64-Bit (Power8 and Power9) Installer	276 MB

# PRINCIPALES ENTORNOS PARA PROGRAMAR EN PYTHON

- Consola de Python: Es la forma más rápida de probar un script, pero terriblemente incómoda para codear, mas que nada se usa para ejecutar una sola linea
- Jupyter notebooks: Vienen en Anaconda, son piolas para ir probando código línea por línea pero no sirve de mucho para un proyecto en si
- Intérpretes online: Ej repl.it es interesante para cuando no tenemos nuestra compu y queremos trabajar en proyectos chicos
- Editores de código: Serían como un Word pero para codear, los más recomendados son:
  - SublimeText,
  - Atom
- IDEs: Son entornos de desarrollo integrados (integran consola + editor de texto etc), ejemplos:
  - Spyder
  - Pycharm



# VARIABLES

- Que es una variable en programación

Hay que ver a las variables como “cajas” donde se van poniendo cosas, esas “cosas” obviamente tienen “datos” pero no necesariamente son simples datos, a veces son “cosas” un poco mas complejas a las que llamamos objetos

- Tipos de Variables:

- Enteros, flotantes, cadenas, booleanos, nulos:

Estos son los tipos simples como números o textos

- Estructuras nativas de datos: Listas, tuplas, diccionarios, fechas:

Son como “datos múltiples” o algo así

- Objetos no nativos

Son estructuras mas complejas porque no solo tienen “datos” sino que tienen “parametrizaciones” o funciones

- Tipado dinámico

En Python el tipo de una variable puede mutar de una línea a otra simplemente asignándole a la misma un dato de otro tipo

## OJO, A TENER EN CUENTA:

- **Los dos lados del igual son diferentes**

$a=b$  significa algo diferente que  $b=a$

- **Python es case sensitive**

La variable “precio” es diferente que la variable “Precio”

- **Es Indentado**

Los espacios “en blanco” tienen un significado super importante

# FUNCIONES DE CADENAS

- `string.capitalize()` => Pasa a mayúscula la primera letra de la variable string
- `string.upper()` => Pasa todas las letras a mayúsculas de la variable string
- `string.lower()` => Pasa todas las letras a minúsculas de la variable string
- `string.title()` => Pasa las primeras letras de cada palabra a mayuscula de la variable string
- `string.rfind("string_buscado")` => Devuelve la última posición donde aparece por primera vez el `string_buscado` en la variable string
- `string.find("string_buscado")` => Devuelve la primera posición donde aparece por primera vez el `string_buscado` en la variable string
- `string.replace("str_buscado","str_reemplazo")` => Reemplaza "str\_buscado" por "str\_reemplazo" en la variable string
- `string.zfill(largo)` => Rellena con ceros a la izquierda hasta completar el largo pasado como argumento
- `string.count("string_buscado")` => Devuelve la cantidad de veces que aparece el `string_buscado` en la variable string
- `len(string)` => Cuenta la cantidad de caracteres de la variable string
- `string.isalnum()` => True si todos los caracteres son alphanumericos
- `string.isalpha()` => True si todos los caracteres son letras
- `string.isdigit()` => True si todos los caracteres son dígitos

# OPERACIONES MATEMÁTICAS NATIVAS

- Suma +
- Resta -
- Producto \*
- División /
- División entera (cociente) //
- Mod (resto) %
- Potencia \*\*
- Valor absoluto abs()
- Raíz cuadrada trucha 😊 \*\*0.5

# LIBRERÍA MATH, FUNCIONES MAS TÍPICAS

- Constantes: pi, e
- Ceil (techo)
- Floor (piso)
- Trunc (truncado) Ojo, no es lo mismo que “floor” ya que con negativos coincide con el “ceil”
- Sqrt
- Factorial
- Log, log10
- Trigonómicas: cos, sin, tan, acos, asin etc..
- isClose

# LIBRERÍA DATETIME

- Sub librerías de datetime
  - datetime (fecha y hora combinadas)
  - date (solo fechas, sin hora)
  - time (solo horas, sin fecha)
  - timedelta (diferencia entre fechas, o bien, cantidad de tiempo)
  - timezone (zonas horarias)
- Atributos de un Objeto datetime
  - year, month, day, hour, minute, second, tzinfo (estos últimos si los datetime, no los date)

## SUB-LIBRERÍA DATE

- `date(año, mes, día)` >> Crea el objeto fecha
- `today()` >> Crea el objeto con la fecha de hoy
- `isocalendar(date Object)` >> devuelve año, semana del año y día de la semana
- `ctime(date Object)` >> Devuelve un string cómodo de lectura
- `isoformat(date Object)` >> Devuelve un string de fecha con formato ISO del objeto fecha pasado
- `fromisoformat(string)` >> transforma una fecha ISO en un date object
- `strftime(date Object, format=str)` >> devuelve un string desde un objeto date con el formato format
- `strptime(string, format=str)` >> devuelve un objeto date equivalente al string con el formato format pasado

### Paradigma Funcional

Ejemplos de como usar un método de un objeto llamando al método y pasándole argumentos o aplicando el método al objeto

```
hoy = datetime.date.today()
```

```
datetime.date.ctime(hoy) >> 'Sun May 10 00:00:00 2020'
```

```
hoy.ctime() >> 'Sun May 10 00:00:00 2020'
```



# SUB-LIBRERÍA DATETIME Y TIMEDELTA

## **datetime**

- Mismas funciones que date: `ctime().strftime()`, `strptime()`
- `datetime()` en lugar de `date()`, y acepta los argumentos `hour`, `minute`, `second`, `microsecond`, `tzinfo`
- `isoformat(date Object, sep='T', timespec='seconds')` >> Acepta separador, y precisión de horario
- `now()` devuelve lo mismo que `today()` que también es válida
- `timestamp()` >> número entero que representa los segundos desde el 1 Ene 1970
- `isoweekday()` >> 1 Lunes, 2 Martes.. 7 Domingo
- `weekday()` >> Idem pero con el lunes como 0 , y 6 para Domingo
- `replace()` >> reemplaza cualquier valor de un objeto `datetime`

## **Timedelta**

Esta sub-librería de `datetime` nos permite expresar tiempo como objetos

- `total_seconds()`

# FORMATO DE FECHAS DATETIME (STRFTIMEY STRPTIME)

Código	Significado	Ejemplo
%a	Nombre del día abreviado	Mon
%A	Nombre del día de semana, 0=Domingo... 6=sábado	Monday
%d	Numero de día del mes rellenado con 0	08
%-d	Numero de día del mes sin rellenar con 0	8
%b	Nombre del mes abreviado	Jan
%B	Nombre del mes	January
%m	Numero de mes del mes rellenado con 0	08
%-m	Numero de mes del mes sin rellenar con 0	8
%y	Numero año sin siglo	19
%Y	Numero año con siglo	2019
%j	Número de día del año rellenado con ceros	031
%-j	Número de día del año sin ceros	31
%U	Número de semana del año (Domingo=Primer día de semana)	52
%W	Número de semana del año (Lunes=Primer día de semana)	52
%c	Representación completa	Mon Sep 30 07:06:05 2013
%x	mes/día/año	07/24/2020

Código	Significado	Ejemplo
%H	Numero de Hora (0-24) con ceros	09
%-H	Numero de Hora (0-24) sin ceros	9
%I	Numero de Hora (0-12) con ceros	09
%-I	Numero de Hora (0-12) sin ceros	9
%p	AM/PM	AM
%M	Minutos con ceros	09
%-M	Minutos sin ceros	9
%S	Segundos con ceros	05
%f	Microsegundos	123456
%X	Hora:minutos:segundos	20:12:55
%H--	numero de hora, + caracteres	20--

# LIBRERÍAS CALENDAR Y LOCALE

## Métodos mas útiles

- `prcal(año)` ⇒ Imprime en pantalla el calendario anual
- `prmonth(año,mes)` ⇒ Imprime en pantalla el calendario de mes
- `isleap(año)` ⇒ Devuelve True o False si es o no bisiesto
- `leapdays(año_from , año_to)` ⇒ Devuelve la cantidad de días bisiestos entre los años dados
- `weekday()` ⇒ Devuelve el número de día de la semana (0 Lunes 6 Domingo)
- `monthrange()` ⇒ devuelve el número de día del primero del mes, y la cantidad de días del mes
- `day_name[ ]` ⇒ No es una función es un diccionario de nombres
- `month_name[ ]` idem

## Locale

- `setlocale(locale.LC_TIME, "esp")` ⇒ Nombres de días y meses en castellano