



برنامه نویسی با پایتون

سعید محقق / تابستان 1400

- حضور در کلاس
- تمرین و پروژه
- کار روزانه
- آموزش ترکیبی (پروژه محور، توضیح و تئوری، کدنویسی)

1. مقدمه و معرفی

2. کدنویسی پایتون در عمل

1. مقدمه و معرفی

ویژگی ها و کاربردهای پایتون

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

- سطح بالا و بسیار پیرکاربرد
- تاکید بر سادگی و خوانایی کدها
- پایاده سازی و توسعه سریع ایده ها ← مناسب برای کاربردهای آکادمیک
- پایتون cross-platform است: کدهای پایتون، در تمام سیستم عامل های مرسوم قابل اجرا هستند ← یکی از دلایل محبوبیت زیاد پایتون

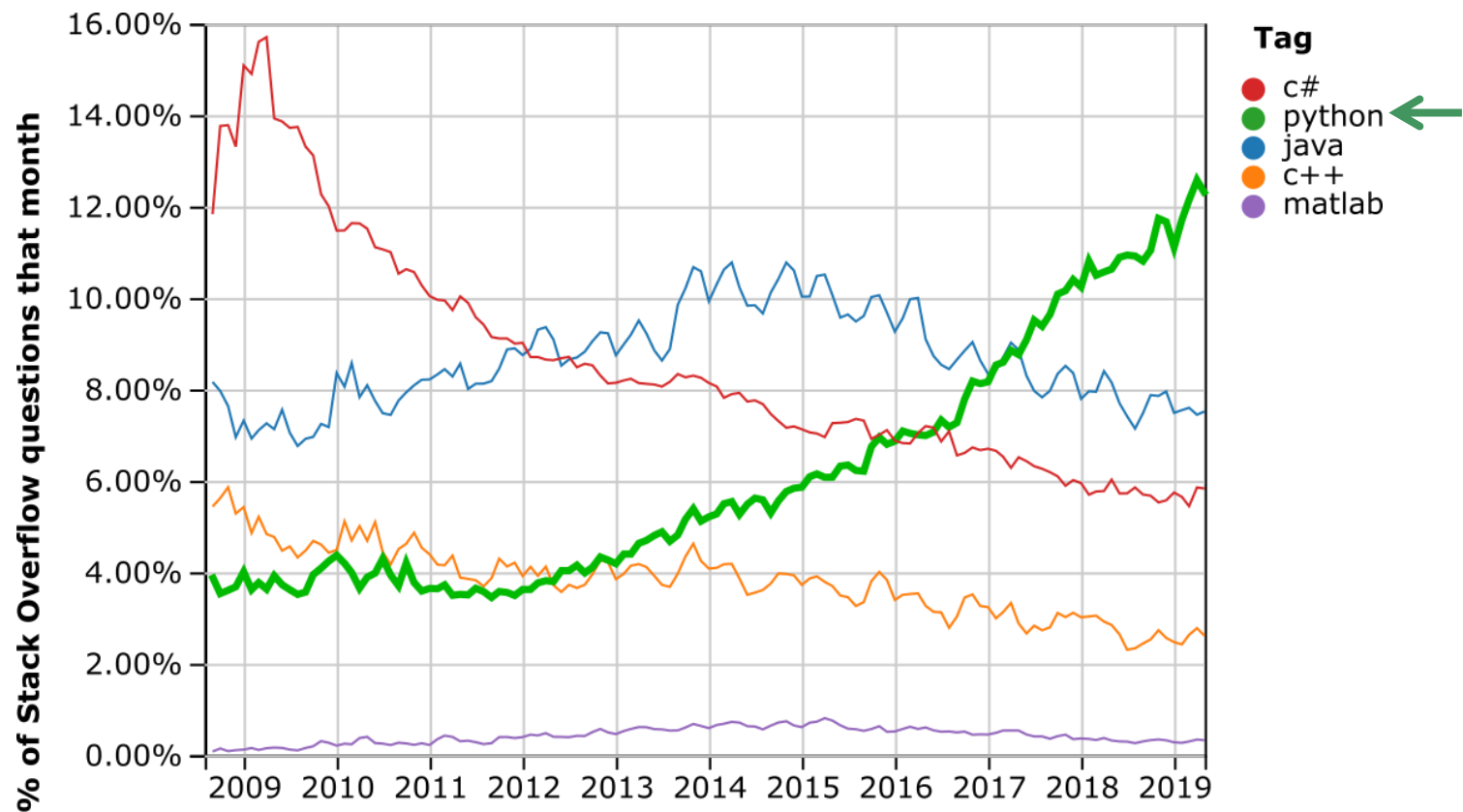
■ نام "پایتون" برگرفته از یک گروه نمایشی کمدی به نام Monty Python (1991)



■ سازنده پایتون (Guido van Rossum)



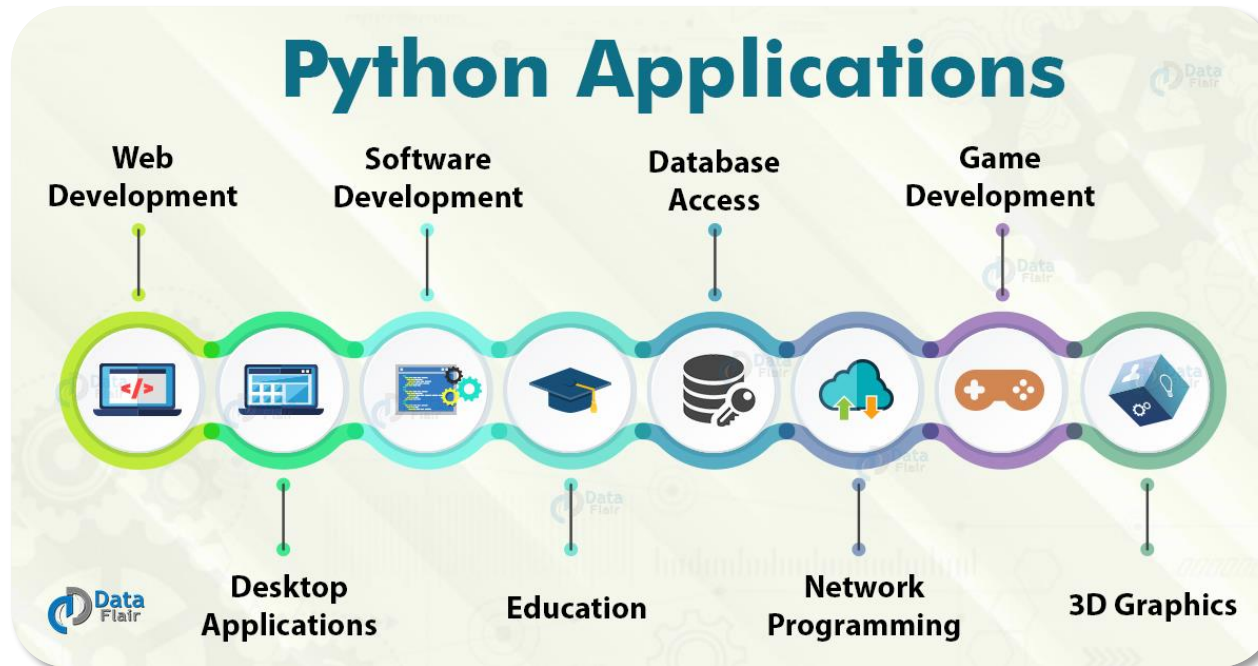
رشد کاربرد و محبوبیت پایتون



رشد کاربرد زبان های برنامه نویسی

در سایت Stack Overflow

insights.stackoverflow.com/trends?tags=java%2Cc%2B%2B%2Cpython%2Cc%23%2Cjavascript%2Cmatlab

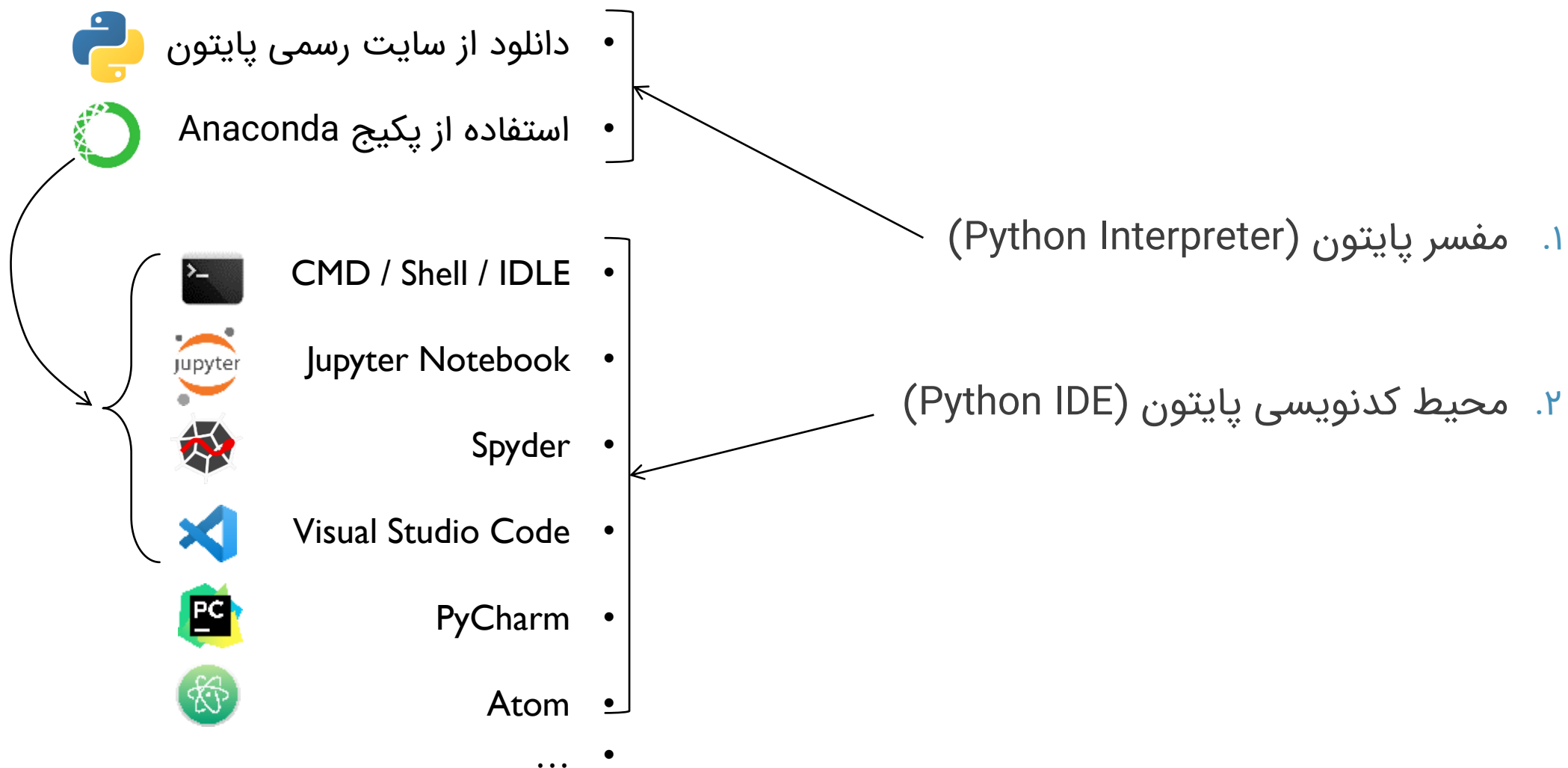


- برنامه نویسی علمی و محاسباتی
- علوم داده و هوش مصنوعی
- برنامه نویسی وب و شبکه
- بازی های کامپیوتری
- نرم افزارهای دسکتاپ و GUI
- آموزش



2. کدنویسی پایتون در عمل

راه اندازی محیط برنامه نویسی و شروع کدنویسی



- در حال حاضر پایتون دارای دو گروه نسخه اصلی است:
- نسخه های 2.x ← پرکاربردترین نسخه ها در گذشته ← پایان پشتیبانی در سال 2020
- نسخه های 3.x ← نسخه اصلی و پیش فرض فعلی

(1) دانلود مفسر پایتون به صورت رایگان از سایت www.python.org (~25 MB)

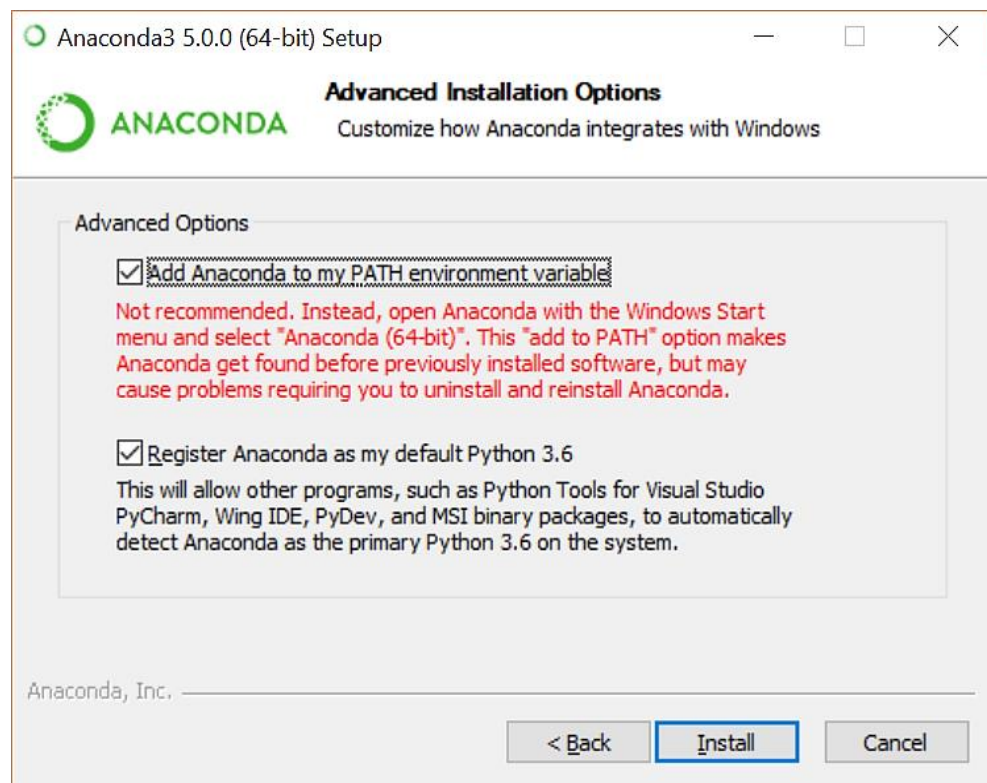
(2) دانلود پکیج Anaconda به صورت رایگان از سایت www.anaconda.com (~500 MB)
(شامل مفسر پایتون + بسیاری از کتابخانه ها + IDE های مختلف)

(3) دانلود پکیج Miniconda به صورت رایگان از سایت
<https://docs.conda.io/en/latest/miniconda.html> (~70 MB)
(شامل مفسر پایتون + قابلیت های Anaconda)

- دانلود و نصب Anaconda برای پایتون ۳

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

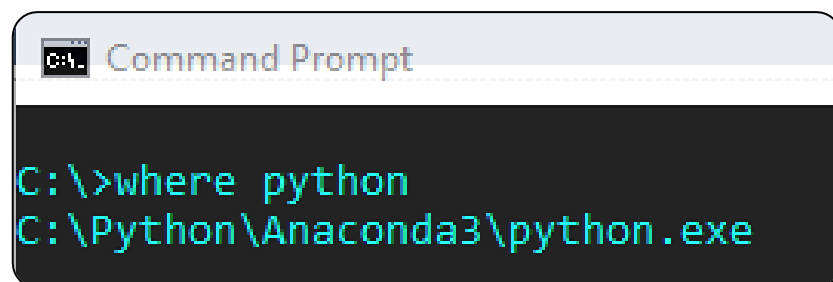
- زدن تیک اضافه شدن به مسیرهای ویندوز در طول نصب ←



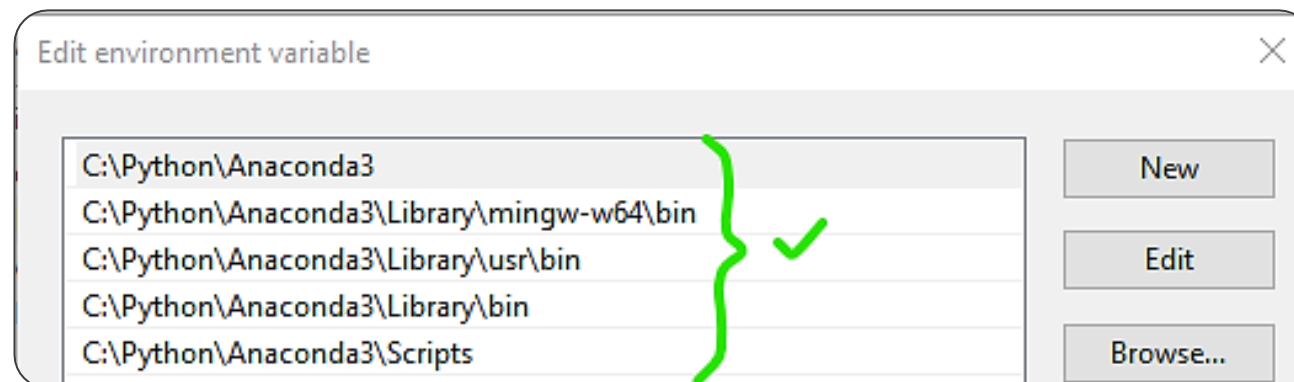
افزودن پایتون به مسیرهای ویندوز

- اگر هنگام نصب Anaconda گزینه افزودن به مسیرهای ویندوز انتخاب شده باشد، مسیرهای پایتون در path پنجره Environment Variables ویندوز وجود خواهند داشت. و با دستور زیر در برنامه CMD می توان مسیر پایتون را نمایش داد:

> where python



```
Command Prompt
C:\>where python
C:\Python\Anaconda3\python.exe
```



- در غیر این صورت، می توان این مسیرها را به صورت دستی، به متغیر path اضافه کرد.

روش ۱:

نوشتن کدها در محیط python در
محیط‌های cli (bash یا cmd)

روش ۲:

نوشتن کدها در یک فایل متنی با پسوند py

اجرای فایل در محیط‌های cli



■ روش ۳:

■ کدنویسی و اجرا در محیط Notebook (آفلاین: Jupyter / آنلاین: Google Colab)



■ روش ۴:

■ کدنویسی و اجرا در IDE ها مانند: IDLE / Spyder / VS Code / PyCharm / ...



■ دستورها

```
>>> print('Hello World')
```

```
>>> import sys
```

```
>>> print(sys.version)
```

<چاپ نسخه پایتون>

■ کامنت ها

```
# One line comment
```

```
'''  
Multiple  
Line  
Comment  
'''
```

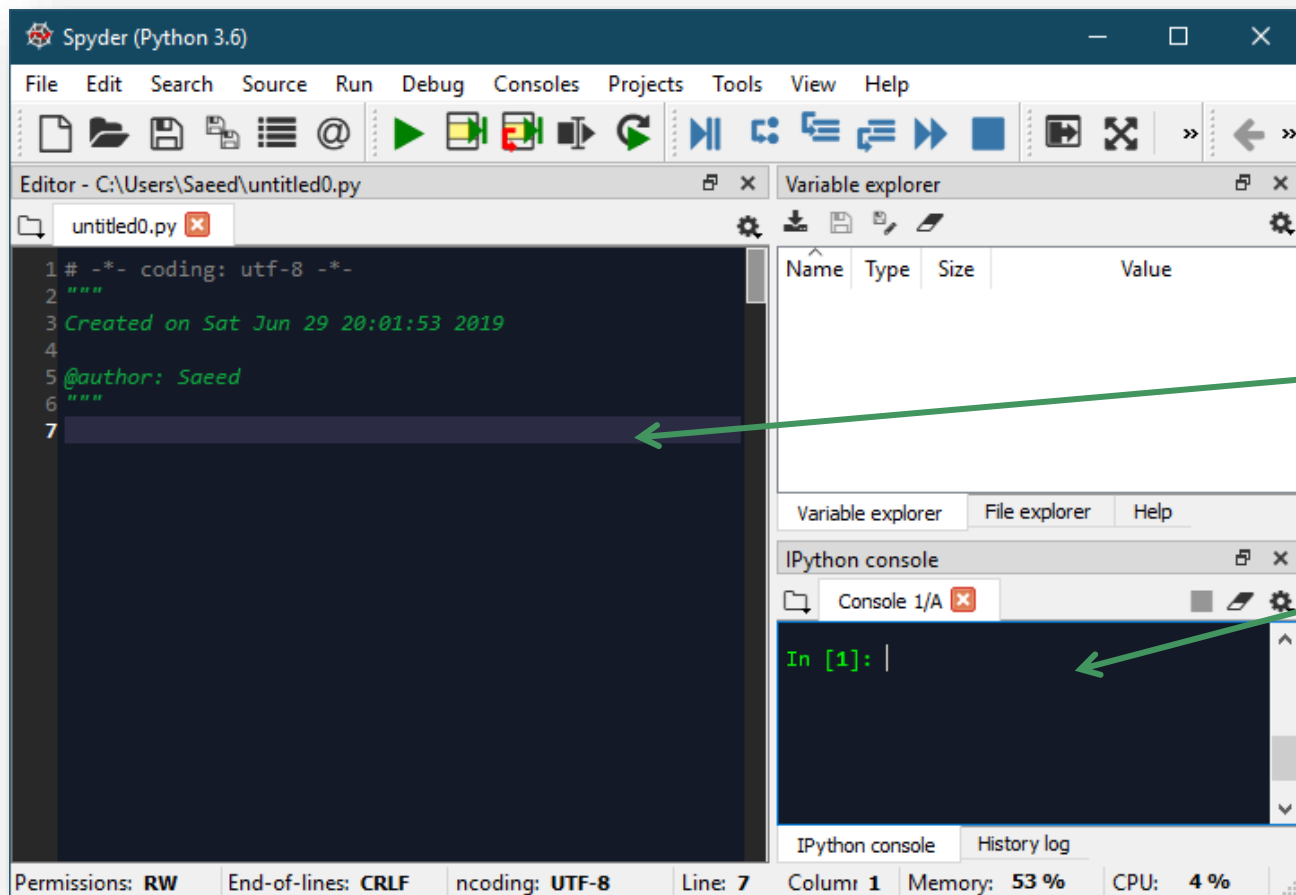
■ متغیرها

```
>>> x = 10
```

```
>>> name, age = 'Ali', 12
```

```
>>> print('My name is', name)
```

روش های اجرای کد



روش های اجرای کدهای پایتون

1. ذخیره کدها در فایل متنی با پسوند .py و اجرای یک جا

2. اجرای خط به خط در کنسول پایتون (IPython یا IDLE، Shell)

قوانین ساختاری در کدنویسی پایتون

- دستورها و نام توابع و متغیرها به حروف بزرگ و کوچک حساس هستند. (Case Sensitive)
- تعداد فاصله و تب در ابتدای هر خط (Indentation) مهم است.
- نام توابع و متغیرها فقط می تواند شامل حروف، اعداد و _ باشد و نباید با عدد شروع شود (از اسامی رزرو شده هم نباشد).

| | | | |
|---------------------|---|------------|---|
| varName, _varName | → | camelCase | ✓ |
| VarName, _VarName | → | PascalCase | ✓ |
| var_name, _var_name | → | snake_case | ✓ |
| 2varName, var\$name | → | invalid | X |

انواع متغیرها

`x = 2` → `integer`

`x = 3.14` → `float`

`x = 'test'` → `string`

`x = True` → `boolean`

`x = [1, 2, 3]`
`x = [1, 2.5, 'a']` } → `list`

`x = (1, 2, 3)`
`x = (1, 2.5, 'a')` } → `tuple`

`x = {1, 2, 3}`
`x = {1, 2.5, 'a'}` } → `set`

`x = {'a':1, 'b':'c'}`
`x = {1:2, 3:'a'}` } → `dictionary`

✓ نوع متغیرها در حین اجرا با توجه به مقادیر آن ها تعیین می شود.

✓ تشخیص نوع متغیر با دستور `type(var)`

✓ انواع `list`، `tuple` و `set` قابل تبدیل به یکدیگر هستند.

✓ آیتم های `list` قابل تغییر ولی در `tuple` غیر قابل تغییر هستند.

✓ در `set` و `dict`، آیتم تکراری وجود نخواهد داشت.

```
>>> X=5
```

Calculations With Variables

| | |
|-------------------------------------|---------------------------------|
| <pre>>>> x+2</pre> | Sum of two variables |
| <pre>7</pre> | |
| <pre>>>> x-2</pre> | Subtraction of two variables |
| <pre>3</pre> | |
| <pre>>>> x*2</pre> | Multiplication of two variables |
| <pre>10</pre> | |
| <pre>>>> x**2</pre> | Exponentiation of a variable |
| <pre>25</pre> | |
| <pre>>>> x%2</pre> | Remainder of a variable |
| <pre>1</pre> | |
| <pre>>>> x/float (2)</pre> | Division of a variable |
| <pre>2.5</pre> | |

| | |
|---------------------------------|----------------|
| <pre>>>> x//2</pre> | Floor division |
| <pre>2</pre> | |
| <pre>>>> x==3</pre> | Equal |
| <pre>False</pre> | |
| <pre>>>> x!=3</pre> | Not equal |
| <pre>True</pre> | |
| <pre>>>> x>5</pre> | Greater than |
| <pre>False</pre> | |
| <pre>>>> x<=5</pre> | Less than |
| <pre>True</pre> | |

if *logical condition* :
→ *statements block*

```
if age<=18:  
    state="Kid"  
elif age>65:  
    state="Retired"  
else:  
    state="Active"
```

- دستور if می تواند با چند دستور elif و تنها یک دستور else در انتها، همراه شود.
- دستورهای درون شرط if، elif و else باید به اندازه یک Tab یا 4 فاصله، جلوتر باشند.

```
if bool(x)==True: ⇔ if x:  
if bool(x)==False: ⇔ if not x:
```

■ استفاده از Enumerate

```
words = ['a', 'b', 'c', 'd']  
for i, word in enumerate(words):  
    print(i+1, word)
```

```
1 a  
2 b  
3 c  
4 d
```

■ استفاده از Iterator

```
nums = [1, 2, 3, 4]  
for num in nums:  
    print(num)
```

```
1  
2  
3  
4
```

■ استفاده از Generator

```
for i in range(5):  
    print(i)
```

```
0  
1  
2  
3  
4
```

این دستورها را امتحان کنید: `[n/2 for n in range(1,21)]` `[n for n in range(1,21)]`

■ استفاده از Continue

```
i = 1
while i < 10:
    i += 1
    if i % 2 == 0:
        continue
    print(i)
```

3
5
7
9

■ استفاده از Break

```
i = 1
while True:
    print(i)
    if i == 4:
        break
    i += 1
```

1
2
3
4

■ حلقه مشروط

```
i = 1
while i < 5:
    print(i)
    i += 1
```

1
2
3
4
5

- <https://projecteuler.net/problem=1>

Multiples of 3 and 5

Problem 1

If we list all the natural numbers below 10 that are multiples of 3 or 5, we get 3, 5, 6 and 9. The sum of these multiples is 23.

Find the sum of all the multiples of 3 or 5 below 1000.

- راه حل در فایل 001.py

- <https://projecteuler.net/problem=2>

Even Fibonacci numbers

Problem 2

Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 2, the first 10 terms will be:

1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...

By considering the terms in the Fibonacci sequence whose values do not exceed four million, find the sum of the even-valued terms.

- راه حل در فایل 002.py

Strings

```
>>> my_string = 'thisStringIsAwesome'
>>> my_string
'thisStringIsAwesome'
```

String Operations

```
>>> my_string * 2
'thisStringIsAwesomethisStringIsAwesome'
>>> my_string + 'Innit'
'thisStringIsAwesomeInnit'
>>> 'm' in my_string
True
```

String Operations

Index starts at 0

```
>>> my_string[3]
>>> my_string[4:9]
```

String Methods

| | |
|---------------------------------|-------------------------|
| >>> my_string.upper() | String to uppercase |
| >>> my_string.lower() | String to lowercase |
| >>> my_string.count('w') | Count String elements |
| >>> my_string.replace('e', 'i') | Replace String elements |
| >>> my_string.strip() | Strip whitespaces |

ترکیب رشته ها و متغیرها

```
>>> S='values are %d, %f' %(x,y)
>>> S='values are {:d},{:f}'.format(x,y)
```

```
%d, {:d} → integer      %s, {:s} → string
%f, {:f} → float
%.2f / %4.2f / {:.2f} / {:4.2f} → limit numbers
```

- <https://projecteuler.net/problem=16>

Power digit sum

Problem 16

$2^{15} = 32768$ and the sum of its digits is $3 + 2 + 7 + 6 + 8 = 26$.

What is the sum of the digits of the number 2^{1000} ?

- راه حل در فایل 016.py

- <https://projecteuler.net/problem=19>

Counting Sundays

Problem 19

You are given the following information, but you may prefer to do some research for yourself.

- 1 Jan 1900 was a Monday.
- Thirty days has September,
April, June and November.
All the rest have thirty-one,
Saving February alone,
Which has twenty-eight, rain or shine.
And on leap years, twenty-nine.
- A leap year occurs on any year evenly divisible by 4, but not on a century unless it is divisible by 400.

How many Sundays fell on the first of the month during the twentieth century (1 Jan 1901 to 31 Dec 2000)?

- راه حل در فایل 019.py

- <https://projecteuler.net/problem=20>

Factorial digit sum

Problem 20

$n!$ means $n \times (n - 1) \times \dots \times 3 \times 2 \times 1$

For example, $10! = 10 \times 9 \times \dots \times 3 \times 2 \times 1 = 3628800$,
and the sum of the digits in the number $10!$ is $3 + 6 + 2 + 8 + 8 + 0 + 0 = 27$.

Find the sum of the digits in the number $100!$

- راه حل در فایل 020.py

- <https://projecteuler.net/problem=6>

Sum square difference

Problem 6

The sum of the squares of the first ten natural numbers is,

$$1^2 + 2^2 + \dots + 10^2 = 385$$

The square of the sum of the first ten natural numbers is,

$$(1 + 2 + \dots + 10)^2 = 55^2 = 3025$$

Hence the difference between the sum of the squares of the first ten natural numbers and the square of the sum is $3025 - 385 = 2640$.

Find the difference between the sum of the squares of the first one hundred natural numbers and the square of the sum.

- راه حل در فایل 006.py

- Numpy: کتابخانه بسیار پرکاربرد برای محاسبات عددی و ماتریسی ← فایل `numpy.pdf`
- Scipy: کتابخانه بسیار پرکاربرد شامل الگوریتم های مختلف علمی ← فایل `scipy.pdf`
- Pandas: کتابخانه پرکاربرد برای محاسبات آماری و کار با داده ها ← فایل `pandas.pdf`
- Scikit-Learn: کتابخانه پرکاربرد برای یادگیری ماشین و پردازش داده ها ← فایل `sklearn.pdf`
- Matplotlib: کتابخانه بسیار پرکاربرد برای نمایش نمودارها و تصاویر ← فایل `matplotlib.pdf`