

# مقدمات پایتون



آزمایشگاه بهینه سازی  
دانشکده علوم ریاضی  
دانشگاه فردوسی مشهد



# آشنایی مفرحانه با مقدمات برنامه نویسی در پایتون



طرح و نظارت

رضا قنبری (عضو هیئت علمی دانشگاه فردوسی)

تدوینگران

دنیا حیدری (مدیر عامل)

خاطره قربانی (عضو هیئت علمی دانشگاه خوارزمی)

زهرا شهری (مسئول آموزش)

# • در دوره اول قرار هست با هم چی یاد بگیریم؟

- ✓ می دونی با پایتون میتوانی انشا یا پیام تبریک تولد بنویسی؟ (Strings)
  - ✓ می دونی متغیر و برچسب اون چیه؟ کجا استفاده میشه؟ (Variable)
  - ✓ می دونی جمع، تفریق، ضرب و تقسیم با پایتون چجوری نوشته میشه؟
  - ✓ می دونی لیست و سایل مورد نیازت برای سفر رو میتوانی با پایتون تهیه کنی و اگه لازم بود چیزی بهش اضافه یا کم کنی؟ (Lists)
  - ✓ میدونی Tuple چیه؟ و چطور میشه توی پایتون یک دیکشنری نوشت؟
  - ✓ یادگیری ماژول لاکپشت ( Turtle Module )
  - ✓ سوال پرسیدن با if و else
  - ✓ استفاده از حلقه for
- همه چیزهایی که گفتم را در این دوره آموزش می بینی.

# معروف ترین محیط های برنامه نویسی پایتون



# • برنامه نویسی چیست؟

به زبان ساده، برنامه نویسی مهارتیه که بهمون کمک میکنه بتونیم با لپتاپ مون با زبان خودش صحبت کنیم و ازش بخوایم خیلی از کارهایی که میخوایم رو برآمون انجام بدی! با زبان های مختلفی میشه با لپتاپ یا کامپیوتر صحبت کرد. پایتون یکی از بهترین اون هاست.



# • با برنامه نویسی چه کارهایی میشه انجام داد؟

علم و تکنولوژی هر روز داره بیشتر و بیشتر رشد میکنه و تقریبا همه چیز داره با کدنویسی انجام میشه. تنها راهی که میشه به این پیشرفت سریع، چنگ زد و از دنیا عقب نموند فقط و فقط یادگرفتن برنامه نویسی هست.



# • چرا پایتون ؟



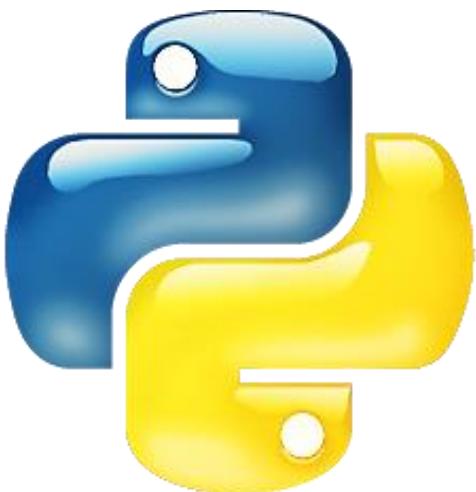
پایتون یک زبان عمومی است و محدود به یک حوزه خاص نمی شود.

یک برنامه نویس پایتون اگر حرفه ای باشد در هر حوزه ای حرفی برای گفتن خواهد داشت.

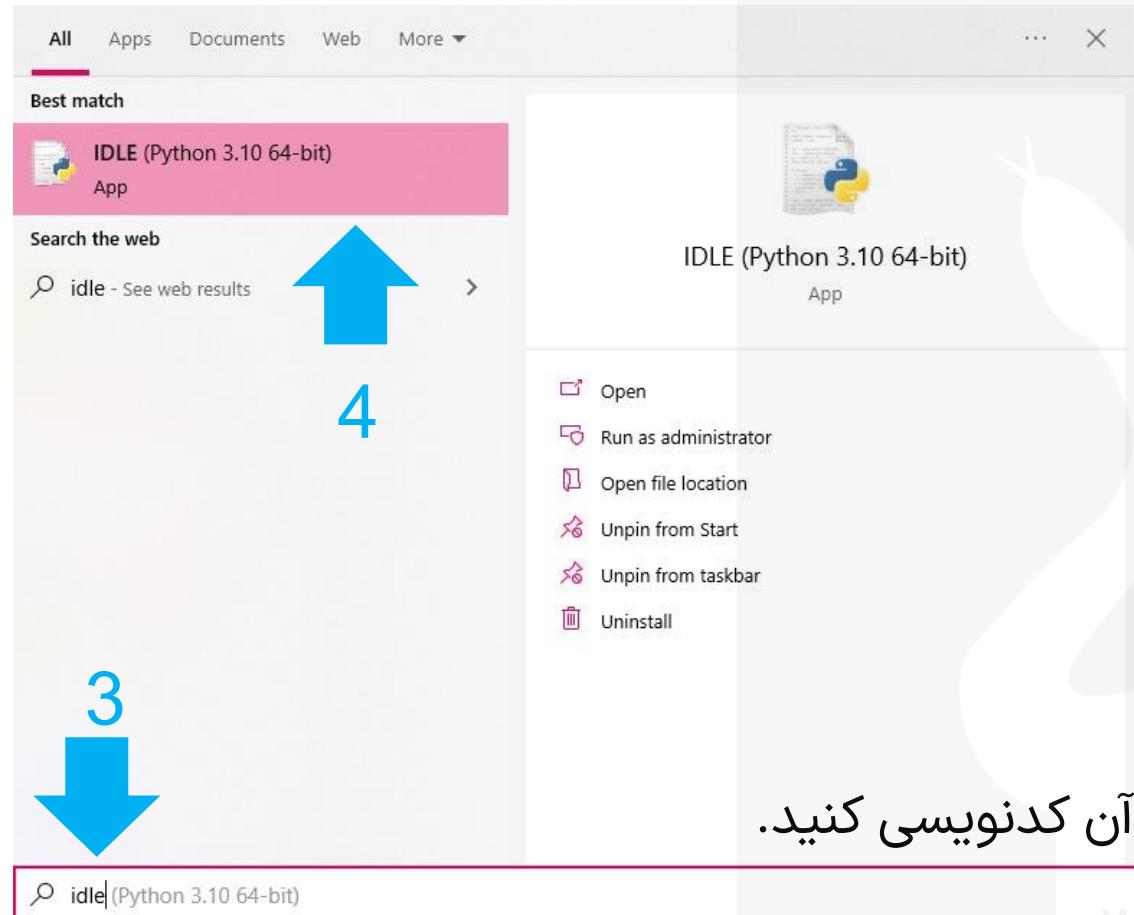
با پایتون دست شما برای ورود به بیشتر حوزه های تکنولوژی باز است.

پایتون بسیار آسان و ساده است حتی کودکان هم میتوانند راحت یاد بگیرند.

پایتون یکی از محبوب ترین زبان های برنامه نویسی ضمن اینکه رایگان، منبع باز و در دسترس همه است کلی راهنمای و کدهای آماده در اینترنت برای پایتون وجود دارد.



# • چطور وارد محیط پایتون بشیم؟



- ۱- پایتون رو نصب کنید.
  - ۲- روی صفحه دسکتاپ لپتاپ تان علامت ذره بین را پیدا کنید اگر نبود وارد منوی استارت (گوشه چپ پایین) شوید
  - ۳- قسمتی که در شکل مقابل با فلش نشان داد ه ایم تایپ کنید: idle
  - ۴- روی برنامه ای که می آید ۲ بار کلیک کنید تا باز شود
  - ۵- شما وارد محیط برنامه نویسی پایتون شده اید.
- بزودی یاد می گیرید چطور یک صفحه تمام سفید بسازید و در آن کدنویسی کنید.

# چطور کلمه بنویسیم؟

```
>>> python
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    python
NameError: name 'python' is not defined
```

```
>>> 'python'
'python'
```

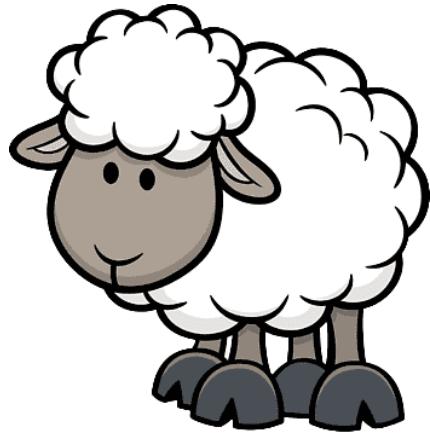
در محیط idle هر کلمه ای که دوست دارد تایپ کنید.

میبینید که پایتون اگه کلمه را بدون یک علامت خاص بنویسیم، ناراحت و قرمز می شود !!  
اسم این علامت **کوتیشن** هست که به این شکل  هست و برای راحتی به آن می گوییم **کوت**  
از روی کیبوردتان پیداиш کنید.



# • دستور print()

دستور print هرچیزی که داخل پرانتز بدهیم را برای ما چاپ می کند .



```
>>> print('I like football')  
I like football  
>>> print('cute sheep')  
cute sheep
```

تمرين:  
یک رشته بسازید و نام و نام خانوادگی تون را به همراه یک پیام دلخواه  
چاپ کنید مانند "Hi there, Peter Poerto".



# • رشته (String)

I'm student

دست به کار بشید! و توی پایتون بنویسید:

```
>>> print('I'm student')
...
SyntaxError: unterminated string literal (detected at line 1)
```

می بینید؟ اگر وسط جمله تون به هر دلیلی از علامت ' استفاده کنید، پایتون گیج میشه و فکر میکنه شما رشته تونو بستید!  
چه راه حلی به ذهن تون میرسه؟

به هر چیزی که داخل ' تک کوتیشن' یا " دو کوتیشن" و حتی "" سه کوتیشن" نوشته بشه رشته یا **string** میگیم

- ۱) از سه کوتیشن برای ساخت رشته استفاده کن
- ۲) قبل از کوتیشن های داخل رشته هستند، علامت بک اسلش \ بذار



---

```
>>> single_quote_str = 'He said, "Aren\'t can\'t shouldn\'t wouldn\'t.'"  
>>> double_quote_str = "He said, \"Aren't can't shouldn't wouldn't.\""  
>>> print(single_quote_str)  
He said, "Aren't can't shouldn't wouldn't."  
>>> print(double_quote_str)  
He said, "Aren't can't shouldn't wouldn't."
```

---



می خوایم توی پایتون یه نامه به دوستمون بنویسیم

اما زمانی که به خط بعد بریم پایتون خطا میده  
هرزمان خواستیم یه پیغام بیش از ۱ خط رو چاپ کنیم، از **"کوتیشن"** استفاده می کنیم تا پایتون به ما اجازه  
رفتن به خط بعد رو بده



```
>>> print(''Hello  
... How are you?  
... I miss you  
... '')  
  
Hello  
How are you?  
I miss you
```

[www.PythonTeek.com](http://www.PythonTeek.com)

# • متغیر (Variable)

به زبان ساده، متغیر ظرفی است که اطلاعات یا داده هایی مثل اسم، تاریخ تولد، سن دوستانت و ... را میتوانه نگهداری کن.

(Label) اسم متغیر

```
>>> soup = 'carrot'
```

تعريف متغیر



## • در انتخاب اسم متغیرها چه چیزهایی را باید رعایت کنیم

- ❖ اسم متغیر میتوانه از حرف ها، عدد ها و کاراکتر آندر لاین "- " تشکیل بشه.
- ❖ اسم متغیر نباید با عدد شروع بشه.
- ❖ فاصله هم در انتخاب اسم متغیر نباید استفاده بشه اما میتوانید از "-" برای جدا کردن استفاده کنین.



## • فایده تعریف متغیر؟

فکر کن یه برنامه ای داریم که تو ش بارها و بارها یه جمله خاص تکرار میشه. بعد از نوشتن ۱۰



خط برنامه، تصمیم میگیریم اون جمله رو عوض کنیم ... وااای

با تعریف متغیر این کار خیلی راحت میشه.

کافیه یکبار اون عبارت خاص رو توی یه متغیر ذخیره کنید

هر بار اون عبارت رو لازم داشتید فقط فقط اسم متغیر رو بیارید.

هر زمان دلتوں خواست خیلی راحت فقط همون یک خط تعریف متغیر رو تغییر بدید.

در اینصورت هرجایی که اسم متغیر رو آورده باشید، اون عبارت اصلاح شده رو برآتون چاپ



میکنه

[www.ythonTeek.com](http://www.ythonTeek.com)

# • جمع، تفریق، ضرب و تقسیم

دقت کن در محاسبه ها، همیشه اول پرانتزها و بعد بقیه عملگرها با ترتیب الویت زیر حساب میشوند

ترتیب اولویت عملگرها (از بالا به پایین)

نماد عملگر	عملگر	
**	توان	Exponent
*, /	ضرب و تقسیم	Multiplication, Division
+, -	جمع و تفریق	Addition, Subtraction

```
>>> result = 30 + 42/3 *5  
>>> print(result)  
100.0  
>>> result = (30 + 42)/3 * 5  
>>> print(result)  
120.0
```



## پروژه ۱:

اگر سه تا ساختمون باشه که رو سقف هر کدومشون ۲۵ تا نینجا مخفی شدن و دو تا تونل باشه که تو هر تونل ۴۰ سامورایی مخفی شدن، حالا بگین چند تا نینجا و سامورایی تو جنگ باهم میجنگن؟؟؟؟



برای تعداد ساختمون، نینجا، تونل و سامورایی مجموعاً ۴ تا متغیر بسازید و جواب مساله رو به کمک اونها بدست بیارید.



## • ترتیب اندیس گذاری در پایتون

یک مثال برای اندیس گذاری رشته: 'PythonTeek'



P	y	t	h	o	n	T	e	e	k
0	1	2	3	4	5	6	7	8	9

طول رشته 'PythonTeek' برابر ۱۰ است اما اندیس گذاری از ۰ تا ۹ است.

میخوای بدونی  
چطور در وسط یک  
رشته، جای خالی نگه  
داری و بعدا اون را  
پر کنی؟؟

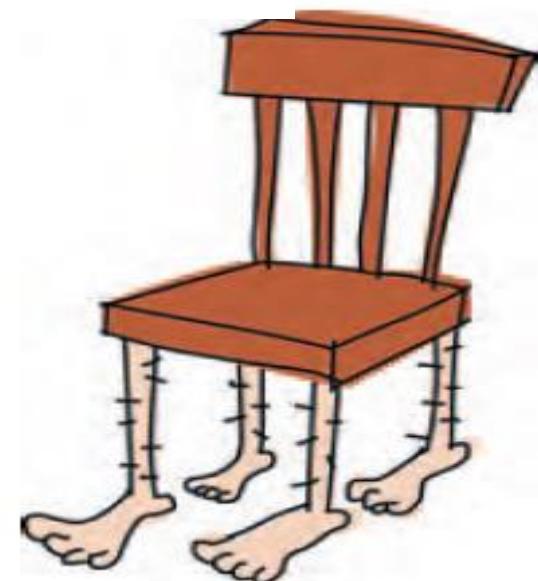
۱. هر جای خالی را با نماد "% " نگه دار.

بعدا که میخوای جای خالی را پر کنی؛ اگر فقط یک جای خالی داشتی،  
محتوای جا خالی را بعد از علامت "%" بنویس و اگرچند تا جای خالی  
داشتی، باید محتوای جاهای خالی را داخل پرانتز بذاری و با ویرگول از  
هم جدا کنی.



```
>>> joke_text = '%s: a device for finding furniture in the dark'  
>>> bodypart1 = 'Knee'  
>>> bodypart2 = 'Shin'  
>>> print(joke_text % bodypart1)  
Knee: a device for finding furniture in the dark  
>>> print(joke_text % bodypart2)  
Shin: a device for finding furniture in the dark
```

```
>>> nums = 'What did the number %s say to the number %s? Nice belt!!'  
>>> print(nums % (0, 8))  
What did the number 0 say to the number 8? Nice belt!!
```



## • تکرار رشته (Multiplying String)



Happy Happy Happy  
Happy Happy Happy Happy  
Happy Happy Happy

---

```
>>> print(10 * 'a')  
aaaaaaaaaa
```

---

## • کدهای برنامه نویسی را کجا بنویسین و چطور ذخیره کنیم ؟

چجوری یک پنجره باز کنیم و کد بنویسیم ؟

**File → New file (window) → اکنون کد خود را بنویسید**

چجوری پنجره باز شده رو ذخیره کنیم ؟

**File → Save As → اسم انتخاب کنید .py**

چجوری برنامه نوشته شده را اجرا کنیم ؟

**Run → Run Module**

## لیست (LIST) •

```
>>> wizard_list = ['spider legs', 'toe of frog', 'eye of newt',
                   'bat wing', 'slug butter', 'snake dandruff']
>>> print(wizard_list)
['spider legs', 'toe of frog', 'eye of newt', 'bat wing', 'slug
butter', 'snake dandruff']
```



```
>>> print(wizard_list[2])
eye of newt
```



```
>>> wizard_list[2] = 'snail tongue'
>>> print(wizard_list)
['spider legs', 'toe of frog', 'snail tongue', 'bat wing', 'slug
butter', 'snake dandruff']
```



```
>>> print(wizard_list[2:5])
['snail tongue', 'bat wing', 'slug butter']
```



•

## اضافه کردن و پاک کردن اشیا در لیست:

---

```
>>> wizard_list.append('bear burp')
>>> print(wizard_list)
['spider legs', 'toe of frog', 'snail tongue', 'bat wing', 'slug
butter', 'snake dandruff', 'bear burp']
```

---




---

```
>>> del wizard_list[5]
>>> print(wizard_list)
['spider legs', 'toe of frog', 'snail tongue', 'bat wing', 'slug
butter', 'bear burp', 'mandrake', 'hemlock', 'swamp gas']
```

---

•

## به هم پیوستن دو لیست:

---

```
>>> list1 = [1, 2, 3, 4]
>>> list2 = ['I', 'tripped', 'over', 'and', 'hit', 'the', 'floor']
>>> print(list1 + list2)
[1, 2, 3, 4, 'I', 'tripped', 'over', 'and', 'hit', 'the', 'floor']
```

---



## لیست ها از رشته ها مفیدتر هستند

- برخلاف رشته، لیست قابل تغییر است.
- اندیس و موقعیت هر عضو لیست را میشه با علامت [] مشخص است.
- میشه اعضا لیست را از یک موقعیت دلخواه تا موقعیت دلخواه دیگری با علامت [ :] جدا کرد.
- به راحتی می توان عضوی را اضافه، کم و یا تغییر داد.
- می تونیم لیست ها را مرتب کنیم و حتی ادغام کنیم.

## پروژه ۲ :

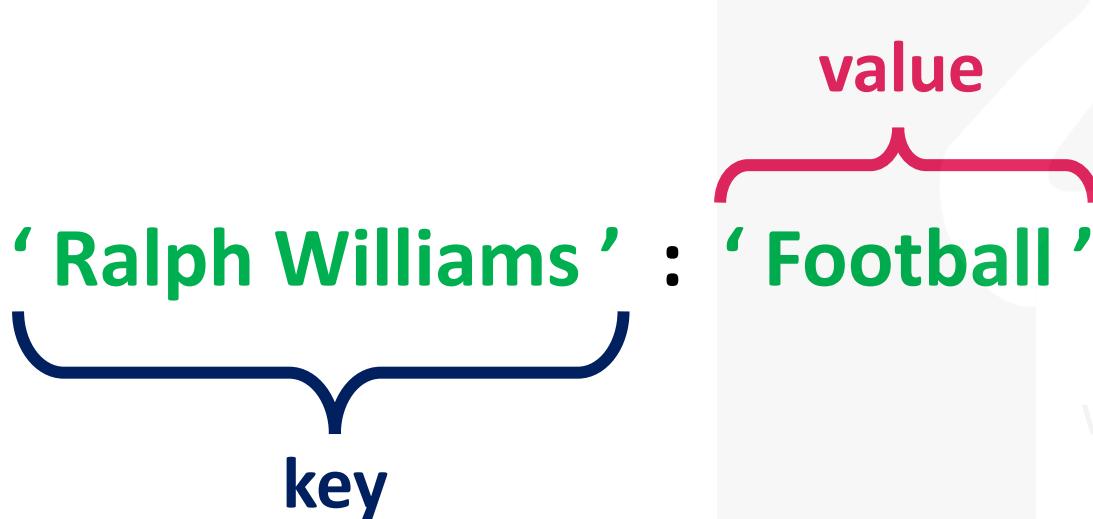
۱. لیستی از سرگرمی های مورد علاقه خود تهیه کنید و نام لیست را **Hobbies** (سرگرمی ها) بگذارید.
۲. لیستی از غذاهای مورد علاقه خود تهیه کنید و نام لیست را **Food** (غذا) بگذارید.
۳. دو لیست را به یکدیگر پیوند بزنید (**join**) و اسم لیست جدید را **Favorites** (علاقه مندی ها) بگذارید.
۴. لیست **Favorites** را چاپ کنید.



# • دیکشنری ها (Maps or Dictionaries)

```
>>> favorite_sports = {'Ralph Williams' : 'Football',  
                      'Michael Tippett' : 'Basketball',  
                      'Edward Elgar' : 'Baseball',  
                      'Rebecca Clarke' : 'Netball',  
                      'Ethel Smyth' : 'Badminton',  
                      'Frank Bridge' : 'Rugby'}
```

دیکشنری ۶ عضوی



دیکشنری ۳ تا ویژگی مهم داره:

۱- علامتش: {} که اسمش آکولاد هست

۲- هر عضوش ۲ بخش داره که هر کدام اسم مخصوص داره و بین

شون : هست

۳- اینجا اندیس نداریم و هر زمان به اندیس احتیاج داشتیم نام  
بخش سمت چپ (key) رو میاریم

## پروژه ۳ :

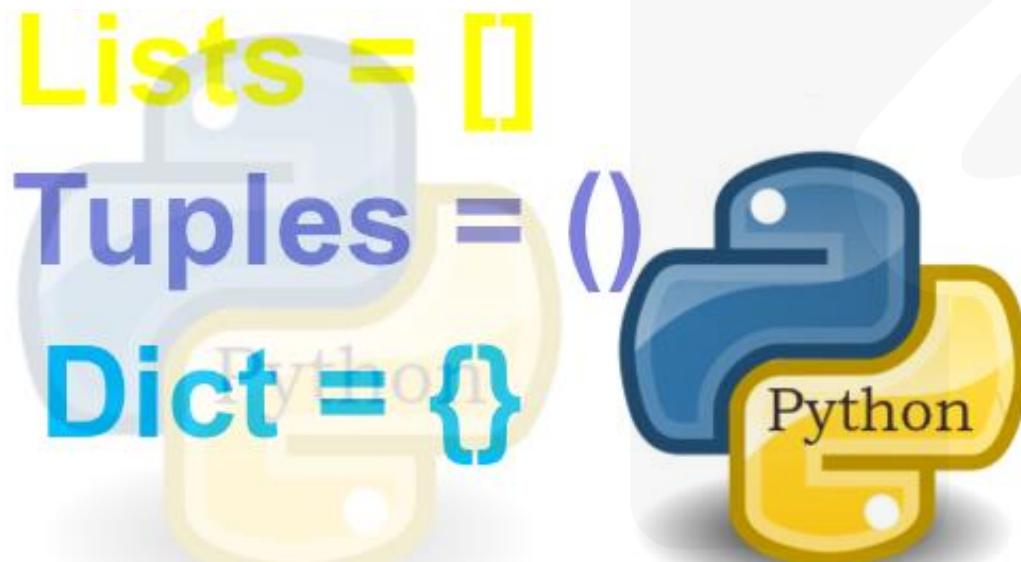
- 
- ۱- دیکشنری ۳ عضوی بنویسید که key های اون نام شما و پدر و مادرتون باشه و value های اون، سن تون باشه
  - ۲- بجای سن مامانتون، عدد ۸ رو بزارید.
  - ۳- عضوی که اطلاعات خودتون هست رو حذف کنید
  - ۴- حالا که تغییرات بالا رو انجام دادید دیکشنری رو چاپ کنید



# • Tuples

چندتایی تقریباً مثل لیست است. اما ابتدا و انتهای یک چندتایی باید پرانتز بذاریم و اطلاعات اون را نمیشه تغییر داد

```
>>> fibs = (0, 1, 1, 2, 3)  
>>> print(fibs[3])  
2
```



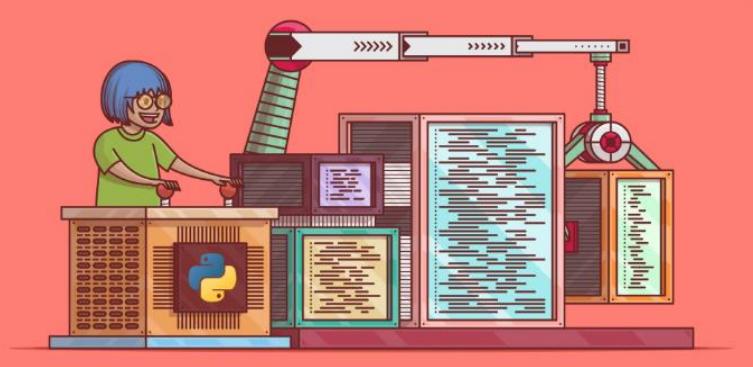
## پروژه ۴ :

- ۱- یک تاپل ۵ عضوی شامل نام دستگاه های عجیب و غریبی که دوست داری در آینده اختراع بشه بساز
- ۲- عضو دارای اندیس ۱ تا ۳ رو نمایش بده
- ۳- عضو دارای اندیس ۲ رو با نام يه دستگاه جدید جایگزین کن
- ۴- تحقیق کن ببین میتونی عضو دارای اندیس صفر رو حذف کنی؟ کل تاپل رو چطور؟

## پروژه ۵ :

یکمی تمرین کن ببین چطور میتونی به کمک دوبار استفاده از علامت دونقطه : تاپل رو برعکس کنی؟  
مثلما  $a=(1,2,3)$  تبدیل بشه به  $(3,2,1)$

# ماژول چیست؟



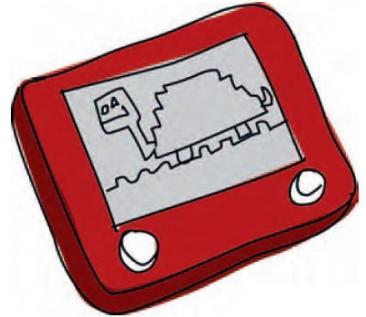
فرض کنید که می‌خواهید برنج درست کنید و برای اینکار وسایل زیر را لازم دارید:

- ۱) برنج      ۲) نمک      ۳) روغن      ۴) آب      ۵) قابلمه      ۶) اجاق گاز

دقت کنید که هیچ کدام از این وسایل را خود شما نساخته‌اید و همه آنها را به صورت آماده دارید. یعنی شما نه برنج را خودتان کاشته‌اید و نه نمک را از دریا خودتان گرفتید و نه روغن را تولید کرده‌اید و نه آب را تصفیه کرده‌اید. در واقع این وسایل از قبل توسط دیگران برای شما آماده و ساخته شده است و تنها کاری که شما باید انجام دهید استفاده از این وسایل برای پختن برنج است.

در زبان پایتون به این چیزهایی که از قبل آماده شده است، «ماژول» می‌گوییم. یعنی لازم نیست همه برنامه‌ها را خودمان از اول بنویسیم، بلکه برخی از برنامه‌ها برای ما از قبل نوشته شده است و ما می‌توانیم از این برنامه‌های آماده - که به آنها ماژول می‌گوییم - استفاده کنیم. استفاده از ماژول‌ها کار ما را بسیار ساده‌تر و سریعتر خواهد کرد.

ماژول به انگلیسی اینگونه نوشته می‌شود: `module`



## ماژول لاکپشت (Turtle Module)

یک لاکپشت در پایتون شبیه یک لاکپشت دنیای واقعی است. می‌دانیم یک لاکپشت نسبت به یک خرگوش خیلی کندتر است و حتی خانه‌اش را به دوش می‌کشد. یک لاکپشت در دنیای پایتون خیلی کوچک است شبیه یک پیکان سیاه «►» که به‌کندی بر روی صفحه‌نمایش حرکت می‌کند. زمانی که لاکپشت در پایتون بر روی صفحه‌نمایش حرکت می‌کند مشابه با حلزون در دنیای واقعی اثری از خود به‌جا می‌گذارد.

با دستور **import**

❖ حالا چجوری به پایتون بگیم که از ماژول **turtle** استفاده کنه؟

---

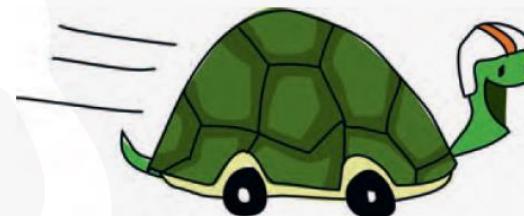
```
>>> import turtle
```

---

- **بوم نقاشی (Canvas):** یک فضایی خالی برای نقاشی کشیدن است. برای درست کردن بوم نقاشی، از تابع `pen` استفاده می کنیم.

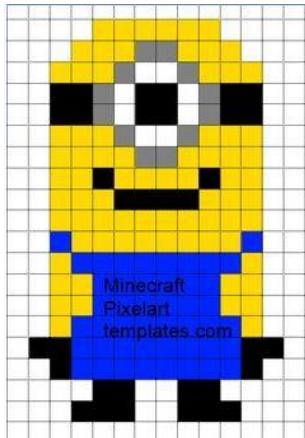
```
>>> t = turtle.Pen()
```

پیکسل  
مس



- **حرکت دادن لاک پشت:**

```
>>> t.forward(50)
```

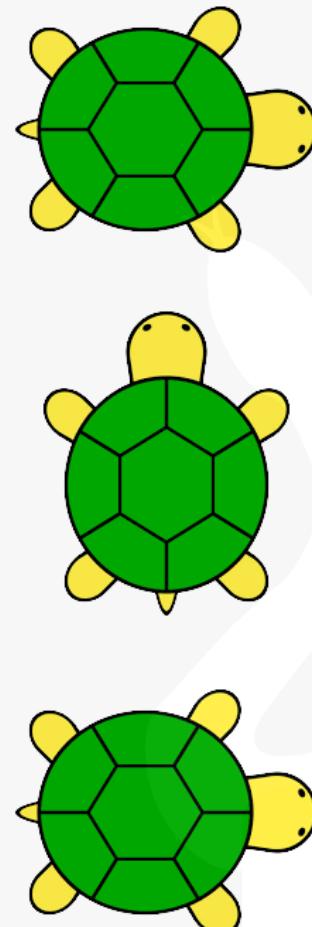


یک پیکسل، یک نقطه روی صفحه‌نمایش می‌باشد «کوچکترین عنصری که می‌توان بر روی صفحه‌نمایش مشاهده کرد». هر چیزی که بر روی صفحه‌نمایش می‌بینید از پیکسل ساخته شده است.

[wwwPythonTeeek.com](http://wwwPythonTeeek.com)

- از لایک پشت میخواهیم که  $90$  درجه به چپ یا راست بچرخد:

```
>>> import turtle  
>>> turtle.left(90)  
  
>>> turtle.right(90)
```



left(90)



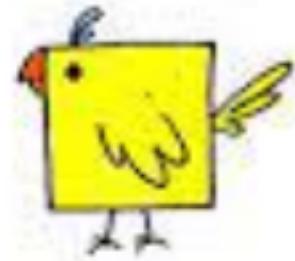
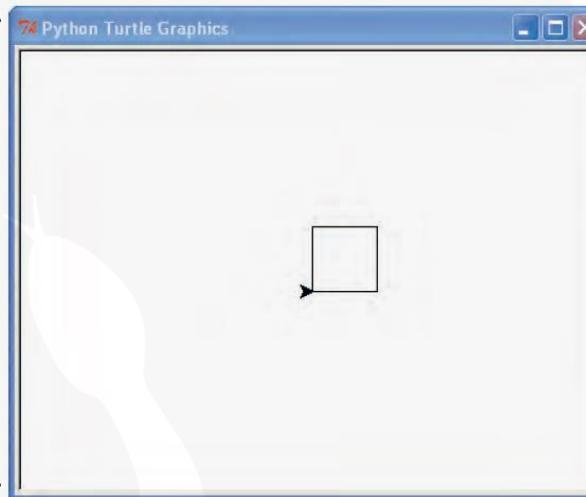
right(90)

رسم مربع:

```
>>> t = turtle.Pen()
```

```
>>> t.forward(50)
>>> t.left(90)
>>> t.forward(50)
>>> t.left(90)
>>> t.forward(50)
>>> t.left(90)
```

میتوانیم برای خودکارمون اسم بزاریم و ازاین به بعد با اون اسم دستور بدیم



پاک کردن صفحه:

```
>>> t.reset()
```

```
>>> t.clear()
```



## • سایر دستورات:

>>> turtle.reset()		پاک کردن صفحه و برگشت لاق پشت به حالت اولیه
>>> turtle.backward(100)		صد پیکسل به سمت عقب حرکت کن.
>>> turtle.up()		هنگام حرکت لاق پشت، چیزی رسم نکن.
>>> turtle.right(90)		جهت لاق پشت را ۹۰ درجه به سمت راست تغییر بده.
>>> turtle.forward(20)		۲۰ پیکسل در جهت سر لاق پشت به سمت جلو حرکت کن. ولی چیزی رسم نمی کند.
>>> turtle.left(90)		جهت لاق پشت را ۹۰ درجه به سمت چپ تغییر بده.
>>> turtle.down()		هنگام حرکت لاق پشت، عمل رسم را انجام بده.
>>> turtle.forward(100)		۱۰۰ پیکسل در جهت سر لاق پشت به سمت جلو حرکت کن. (همراه با رسم)



## • مخفف ها :

اگه دوست داشتی میتونی بجای بعضی دستورات صفحه قبل، از کوتاه شده اونها استفاده کنی

`turtle.fd(100)`

forward

`turtle.bk(100)`

backward

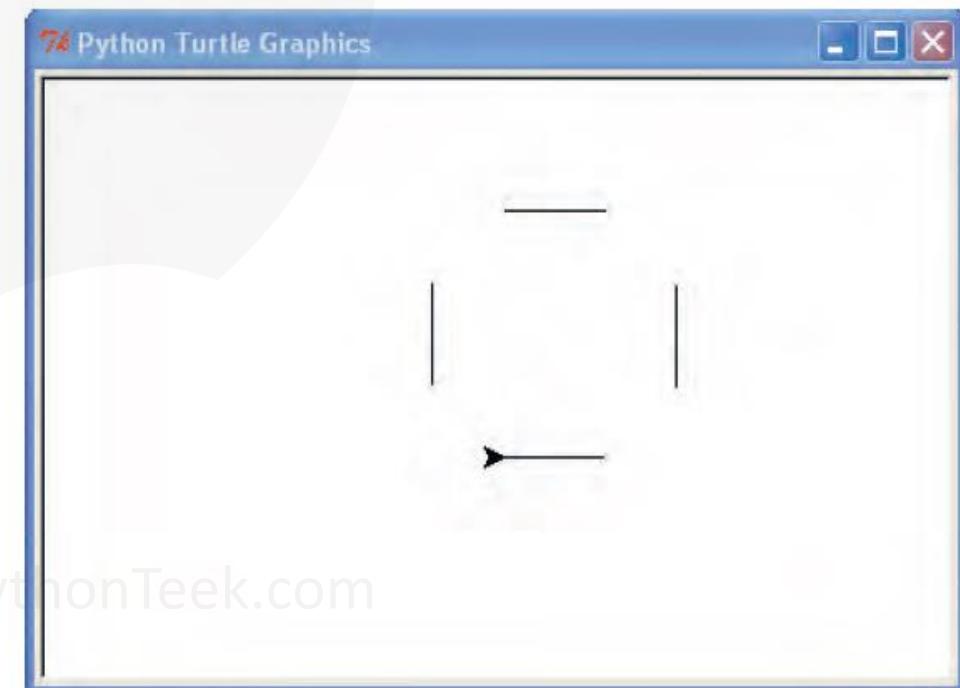
`turtle.rt(90)`

`turtle.lt(90)`

???

تمرین ۱ :

با دستورات ترتل یک مربع بدون گوشه  
مثل شکل سمت راست بسازید



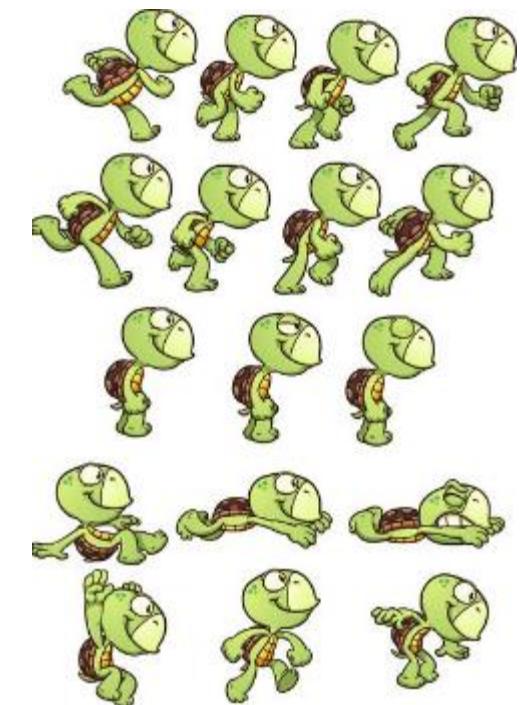


## نمایش لاک پشت به جای نماد پیکان :

>>> turtle.shape('turtle')		نمایش لاک پشت
>>> turtle.shape('circle')		نمایش دایره
>>> turtle.shape('square')		نمایش مربع
>>> turtle.shape('triangle')		نمایش مثلث
>>> turtle.shape('arrow')		نمایش فلش
>>> turtle.shape('classic')		نمایش پیکان یا همان شکل اصلی

# کنترل سرعت حرکت لاک پشت:

دستور	سرعت	n
<code>&gt;&gt;&gt; turtle.speed(0)</code>	خیلی سریع	.
<code>&gt;&gt;&gt; turtle.speed(10)</code>	سریع	۱۰
<code>&gt;&gt;&gt; turtle.speed(6)</code>	نرمال	۶
<code>&gt;&gt;&gt; turtle.speed(3)</code>	کند	۳
<code>&gt;&gt;&gt; turtle.speed(1)</code>	خیلی کند	۱

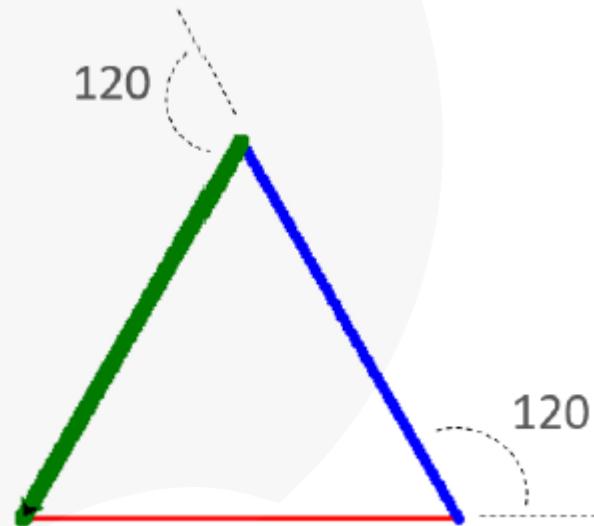


## تغییر رنگ و ضخامت خط ها:

```
turtle.pencolor('red')  
turtle.pensize(2)  
turtle.fd(100)
```

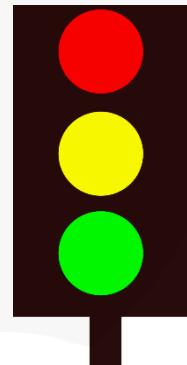
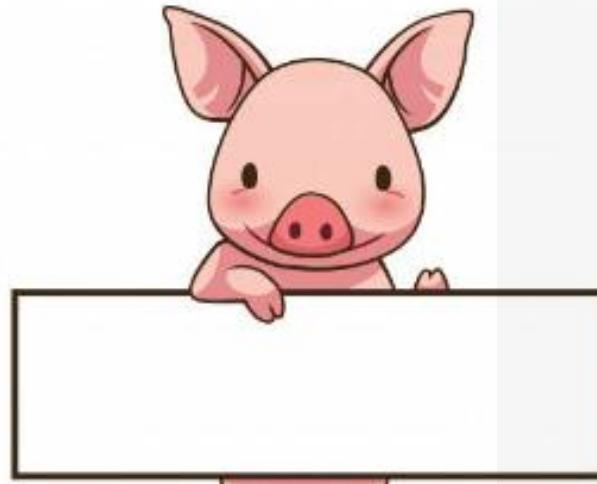
```
turtle.pencolor('blue')  
turtle.pensize(5)  
turtle.left(120)  
turtle.fd(100)
```

```
turtle.pencolor('green')  
turtle.pensize(8)  
turtle.left(120)  
turtle.fd(100)
```



• تغییر رنگ بوم :

```
import turtle  
t = turtle.Pen()  
t.Screen().bgcolor('black')
```



تمرین ۲ :

یک چراغ راهنمایی زیبا رسم کنید.

تمرین ۳ :

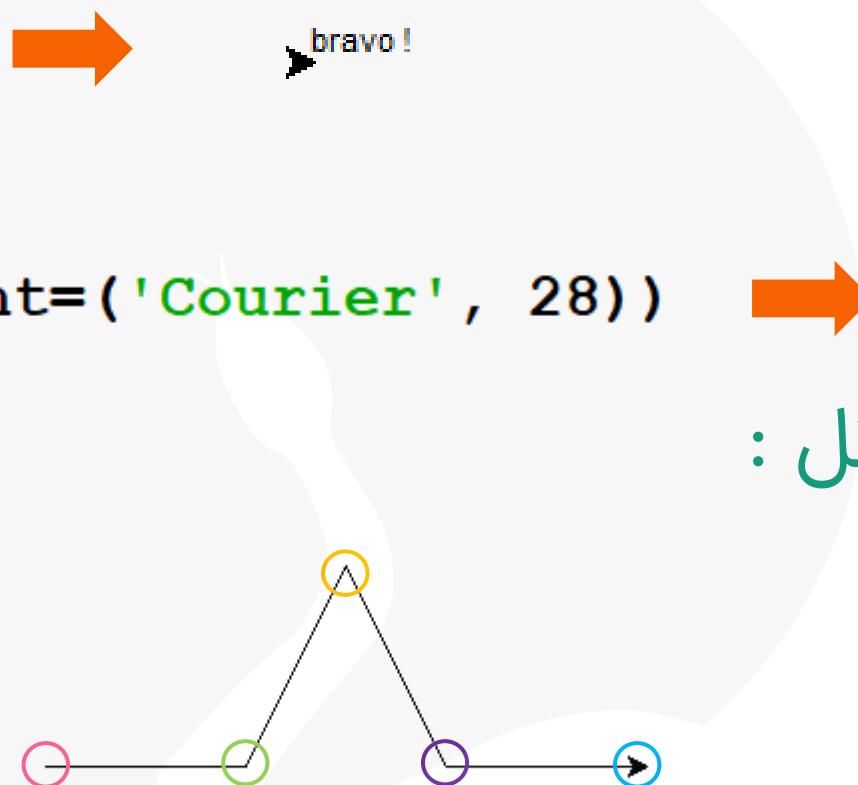
یک مستطیل با طول ۳۰۰ پیکسل و عرض ۱۵۰ پیکسل رسم کنید.

بطوریکه هر ضلع آن، رنگ، ضخامت، سرعت و نمایش متفاوتی(لاکپشت، مثلث، مستطیل و دایره) داشته باشد.

```
import turtle  
t = turtle.Pen()  
t.write('bravo !')
```

```
t.write('bravo !', font=('Courier', 28))
```

```
import turtle  
t = turtle.Pen()  
t.goto(100,0) ○  
t.goto(150,100) ○  
t.goto(200,0) ○  
t.goto(300,0) ○
```



نوشتن در ترتل :

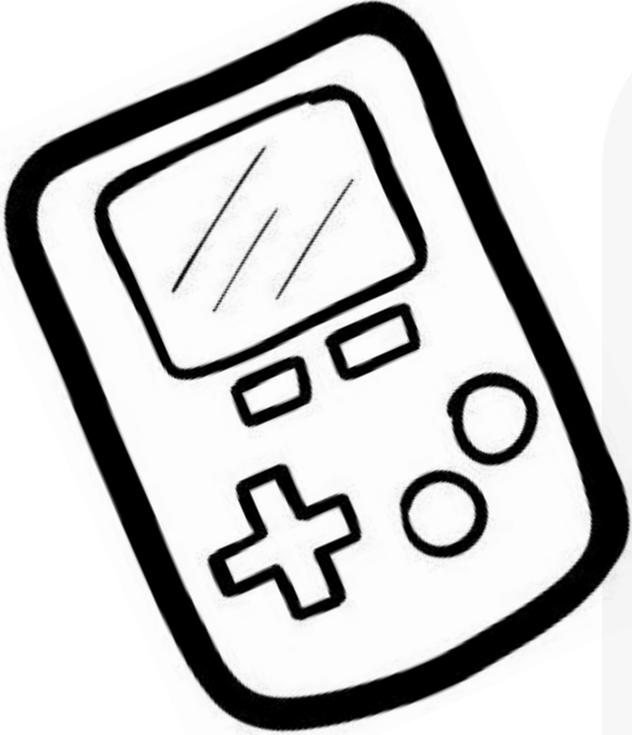
bravo !

جابجا شدن در ترتل :

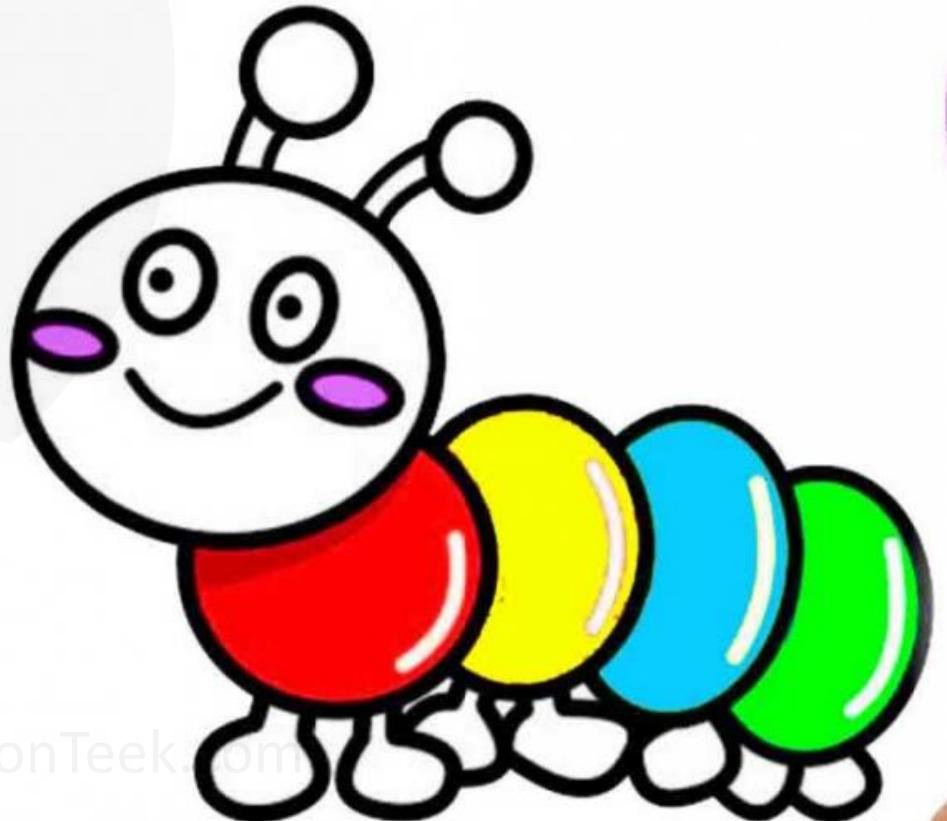
- در این روش فقط آدرس نقاط مهمه
- برای رسیدن به هر نقطه دوباره از نقطه شروع (نقطه صورتی) آدرس میدیم

تمرین ۴ : ستاره بالا را رنگ آمیزی و کامل کنید. همچنین روی بوم یک پیغام بنویسید.

## پروژه ۶:



حالا که یادگرفتی با کمک ترتل  
نقاشی بکشی  
میتوانی نقاشی های ساده ای  
که دوست داری رو انتخاب  
کنی و شروع کنی  
اگه دوست داشتی از نقاشی  
های این صفحه هم میتوانی  
ایده بگیری



# نحوه نوشتن جملات شرطی:

بگذارید مثالی بزنم، راننده قطاری به نزدیک یک دو راهی می‌رسد و اگر «بیشتر از 100 لیتر سوخت داشت»، می‌تواند به «سمت زاهدان» برود و در غیر اینصورت (یعنی اگر سوخت کافی نداشته باشد) باید به «طرف جایگاه سوخت گیری» حرکت کند.



این دستور را به شکل کامپیوتری برای شما می‌نویسم:

اگر سوخت بزرگتر از 100 بود:

حرکت به سمت زاهدان

: وگرنه :

حرکت به سمت جایگاه سوخت گیری

```
if سوخت > 100:
```

حرکت به سمت زاهدان

```
else:
```

حرکت به سمت جایگاه سوخت گیری



- چند سالته؟ اگر سنت بیشتر از ۲۰ هست پس خیلی مسنی !

```
>>> age = 25
>>> if age > 20:
    print('You are too old!')
    print('Why are you here?')
SyntaxError: unexpected indent
```

```
>>> age = 25
>>> if age > 20:
    print('You are too old!')
    print('Why are you here?')
    print('Why aren\'t you mowing a lawn or sorting papers?')
```

```
>>> age = 13
>>> if age > 20:
    print('You are too old!')
```



فاصله ها در نوشتن عبارات اهمیت دارند.

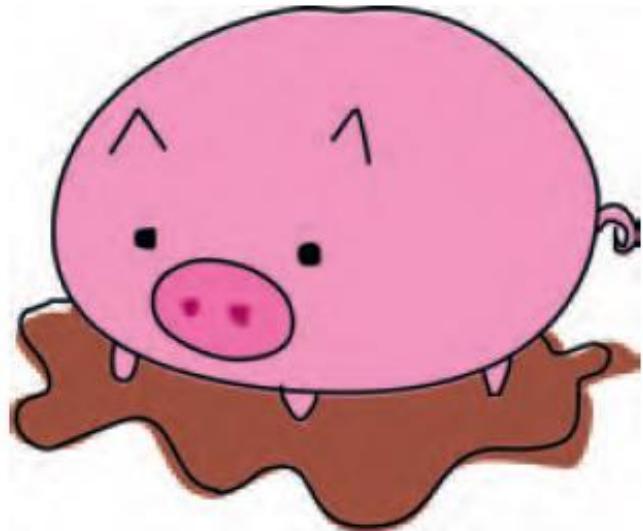
فاصله مناسب : ۴ تا فاصله از زیر if



یعنی اینطوری



## • شرط ها در مقایسه کردن به ما کمک می کنند



```
>>> age = 12  
>>> if age == 12:  
    print("A pig fell in the mud!")  
else:  
    print("Shh. It's a secret.")
```

A pig fell in the mud!

یک شرط، دستور برنامهنویسی است که 'بله' (درست) یا 'خیر'('no') (غلط یا False) را برمی‌گرداند. علامت‌های مشخصی (یا عملگرهای) جهت ایجاد شرط وجود دارند، همچون:

=	برابر
!=	نابرابر
>	بزرگ‌تر از
<	کم‌تر از
>=	بزرگ‌تر یا مساوی
<=	کوچک‌تر یا مساوی

: if-else

```
>>> age = 12
>>> if age == 12: شرط
...     print('You and I are classmates')
... else:
...     print('oh sorry !')
...
...
...
You and I are classmates
```

دستورات

تمام خط های دستورات باید همگی  
کامل‌لازیم باشند



```
age = 12
if age == 12:
    print('Hello')
    print('You and I are classmates')
else:
    print('oh sorry !')
    print('nice to meet you')
```

block

block

[wwwPythonTeeek.com](http://wwwPythonTeeek.com)

# • دستورات if و elif .

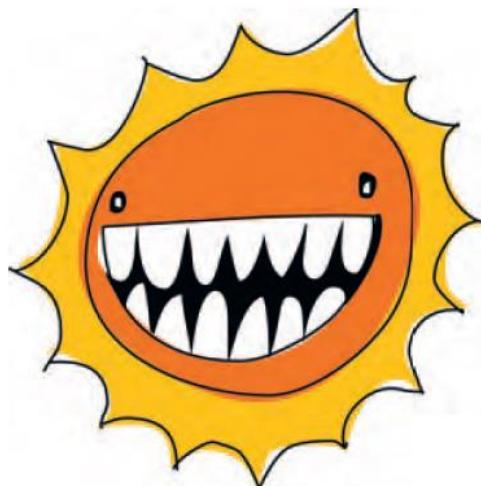
```
>>> age = 12
❶ >>> if age == 10:
❷     print("What do you call an unhappy cranberry?")
❸     print("A blueberrv!")
❹ elif age == 11:
❺     print("What did the green grape say to the blue grape?")
❻     print("Breathe! Breathe!")
❾ elif age == 12:
❿     print("What did o say to 8?")
❬     print("Hi guys!")
❭ elif age == 13:
❮     print("Why wasn't 10 afraid of 7?")
❯     print("Because rather than eating 9, 7 8 pi.")
❰ else:
❱     print("Huh?")
```

What did o say to 8? Hi guys!



ترکیب شرط ها:  شما می توانید با استفاده از **and** و **or** چند شرط را باهم ترکیب کنید.

```
gender = 'female'  
height = 120  
if gender == 'female' or gender == 'male':  
    print('welcome to cinema !')  
    if height < 150 and height > 110:  
        print('You sit in the first to third row')  
    else:  
        print('Please sit in row 4 onwards')
```



در مثال بالا، اول جنسیت و قد تعریف شده بعد اگر جنسیت، خانم یا آقا بود، پیام خوشامدگویی چاپ میشه دراینصورت اگر: قد بین ۱۱۰ تا ۱۵۰ سانتیمتر بود، پیغام: "ردیف اول تا سوم صندلی های سینما بنشینید" درغیراینصورت: پیغام: لطفا ردیف ۴ به بعد بنشینید

# متغیرها بدون هیچ مقدار:

None

□ ما می توانیم هیچ چیز یا مقدار خالی (**None**) را برای یک متغیر در نظر بگیریم. در پایتون **None** یعنی هیچ مقدار. **None** با مقدار **0** فرق دارد.

```
>>> myval = None  
>>> print(myval)  
None
```

کاربردهای

**None**



□ مقدار **None** برای یک متغیر یکی از راههای بازگرداندن آن به حالت اولیه و خالی خود است.

□ اگر بخواهید متغیری را بعدا مقدار دهی بکنید، میتواند مقدار اولیه اش را **None** در نظر بگیرید.

# تفاوت میان رشته ها (strings) و اعداد (numbers)

- تفاوت میان عدد ۱۰ و رشته '۱۰' در چیست؟  
پایتون عدد داخل '' را به عنوان عدد نمی بیند.
- اما پایتون یک **جادویی** دارد که میتواند عدد را به رشته و رشته را به عدد تبدیل کند.

---

```
>>> age = '10'  
>>> converted_age = int(age)
```

---

---

```
>>> age = 10  
>>> converted_age = str(age)
```

---





## • تمرین ۱۴:

- کد زیر چه کاری را انجام می دهد؟

---

```
>>> money = 2000
>>> if money > 1000:
    print("I'm rich!!")
else:
    print("I'm not rich!!")
    print("But I might be later...")
```

---

## سوال پرسیدن از کاربر به کمک : input()

```
>>> favorite_foods = input('How many foods do you really like?')  
How many foods do you really like?4  
>>> print(favorite_foods)  
4
```

دقت کنید!  
دستور `input` جواب شما را (چه عدد چه string) بصورت string ذخیره میکند!



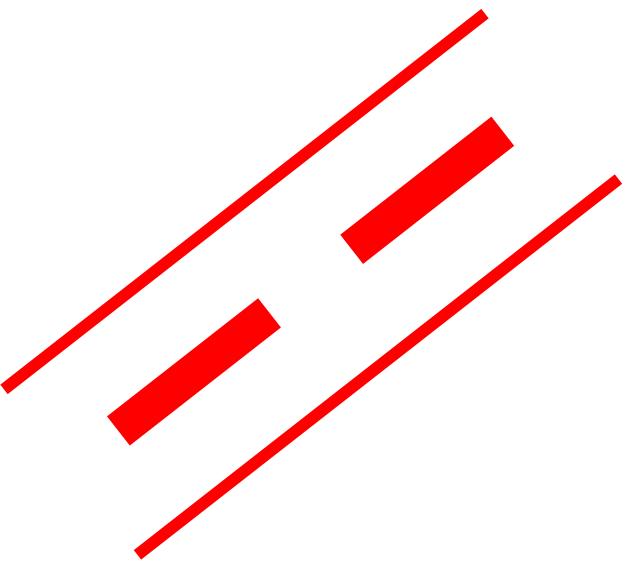
```
>>> type(favorite_foods)  
<class 'str'>
```

بنابراین اگر نیازداشتید از متغیر `favorite_foods` به عنوان عدد استفاده کنید کافیست از دستور `int()` قبل از `input` استفاده کنید.

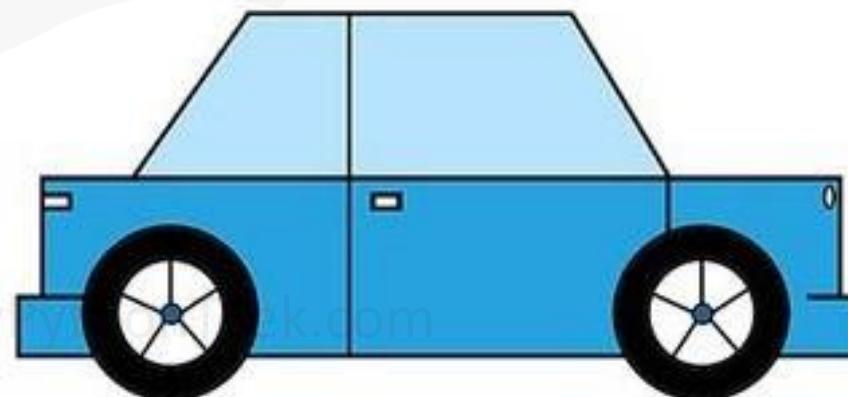
```
favorite_foods = int(input('...'))
```



## • پروژه ۷ : (ترکیب turtle + if-elif-else



- ۱- از کاربر فاصله خانه تا مدرسه اش را بپرسید و در متغیری بنام d ذخیره کنید.
- ۲- اگر d بین ۱ تا ۳ کیلومتر بود، به کمک ترتل، **خیابان** رسم کنید.  
در غیر این صورت اگر d بین ۳ تا ۵ کیلومتر بود، به کمک ترتل، **اتوبوس** رسم کنید.  
در غیر این صورت به کمک ترتل، **ماشین** رسم کنید.

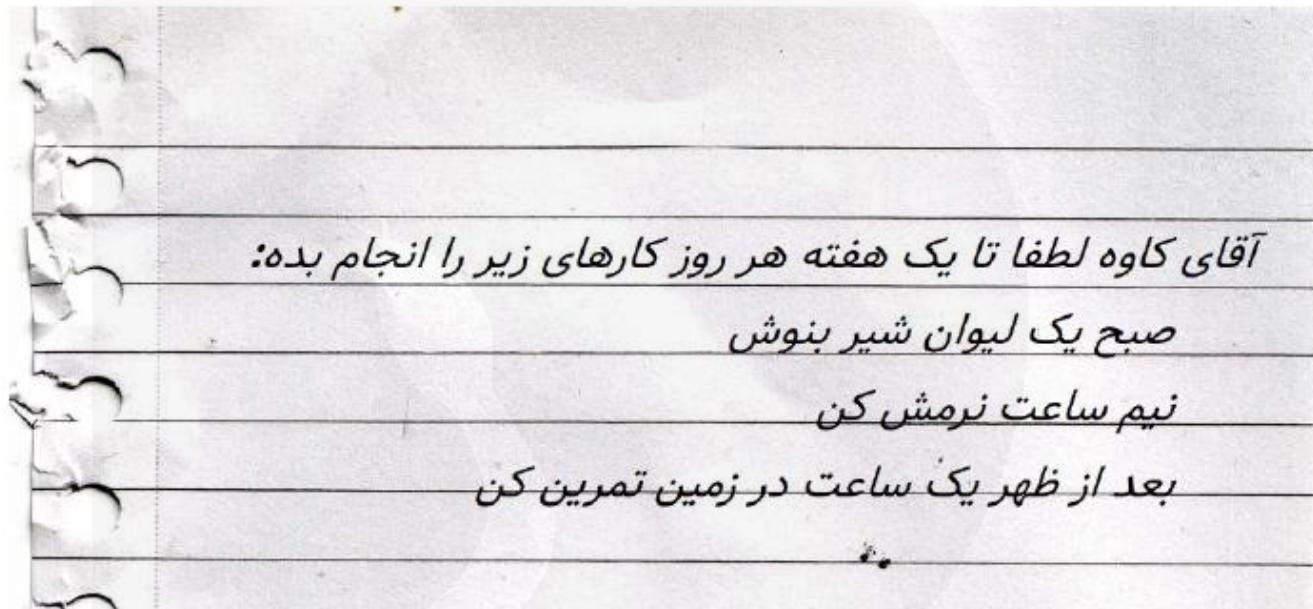


# حلقه:

بگذارید یک مثال ساده از حلقه‌ها بزنم تا منظورم را دقیقاً متوجه شوید. هر روز خورشید از مشرق بیرون می‌آید و در مغرب غروب می‌کند. هر روز زمین به دور خودش یک دور کامل می‌چرخد. زمین سال‌هاست که به دور خورشید می‌چرخد. در همه این مثال‌ها یک چیز مشترک وجود دارد. آن چیز اتفاقی که با یک نظم خاص بارها و بارها تکرار می‌شود که در کامپیوتر به این تکرارهای پیاپی اصطلاحاً «حلقه» می‌گوییم.



اجازه دهید با یک مثال خیلی ساده منظورم را بگویم. شما برای شرکت در مسابقات فوتبال مدرسه انتخاب شده‌اید و مربی می‌خواهد که شما برای مسابقه کاملاً آماده باشید و این دستورها را بر شما می‌نویسید.



اگر خوب دقت کنید می‌بینید که :

- ۱) مربی از کاوه خواسته که کارهایی را ۷ روز انجام دهد . مربی در انتهای سطر اول، علامت دو نقطه گذاشته است و این علامت نشان می‌دهد که این سطر ادامه دار است.  
۲) جملات سطرهای دوم، سوم و چهارم کمی جلوتر نوشته شده‌اند و نشان دهنده کارهایی است که کاوه باید هر روز انجام دهد. این شکل نوشتمن دستورها ترکیبی زیبا است و ما به شکل واضحی می‌فهمیم که منظور مربی انجام چه کارهایی است.

## استفاده از حلقه :for

- در پایتون برای کم کردن تایپ و کاهش تکرار از دستور for استفاده می کنیم:

---

```
>>> for x in range(0, 5):
    print('hello')
```

```
hello
hello
hello
hello
hello
```

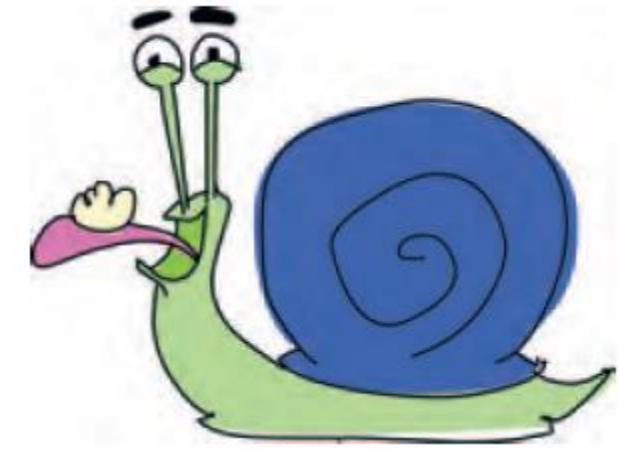
- ترکیب باحال

---

```
>>> print(list(range(10, 20)))
[10, 11, 12, 13, 14, 15, 16, 17, 18, 19]
```

## • مثالی دیگر

```
>>> wizard_list = ['spider legs', 'toe of frog', 'snail tongue',  
                  'bat wing', 'slug butter', 'bear burp']  
>>> for i in wizard_list:  
    print(i)  
spider legs  
toe of frog  
snail tongue  
bat wing  
slug butter  
bear burp
```



## • فاصله گذاشتن بسیار اهمیت دارد. به مثال زیر دقت کنید:

```
>>> hugehairypants = ['huge', 'hairy', 'pants']  
>>> for i in hugehairypants:  
    print(i)  
    print(i)
```

SyntaxError: unexpected indent





## • مثالی از حلقه های تو در تو

به نظر شما خروجی دستور زیر چیست؟

```
>>> pizza = ['Sausage' , 'cheese' , 'mushroom']  
>>> for i in pizza:  
...     print(i)  
...     for j in pizza:  
...         print(j)
```

---

```
>>> for x in range(0, 20):  
    print('hello %s' % x)  
    if x < 9:  
        break
```

---

پروژه  $\wedge$  : حلقه مقابل چه کاری را انجام می دهد؟

## • پروژه ۹ :

یک ۵ ضلعی رنگارنگ بسازید!

یک لیست از رنگها تعریف کنید و برای رنگ آمیزی هر ضلع، به کمک حلقه for هربار یک رنگ از لیست انتخاب کنید تا تمام اضلاع ستاره رنگ بشه.



pngtree.com

## پروژه ۱۰ :

قراره برنامه ای بنویسید که دزدهای گاو صندوق رو شناسایی کنه!

به کمک حلقه for تا ۳ بار رمز گاو صندوق رو از کاربر بپرسید.

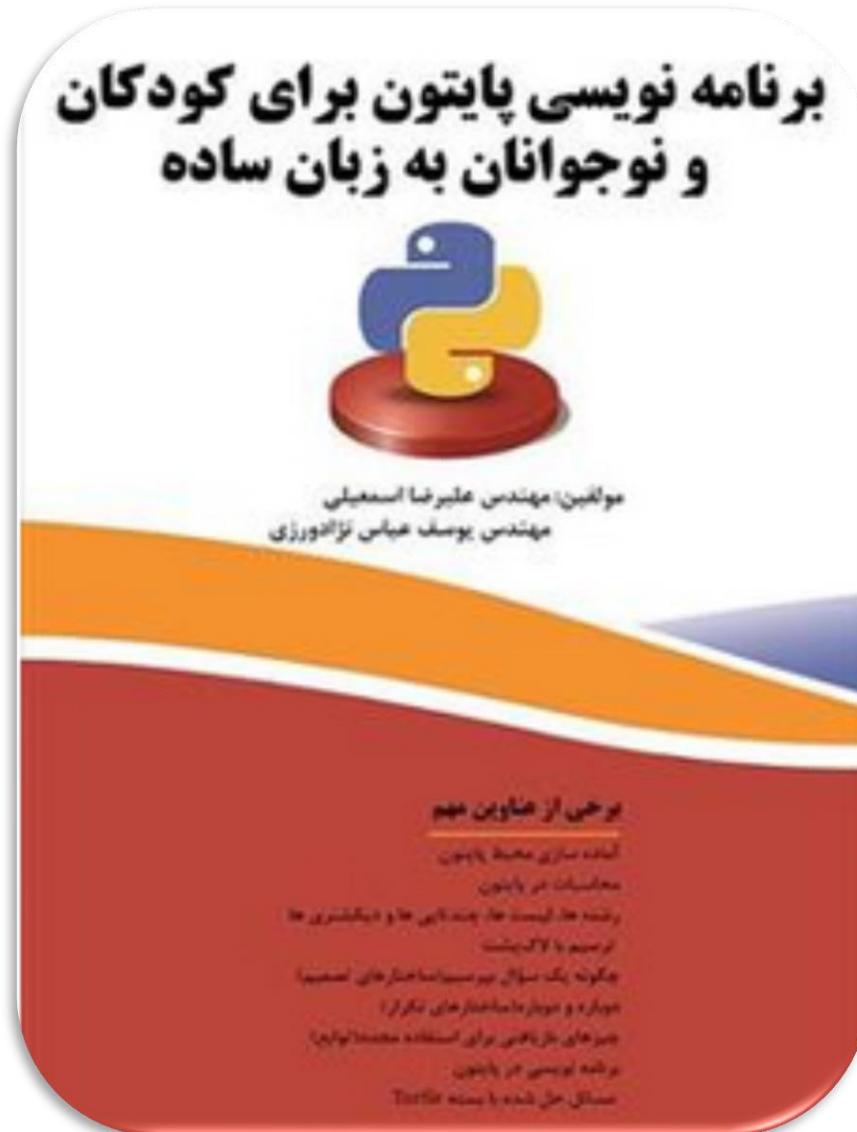
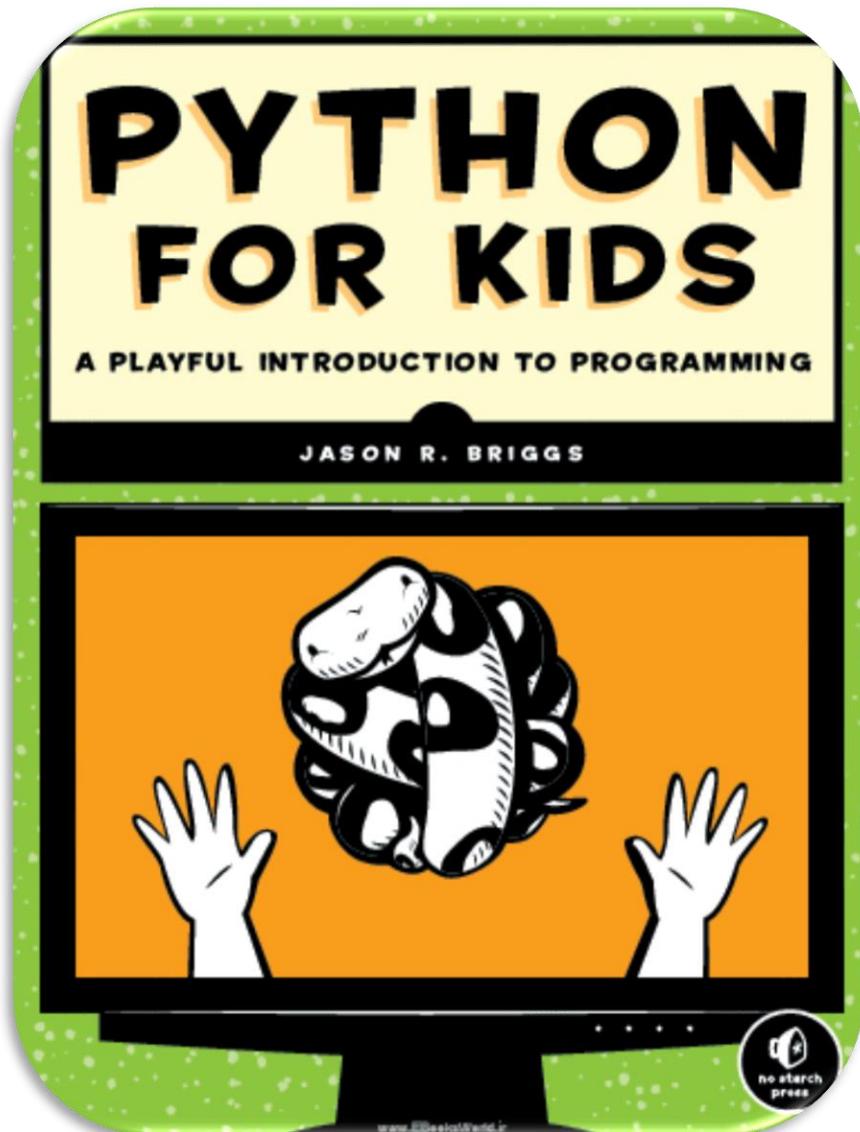
اگر درست وارد شد، به کمک ترتل عکس **کلید** رسم کنید.

در غیر اینصورت به کمک ترتل عکس **قفل** رسم کنید و پیغام: ”**your password is incorrect**“ را چاپ کنید.

# راهنمای رنگ ها در پایتون

- ■ ■ رنگ بنفس : دستورها پایتون مانند دستور `print` بنفس خواهند شد.
- ■ ■ رنگ آبی : خروجی و یا نتیجه دستورها با این رنگ مشخص می‌شوند.
- ■ ■ رنگ نارنجی : دستورها ویژه زبان پایتون مانند `if` و `else` به این رنگ در می‌آیند. به این دستورها ویژه اصطلاحا (بخوانید کی ورد) گفته می‌شود. Keyword
- ■ ■ رنگ سیاه: رنگ سایر دستورها برنامه مانند `2+2` است.
- ■ ■ رنگ قرمز: اگر نتیجه یا خروجی یک دستور خطاب باشد، با این رنگ نمایش داده می‌شود.
- ■ ■ رنگ سبز: رنگ `String`ها (رشته‌ها) که در داخل کوتیشن‌ها قرار گرفته‌اند.

# منابع



# این ترم هم به پایان رسید

## تماس با ما

مدیر عامل مدرسه خلاقیت و نوآوری پایتون تیک

دکتر دنیا حیدری ۰۹۱۵۱۵۹۰۹۰۷

## مدیر آموزش

مهندس زهرا شهری ۰۹۱۵۵۷۱۴۶۹۵۹

