آموزش رایگان پایتون (python) - آشنایی با نوع های داده‌ای رشته‌ای،منطقی و تاریخ‌وزمان در پایتون

در شماره گذشته آموزش پایتون با نوع‌های داده‌ای صحیح، اعشاری و مختلط آشنا شدیم. در این بخش به سراغ سایر نوع‌های داده‌ای موجود در پایتون خواهیم رفت.

برای مطالعه بخش هفتم آموزش رایگان پایتون اینجا کلیک کنید

چرا به نوع‌های داده‌های عددی مختلف نیاز داریم؟

پرسشی که برخی از توسعه‌دهندگان مبتدی مطرح می‌کنند این است که اساسا چرا به نوع‌های عددی مختلف نیاز داریم، در صورتی که می‌توانیم تنها با یک مقدار صحیح همه محاسبات را انجام دهیم. برای آن‌که شناخت درستی از ضرورت وجود نوع‌های عددی مختلف به دست آورید، ابتدا باید ببینید که کامپیوترها چگونه کار می‌کنند؟ یک مقدار صحیح در کامپیوترها در قالب مجموعه‌ای از بیت‌هایی که کامپیوتر به شکل مستقیم آن‌ها را می‌خواند ذخیره می‌شود. در نقطه مقابل، اعدادی که نقطه اعشاری دارند به شیوه کاملا متفاوتی ذخیره‌سازی می‌شوند. برای درک بهتر این مسئله به زمان مدرسه خود باز گردید. در زمان تحصیل شما در کلاس‌های مختلفی درس می‌خواندید که هر یک از آن‌ها مطلب متفاوتی را به شیوه‌ای متفاوت به شما آموزش می‌دادند. به‌طور مثال، کلاس‌های انتگرال و شیمی به شیوه‌ای کاملا متفاوت مطالب را به شما یاد می‌دادند، در دنیای کامپیوترها نیز یک چنین مسئله‌ای صادق است. هر نوع داده‌ای برای کار خاصی در نظر گرفته شده است. مقدار اعشاری در قالب یک بیت علامت‌دار (مثبت یا منفی)، منتیسا (بخش کسری یک عدد) و توان که عدد را دو برابر می‌کند ذخیره می‌شود. برای آن‌که در یک معادله بتوانید بخش کسری یک عدد را به دست آورید باید از معادله‌ای شبیه به مثال زیر استفاده کنید:

```
Value = Mantissa * 2^Exponent
```

در یک لحظه، کامپیوترها محاسبات مختلفی را روی اعداد اعشاری انجام می‌دهند، اما همه آن‌ها بر مبنای استاندارد IEEE-754 یک مقدار اعشاری را به شما نشان می‌دهند. برای اطلاعات بیشتر در خصوص این استاندارد به آدرس http://grouper.ieee.org/groups/754/ مراجعه کنید. همان‌گونه که ممکن است حدس زده باشید، اعداد اعشاری به دلیل پیچیدگی خاصی که دارند حافظه بیشتری را مصرف می‌کنند. علاوه بر این، آن‌ها از یک بخش متفاوت و خاصی از پردازنده‌ها استفاده می‌کنند که سرعت آن کمتر از بخشی است که برای محاسبه مقادیر صحیح از آن استفاده می‌شود. در نهایت، اعداد صحیح دقیق هستند، در حالی که اعداد اعشاری را نمی‌توان برای نمایش یک مقدار

دقیق استفاده کرد. در نتیجه برای بیان مقادیر اعشار همواره از مقدار تقریبی استفاده کرده یا در صورت لزوم یک مقدار اعشاری را گرد می‌کنید. با این حال، نوع‌های داده‌ای اعشاری می‌توانند مقادیر خیلی بزرگ را ذخیره‌سازی کنند. با توجه به این‌که در دنیای واقعی شما برای انجام برخی از کارها به محاسبات خیلی دقیقی نیاز دارید، وجود مقادیر اعشاری اجتناب‌ناپذیر است. اما با استفاده از مقادیر صحیح می‌توانید میزان حافظه موردنیاز را کم کرده و سرعت پردازش‌ها را سریع‌تر کنید. دقت کنید در زمان برنامه‌نویسی باید هر نوع داده‌ای را در جای درست خود استفاده کنید.

مقادیر بولین (منطقی)

شاید از شنیدن این حرف تعجب کنید، اما کامپیوترها همیشه یک پاسخ مستقیم و سرراست به شما می‌دهند! یک کامپیوتر هیچ‌گاه به شما پاسخ ممکن است را نخواهد گفت. هر پاسخی که شما از یک کامپیوتر دریافت می‌کنید یکی از دو وضعیت درست (True) یا اشتباه (False) را دارد. در دنیای ریاضیات شاخه‌ای به نام جبر بول (Boolean) وجود دارد که اولین بار از سوی جورج بول تعریف شد. همه کامپیوترها بر مبنای جبر بول تصمیم‌گیری می‌کنند. بر خلاف باور بسیاری از مردم جبر بول خیلی پیش‌تر از اختراع کامپیوتر و از سال 1854 وجود داشته است.

مشاهده نوع داده‌ای یک متغیر

برخی موارد نیاز دارید در مورد نوع یک متغیر اطلاعی داشته باشید. در برخی موارد این امکان وجود دارد که از طریق نگاه کرده به کدها نوع یک متغیر را به درستی مشاهده کرد. برای مشاهده نوع متغیرها در **پایتون** باید از متد type() استفاده کنید. به‌طور مثال، اگر مقدار 5 را درون متغیری به نام myInt قرار داده و از ترکیب نحوی myInt=5 استفاده کرده‌اید، برای مشاهده نوع این متغیر کافی است از فرمان type(myInt) استفاده کرده و کلید اینتر را فشار دهید تا نوع متغیر را مشاهده کنید. خروجی این فرمان برابر با عبارت <’class ‘int> است که نشان می‌دهد متغیر شما یک مقدار صحیح را نگه‌داری می‌کند. برای مشاهده نوع هر متغیر استفاده شده در **پایتون** از این ترکیب می‌توانید استفاده کنید.

زمانی که از یک مقدار بولین در پایتون استفاده می‌کنید، در حقیقت در حال استفاده از نوع bool است. هر متغیر منطقی تنها یکی از دو مقدار True یا False را شامل می‌شود. شما نیز در زمان استفاده از این متغیرها می‌توانید یکی از دو کلیدواژهTrue یا False را استفاده کرده یا عبارتی را بنویسید که یک مقدار هم ارز با true یا false را تولید کند. به‌طور مثال، ترکیب زیر را تصور کنید.

```
myBool=1>2
```

خروجی این مقدار برابر با False خواهد بود، زیرا 1 هیچ‌گاه بزرگ‌تر از 2 نخواهد بود. در این سری از آموزش‌ها به اشکال مختلفی از متغیرهای منطقی استفاده خواهیم کرد.

رشته‌ها در پایتون

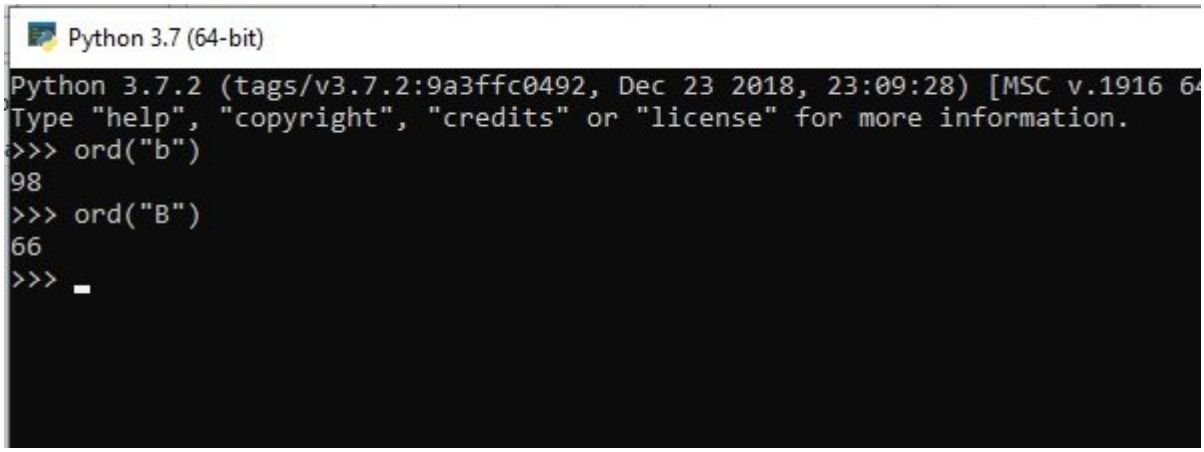
در دنیای برنامه‌نویسی رشته‌ها در مقایسه با سایر نوع‌های داده‌ای به سادگی درک می‌شوند و پیچیدگی خاصی ندارند، اما کامپیوترها هیچ‌گونه شناختی از رشته‌ها ندارند. اگر مقاله‌های قبلی این سری را مطالعه کرده باشید، قبلا رشته‌ها را مشاهده کرده‌اید. زمانی که از فرمان prtint() برای چاپ متن روی صفحه استفاده می‌کردیم، رشته‌ها را درون کاراکترهای نقل قول/کوتیشین‌ها "" قرار می‌دادیم و از آن‌ها استفاده می‌کردیم. به‌طور مثال، متغیر زیر را تصور کنید.

```
myString ="Python is a great language."
```

اختصاص یک رشته از کاراکترها به هر متغیری با استفاده از کاراکترهای نقل قول انجام می‌شود. دقت کنید در این حالت کامپیوتر هیچ‌گونه کاراکتری را مشاهده نمی‌کند. هر کاراکتری که شما از آن استفاده می‌کنید به معادل عددی آن تبدیل شده و درون حافظه کامپیوتر قرار می‌گیرد. به‌طور مثال کاراکتر A دارای کد 65 در سیستم دهگان است. برای آن‌که معادل عددی این کاراکتر یا هر کاراکتری را مشاهده کنید پایتون فرمان ord (“”)را در اختیارتان قرار می‌دهد. به‌طور مثال اگر فرمان زیر را در محیط خط فرمان پایتون تایپ کرده و کلید اینتر را فشار دهید، پایتون مقدار 65 را در خروجی به شما نشان خواهد داد. (دقت کنید کدهای عددی کاراکترهای بزرگ و کوچک متفاوت از

یکدیگر هستند.)

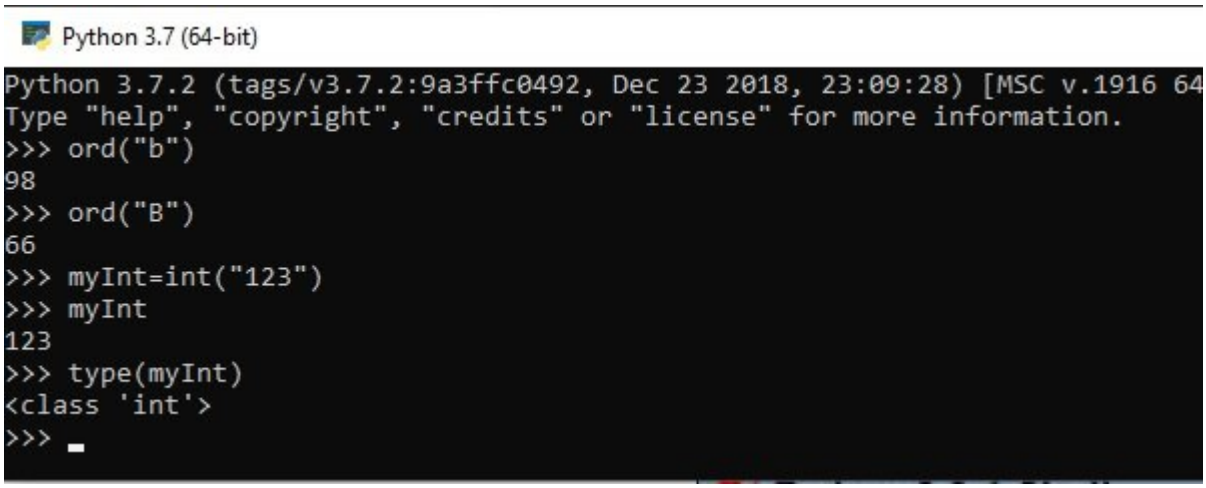
```
ord(“A”)
```



از آن‌جایی که کامپیوترها واقعا رشته‌ها را درک نمی‌کنند، اما رشته‌ها یک نقش کلیدی در دنیای برنامه‌نویسی دارند، گاهی اوقات لازم است تا یک رشته به معادل عددی آن تبدیل شود. شما می‌توانید از فرمان‌های ()int و ()float برای انجام این تبدیل استفاده کنید. به‌طور مثال، اگر در محیط خط فرمان **پایتون** عبارت زیر را تایپ کرده و کلید اینتر را فشار دهید، شما یک رشته را به مقدار عددی تبدیل کرده و درون متغیر myInt قرار داده‌اید.

```
myInt=int(“123”)
```

دقت کنید زمانی که هرگونه کاراکتری را درون علامت‌های کوتیشین قرار می‌دهید در حقیقت در حال تایپ یک رشته هستید. در مثال بالا شما رشته 123 را به معادل عددی تبدیل کرده و درون متغیر فوق قرار داده‌اید.



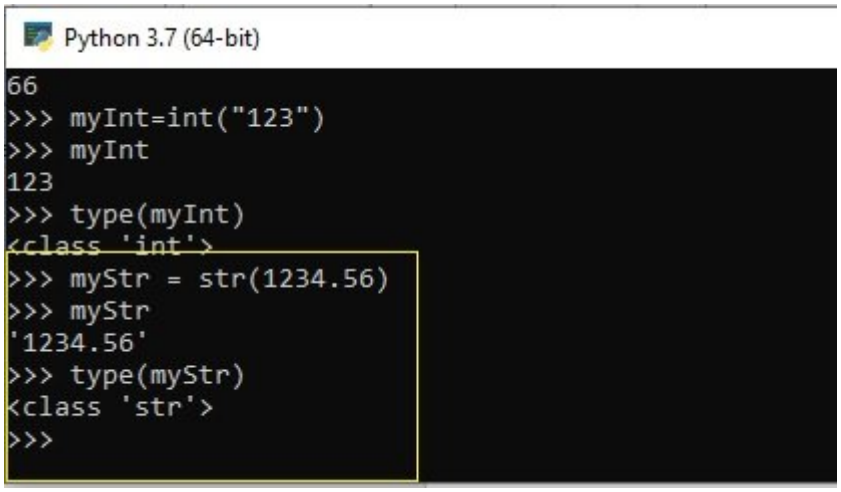
در تصویر بالا ما یک رشته به نام 123 را به معادل عددی آن تبدیل کرده و درون متغیر myInt قرار دادیم. در ادامه متغیر myInt را فراخوانی کردیم که مقدار 123 را روی صفحه‌نمایش نشان داد. در نهایت از فرمان ()typeاستفاده کردیم تا نوع داده‌ای متغیر myInt را مشاهده کنیم.

برعکس این قضیه نیز صادق است. شما می‌توانید یک مقدار عددی را به معادل رشته‌ای آن تبدیل کنید. برای این منظور فرمان ()str در اختیارتان قرار دارد. به‌طور مثال، اگر فرمان زیر را تایپ کرده و کلید اینتر را فشار دهید، مقدار فوق به معادل رشته‌ای آن تبدیل می‌شود.

```
myStr = str(1234.56)
```

با اجرای فرمان فوق رشته‌ای به نام 12345.56 درون متغیر myStr قرار می‌گیرد. شکل زیر این نوع از تبدیل را

نشان می‌دهد.



در مقاله‌های آتی مشاهده خواهید کرد که این تبدیل نوع‌ها انجام برخی کارهای غیرممکن را امکان‌پذیر می‌کند.

کار با تاریخ و زمان

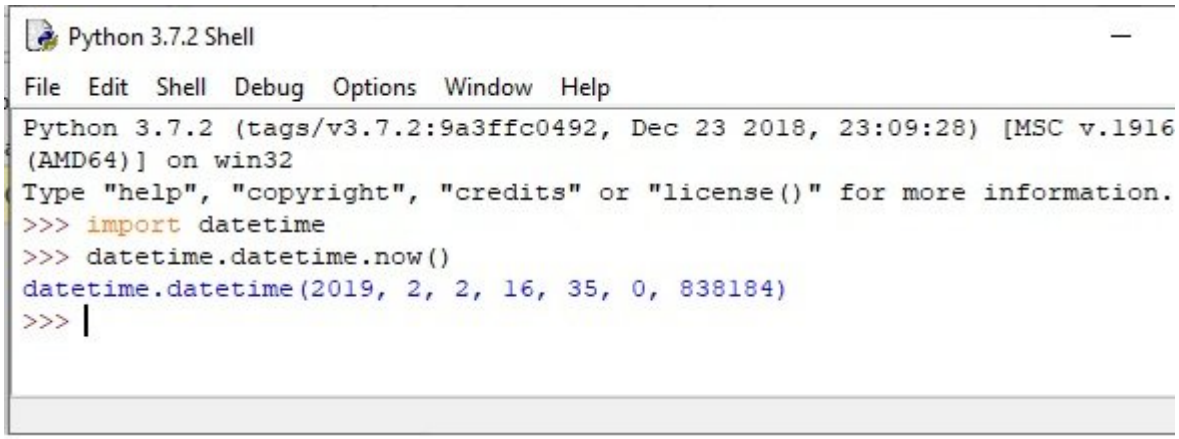
تاریخ و زمان جزء آن گروه از نوع‌های داده‌ای هستند که بیشتر مردم با آن‌ها کار می‌کنند. زندگی ما بر مبنای هماهنگ شدن با تاریخ و زمان حرکت می‌کند. ما قرار ملاقات‌ها و رویدادها را بر مبنای تاریخ و زمان مشخصی تنظیم می‌کنیم. با توجه به اهمیت فوق‌العاده زیاد تاریخ و زمان در زندگی ما، زبان‌های برنامه‌نویسی سعی کرده‌اند نوع‌های داده‌ای قدرتمندی برای کار کردن با تاریخ و زمان در اختیار برنامه‌نویسان قرار دهند. اما فراموش نکنید که کامپیوترها هیچ ذهنیتی در ارتباط با تاریخ و زمان ندارند، زیرا آن‌ها فقط اعداد را درک می‌کنند. برای کار کردن با تاریخ و زمان، شما باید در **پایتون** یک کار اضافی انجام دهید. برای آن‌که بتوانید با نوع‌های داده‌ای کار کنید باید فرمان import datetime را در ابتدای کدهای خود قرار دهید. به لحاظ تکنیکی به این فرمان وارد کردن یک ماژول گفته می‌شود. در مقاله‌های آتی با نحوه کار کردن با ماژول‌ها و وارد کردن آن‌ها در برنامه‌های کاربردی بیشتر آشنا خواهید شد. در حال حاضر همین که بدانید برای کار کردن با تاریخ و زمان باید ماژول مربوطه را چگونه وارد کنید کافی است.

کامپیوترها از یک ساعت داخلی استفاده می‌کنند. اما ساعت برای این منظور در کامپیوترها قرار گرفته است که انسان‌ها بتوانند از کامپیوترها استفاده کنند، هرچند برخی از نرم‌افزارها نیز از ساعت کامپیوتر استفاده می‌کنند. برای آن‌که ساعت جاری را مشاهده کنید **پایتون** فرمان datetime.datetime.now() را در اختیارتان قرار می‌دهد. این فرمان اطلاعات کامل تاریخ و ساعت جاری سیستم را به شما نشان می‌دهد.

برای مشاهده تاریخ و ساعت جاری کامپیوتر خود محیط IDLE را باز کرده و دستورات زیر را درون آن تایپ کنید.

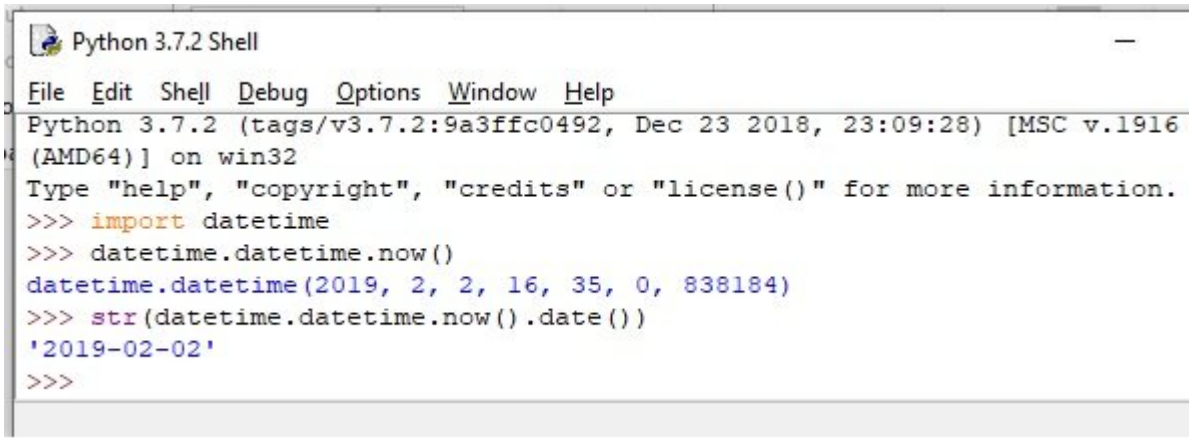
```
Import datetime

datetime.datetime.now()
```

خروجی دستورات فوق همانند شکل زیر خواهد بود:

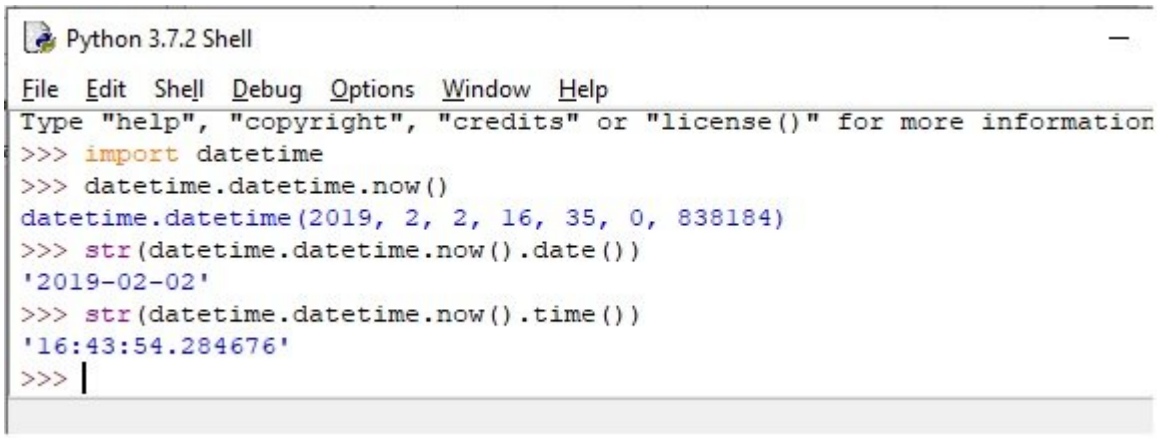


همان‌نه که در شکل بالا مشاهده می‌کنید، خواندن تاریخ و ساعت به این شکل کار مشکلی است. برای آن‌که بتوانید تاریخ و ساعت را به شکل قابل فهم‌تری مشاهده کنید، باید این خروجی را ویرایش کنید. اکنون زمان آن فرا رسیده است اطلاعاتی که تا این مرحله به دست آورده‌اید را با یکدیگر ترکیب کرده و این خروجی را ویرایش کنید. برای ویرایش این خروجی ما از تکنیک تبدیل تاریخ و زمان به یک رشته که در پاراگراف قبل به آن اشاره کردیم استفاده می‌کنیم. در نتیجه فرمان فوق را به شکل زیر ویرایش می‌کنیم.

Str(datetime.datetime.now().date())



همان‌گونه که مشاهده می‌کنید، اکنون تاریخ به شکل بهتری مشاهده می‌شود. اگر در فرمان فوق به جای date() از time() استفاده کنید، شما ساعت جاری را به شکل بهتری مشاهده می‌کنید. شما برای هر یک از مولفه‌های روز، ماه، سال، ساعت، دقیقه، ثانیه و میکروثانیه دستورات متناظر (day, month, year, hour ,minute, second و microsecond) را دارید که به شکل تفکیک شده این اطلاعات را در اختیارتان قرار می‌دهند. اگر به جای date در فرمان فوق از time استفاده کنید، خروجی ساعت جاری سیستم‌تان خواهد بود.



در این بخش از آموزش **پایتون** شما را با نوع‌های داده‌ای آشنا کردیم. در شماره آینده به سراغ نحوه نمایش داده‌ها، عملگرها و مطالب دیگر خواهیم رفت.

معرفی کانال آموزش برنامه‌نویسی و پایتون:

برنامه‌نویسی | پایتون: @Python_0to100
(کانال سری آموزش های رایگان پایتون)

آموزش پایتون: @learnpy
(آموزش پایتون با فلش کارت)

تاریخ انتشار:
14 بهمن 1397

نشانی منبع:
https://www.shabakeh-mag.com/workshop/programming/14536/%D8%A2%D9%85%D9%88%D8%B2%D8%B4-%D8%B1%D8%A7%DB%8C%DA%AF%D8%A7%D9%86-%D9%BE%D8%A7%DB%8C%D8%AA%D9%88%D9%86-python-%E2%80%93-%D8%A2%D8%B4%D9%86%D8%A7%DB%8C%DB%8C-%D8%A8%D8%A7-%D9%86%D9%88%D8%B9-%D9%87%D8%A7%DB%8C-%D8%AF%D8%A7%D8%AF%D9%87%E2%80%8C%D8%A7%DB%8C-%D8%B1%D8%B4%D8%AA%D9%87%E2%80%8C%D8%A7%DB%8C%D8%8C%D9%85%D9%86%D8%B7%D9%82%DB%8C-%D9%88