فصل سه

دنیای متغیرها

و عملگرها

متغیر ها چه هستند ؟

متغیر ها اسامی هستند که به داده‌های مورد نیاز در برنامه خود برای دستکاری اختصاص می‌دهید . برای مثال فرض کنید که برنامه ما می‌خواهد سن یک کاربر را ذخیره کند . به این منظور ما می‌توانیم این داده را با نام userAge ذخیره کنیم و به صورت زیر متغیر را تعریف کنیم :

```
userAge = 0
```

پس از آنکه متغیر userAge را تعریف نمودید ، برنامه شما بخشی از فضای حافظه کامپیوتر شما را به منظور ذخیره این داده در نظر می‌گیرد . سپس شما می‌توانید به این داده با اشاره به نام آن ، دسترسی پیدا کنید یا آن را تغییر دهید . هر زمان که شما یک متغیر جدید ایجاد می‌کنید ، بایستی به آن یک مقدار اولیه بدهید . در این مثال ما به آن مقدار عدد صفر را دارد . ما همیشه این مقدار را می‌توانیم در برنامه خود تغییر دهیم . همچنین می‌توانیم چندین متغیر را به صورت یکدفعه تعریف کنیم . به این منظور به سادگی از ساختار نوشتاری زیر استفاده می‌کنیم :

```
userAge , userName = 30 , 'Peter'
```

این نوع تعریف و مقدار دهی معادل تعریف و مقدار دهی به صورت زیر می‌باشد .

```
UserAge = 30
userName = 'Peter'
```

در دو مثال بالا یک کار را به دو شیوه متفاوت انجام دادیم و آن تعریف دو متغیر userAge و userName بود و به ترتیب مقدار دهی عدد ۳۰ و مقدار رشته ای Peter به آن دو

نام گذاری یک متغیر در پایتون

نام یک متغیر در پایتون فقط می‌تواند حروف (a-z و A-Z) ، اعداد و آندرلاین باشد .
هرچند حرف اول متغیرها را نمی‌توان یک عدد گذاشت . مثلاً می‌توان متغیرهایی با نام های user_name،userName یا userName2 ایجاد کرد ولی نمی‌توان متغیری با نام 2userName ساخت .
به علاوه برخی کلمات رزور شده وجود دارند که شما نمی‌توانید آن‌ها را به عنوان نام متغیر استفاده کنید . چرا ؟ به این دلیل که آن‌ها قبلاً معنای مشخصی در زبان پایتون دارند . این کلمات رزرو شده شامل print، input ، if، while و ... هستند .
درباره هر کدام از آن‌ها در درس‌های بعدی توضیح خواهیم داد .
نکته آخر و مهم اینکه نام متغیرها در زبان برنامه نویسی پایتون نسبت به حروف و بزرگ و کوچک حساس است . مثلاً username و userName یکی نیستند .
این‌ها دو متغیر جداگانه در زبان پایتون محسوب می‌شوند چرا که دارای حروف بزرگ و کوچک متفاوتی هستند . در نام گذاری متغیرها شما می‌توانید از استایل camelCase و یا آندرلاین نیز استفاده کنیم که اختیاری می‌باشند .

علامت واگذاری

به یاد داشته باشید که علامت = در عبارت userAge = 0 معنی متفاوتی از آنچه که ما در ریاضیات یادگرفتیم دارد . در برنامه نویسی علامت = مساوی با نام Assignment به معنی واگذاری شناخته می‌شود . به این معنی که ما داریم مقداری را در سمت راست علامت = به متغیر اختصاص می‌دهیم . یک راه خوب برای درک این عبارت userAge = 0 این است که آن را مشابه عبارت userAge <- 0 در نظر بگیریم . در برنامه نویسی دو عبارت x = y و y = x معانی کاملاً متفاوتی دارند در حالی که در ریاضیات این دو برابرند . به همین دلیل است که من همیشه به استادان ریاضی عرض می‌کنم که ریاضیات هیچ کاربردی در برنامه نویسی ندارد ;)

گیج شدید ؟ یک مثال راهگشاست .

کد زیر را درون ادیتور وارد کنید و آن را ذخیره کنید :

```
x = 5
y = 10
x = y
print ("x = ", x)
print ("y = ", y)
```

اکنون برنامه را با فرمت py. ذخیره کرده و آن را اجرا کنید . شما بایستی خروجی زیر را دریافت کنید .

خروجی بدست آمده به صورت زیر خواهد بود :

```
x = 10
y = 10
```

هرچند که x دارای یک مقدار اولیه ۵ می‌باشد (که در خط اول تعیین شده است) ، خط سوم x = y مقدار y را به x نسبت می‌دهد . نه اینکه x را برابر y قرار دهد .

بلکه مقدار y را درون متغیر x می‌ریزد . چون مقدار y ۱۰ می‌باشد پس مقدار x هم ۱۰ می‌شود . حالا عبارت زیر چطور ؟

```
x = 5
y = 10
y = x
print ("x = ", x)
print ("y = ", y)
```

تفاوت چیست ؟ فقط در خط سوم جای x و y را عوض کردیم . درواقع این بار به جای اینکه مقدار متغیر y یعنی ۱۰ درون x بریزد و هر دو مقدار 10 را داشته باشند ، مقدار متغیر x یعنی 5 درون متغیر y می‌ریزد و هر دو مقدار ۵ را خواهند داشت .

عملگرهای اصلی

علاوه بر واگذاری یک متغیر به یک مقدار اولیه ما می‌توانیم عملگرهای محاسباتی را بر روی متغیر ها انجام دهیم. عملگرهای اصلی در پایتون شامل + , - , * , / , // , % و ** می‌باشند که به ترتیب معادل جمع ، تفریق ، ضرب ، تقسیم ، تقسیم کف ، باقی‌مانده و توان می‌باشند. به مثال های زیر دقت کنید. فرض کنید که x=5 و y=2 موارد زیر را بررسی کنید :

```
x + y = 7 جمع
x - y = 3 تفریق
x * y = 10 ضرب
x / y = 2.5 تقسیم
x // y = 2 تقسیم کف (پاسخ را به پایین رند می کند)
x % y =1 (باقی مانده تقسیم 5 بر ۲ را یعنی عدد ۱ را به شما می دهد)
x**y = 25 توان
```

دیگر عملگرهای واگذاری

در کنار عملگر = دیگر عملگرهای واگذاری در پایتون (و دیگر زبان‌های برنامه نویسی) وجود دارند . این عملگرها شبیه =+ و =- و =* می‌باشند .

فرض کنید که متغیر x با یک متغیر اولیه ۱۰ را داریم . اگر که بخواهیم به صورت افزایش بالا ببریم مثلاً ۲ تایی می‌توانیم بنویسیم ، x = x + 2

این برنامه ابتدا عبارت را در سمت راست ارزیابی می‌کند و سپس جواب را به مقدار x در سمت چپ واگذار می‌کند . پس در نهایت عبارت بالا مقدار ۱۲ را به متغیر x می‌دهد .

حال به جای عبارت بالا می‌توانیم بنویسیم x += 2 که دقیقاً همان عمل‌کرد و معنی را دارد . درواقع علامت =+ یک عملگر کوتاه نویسی است که از ترکیب نشانه واگذاری با عملگر جمع پدید می‌آید . به صورت مشابه این شیوه خلاصه نویسی برای دیگر عملگرها مثل تفریق و ... نیز وجود دارند .

فصل چهار

انواع داده ای

در پایتون

انواع داده‌ای

خوب متغیرها و انواع داده‌ای در پایتون را یادگرفتیم . در ادامه ابتدا به توضیح انواع داده‌ای در پایتون می‌پردازیم . به ویژه اینتجرها ، فلوت ها و رشته‌ها . سپس مفاهیمType Casting را بررسی کرده و سه نوع داده پیشرفته را در پایتون بررسی می‌کنیم یعنی tuple،list و دیکشنری

اینتجرها Integers

اعدادی بدون بخش اعشاری می‌باشند مثل 7 , 5 , 2 , 0 , -4 , -7 و ...

به منظور اعلان یک عدد اینتجر در پایتون به سادگی نام متغیر را برابر مقدار عددی قرار دهیم مثل :

userAge = 20 , mobileNumber = 09121111111

فلوت Float

فلوت ها اشاره به اعدادی دارند که دارای بخش اعشاری هستند مثل 1.234 و 0.0232 و 2.12

به منظور اعلان یک عدد فلوت در پایتون به صورت مثال زیر عمل می‌کنیم :

userHeight = 1.82 , userWeight = 67.2

رشته String

رشته یا استرینگ به متن اشاره می‌کند . برای اعلان یک مقدار رشته بایستی مقدار را درون تک کوتیشن یا دابل کوتیشن قرار دهیم :

```
variableName = " مقدار اولیه "
variableName = ' مقدار اولیه '
```

مثال

```
userName = 'Emily' , Family = 'Van Camp' , userAge = '28'
```

در مثال بالا سه متغیر را تعریف کردیم . ابتدا به نام کاربری مقدار emily را اختصاص داده و سپس به نام خانوادگی Van Camp و در آخر متغیر سن را ۲۸ تعیین کردیم .
نکته اینکه چون مقدار عددی سن یعنی عدد ۲۸ را درون کوتیشن قرار دادیم اکنون مقدار متغیر userAge یک مقدار رشته ای می‌باشد . در مقابل اگر که نوشته بودید userAge = 30 (بدون کوتیشن) مقدار متغیر userAge اینتجر بود .
ما می‌توانیم چندین زیر رشته را با استفاده از علامت + به هم الحاق کنیم . برای مثال :

```
'' Emily '' + '' VanCamp ''
```

این عبارت معادل مقدار " Emily VanCamp " را به شما می‌دهد .

توابع درون ساخت رشته ای

پایتون شامل تعدادی تابع درون ساخت به منظور دستکاری رشته‌ها می‌باشد . تابع به معنی بلوکی از کدهای قابل استفاده مجدد است که وظایف خاصی را انجام می‌دهند .

جلوتر درباره توابع بیشتر صحبت خواهیم کرد . مثالی از یک تابع موجود در پایتون ، تابع upper() می‌باشد . این تابع به منظور بزرگ کردن همه حروف یک رشته خاص به کار می‌رود . برای مثال اگر رشته Emily را داشته باشید با استفاده از این تابع به صورت زیر :

```
'Emily'.upper()
```

خروجی EMILY را خواهیم داشت .

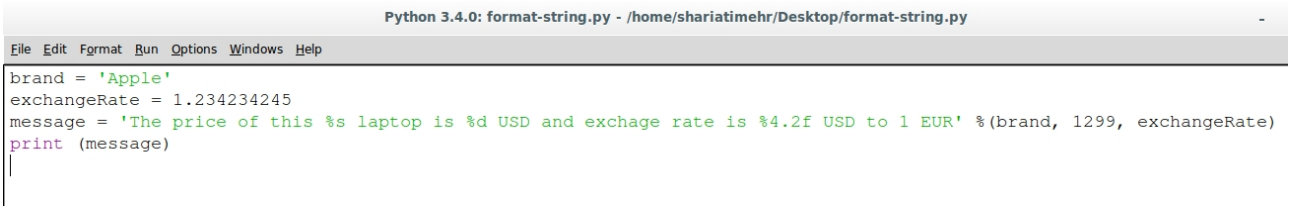
فرمت دهی رشته ها با عملگر %

رشته‌ها را می‌توان با استفاده از عملگر % فرمت دهی کرد . این عملگر به شما کنترل بیشتری بر روی نحوه نمایش و ذخیره سازی رشته می‌دهد . سینتکس استفاده شما برای این عملگر به صورت زیر می‌باشد :

" String to Format " % (مقادیر یا متغیرهایی که درون رشته درج می‌شوند و با کاما از هم جدا می‌شوند)

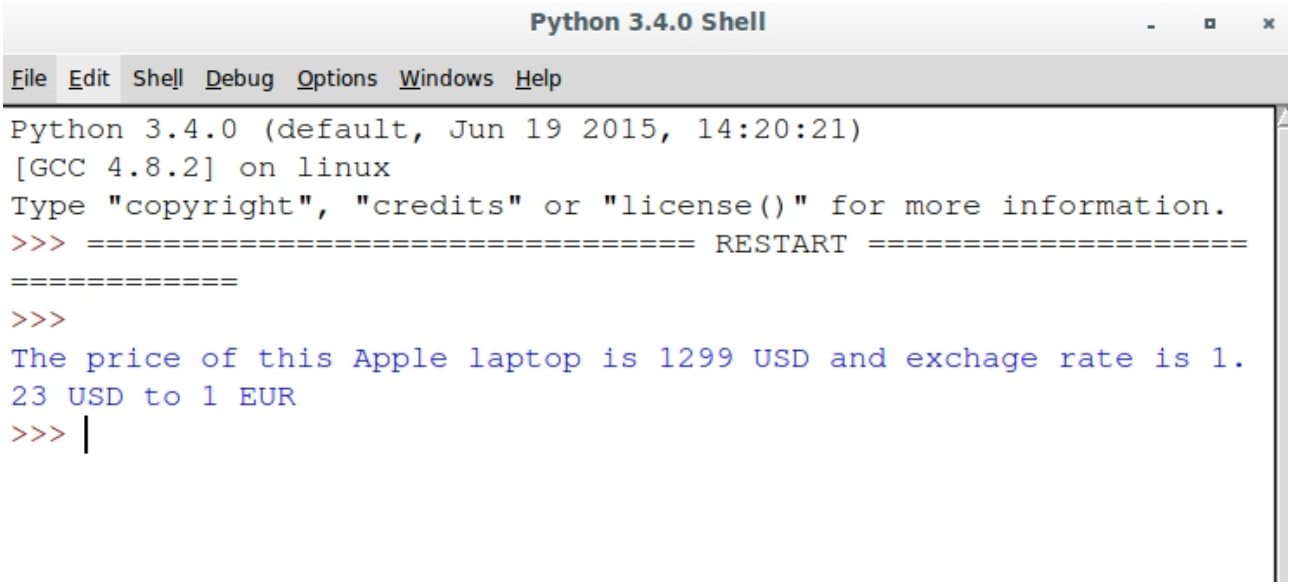
این سینتکس سه بخش دارد . بخش اول رشته ای که می‌خواهیم فرمت دهی کنیم که درون کوتیشن قرار می‌گیرد . سپس علامت درصد % را قرار می‌دهیم . بخش سوم درون پرانتز مقادیر یا متغیرهایی که می‌خواهیم درون رشته درج کنیم را قرار می‌دهیم . کد زیر را درون IDLE وارد کنید و اجرا کنید :

```
brand = 'Apple'
exchangeRate = 1.234234245
message = 'The price of this %s laptop is %d
USD and exchage rate is %4.2f USD to 1 EUR'
%(brand, 1299, exchangeRate)
print (message)
```



در مثال بالا رشته The price of this %s laptop is %d USD and exchage rate is %4.2f USD to 1 EUR که درون کوتیشن قرار دارد ، رشته ای است که می‌خواهیم آن را فرمت دهی کنیم . ما از فرمت دهنده های %s و %d و %4.2f به عنوان جایگاه های خود درون متن استفاده کرده . سپس علامت درصد را آورده و مقادیر موجود درون پرانتز را یعنی متغیرbrand و مقدار 1299 و متغیر exchangeRate را به ترتیب درون جایگاه ها قرار می‌دهیم .

اگر خروجی بگیریم مقدار زیر حاصل می‌شود :



فرمت دهنده %s به منظور ارایه یک رشته استفاده می‌شود در اینجا مثلاً Apple .
فرمت دهنده %d ارایه کننده یک مقدار عددی اینتجر می‌باشد در اینجا 1299 . اگر بخواهیم قبل از مقدار عددی فاصله اضافه کنیم ، می‌توانیم یک شماره بین علامت % و حرف d وارد کنیم که نشان دهنده طول رشته مورد نظر است .

برای نمونه :

```
''%5d'' %(123)
```

به ما مقدار "123 " به همراه دو فاصله در ابتدای آن را خواهد داد . یعنی طول کاراکتر ما یعنی 123 مقدار سه می‌باشد و ما بهش عدد ۵ رو دادیم . پس ۲ فاصله در ابتدا ایجاد می‌کند .

فرمت دهنده %f به منظور فرمت دهی فلوت ها (اعداد اعشاری) به کار می‌رود . در اینجا ما %4.2f را داریم که عدد ۴ اشاره به طول کل رشته و عدد ۲ اشاره به دو جایگاه اعشاری دارد . یعنی طول کل رشته ما ۴ کاراکتر که دو رقم آن اعشاری است . اگر بخواهیم قبل از عدد فاصله اضافه کنیم می‌توانیم به صورت %7.2f استفاده کنیم که در این حالت سه فاصله به کل طول رشته در ابتدای آن اضافه می‌شود .

فرمت دهی رشته ها با تابع Format

علاوه بر استفاده از عملگر ٪ برای فرمت دهی رشته‌ها ، پایتون متد دیگری برای فرمت دهی رشته‌ها را به ارایه می‌دهد که سینتکس آن به صورت زیر می‌باشد :

```
''string to be formatted''.format
```

(مقادیر یا متغیرهایی برای درج در رشته که با کاما جدا می‌شوند)

وقتی که متد format استفاده می‌کنیم دیگر از نگهدارنده های %s یا %f یا %d استفاده نمی‌کنیم . در عوض از آکولاد به صورت زیر استفاده می‌کنیم :

```
message = 'The price of this {0:s} laptop is
{1:d} USD and the exchange rate is {2:4.2f}
USD to 1 EUR'.format('Apple', 1299,
1.235235245)
```

درون آکولاد ابتدا جایگاه پارامتر مورد استفاده را می نویسیم سپس دو نقطه . پس از دو نقطه فرمت دهنده را وارد می‌کنیم . درون آکولاد نبایستی فاصله‌ای قرار داد .
وقتی که عبارت ('Apple', 1299, 1.235235245) را می نویسیم ، درواقع سه پارامتر را به تابع format پاس می‌دهیم .پارامترها داده‌هایی هستند که متد به منظور انجام وظایفش به آن‌ها نیاز دارد . پارامتر Apple دارای دارای موقعیت 0 می‌باشد ، پارامتر ۱۲۹۹ دارای موقعیت 1 و پارامتر 1.235235245 دارای موقعیت 2 می‌باشد . موقعیت ها از ایندکس صفر شروع می‌شوند .

وقتی که می نویسیم {s:0} از مفسر می‌خواهیم که این پارامتر را با موقعیت پارامتری ۰ که یک مقدار رشته ای است جایگزین کنیم (فرمت دهنده ما s می‌باشد چون مقدار ما رشته ای است).

وقتی که ما می نویسیم {d:1} به پارامتر موجود در موقعیت ۱ که یک مقدار عددی است (فرمت دهنده ما هم d می باشد) اشاره می‌کنیم .
وقتی عبارت {2:4.2f} را می نویسیم . به پارامتر موقعیت ۲ که مقداری اعشاری است اشاره می‌کنیم و می‌خواهیم که با دو جایگاه اعشاری و طول کل ۴ رشته فرمت دهی کنیم

اگر که مقدار message را چاپ کنیم خروجی به صورت زیر خواهد بود :

```
The price of this Apple laptop is 1299 USD
and the exchange rate is 1.24 USD to 1 EUR
```

نکته : اگر که نمی‌خواهیم رشته را فرمت دهی کنید و فقط می‌خواهید جایگذاری کنید می‌توانید به صورت زیر بنویسیم :

```
message = 'The price of this {} laptop is {}
USD and the exchange rate is {} USD to 1
EUR'.format('Apple', 1299, 1.234234245)
```

در این حالت نیازی به قرار دادن موقعیت پارامتر ها نداریم و مفسر پایتون به صورت خودکار به ترتیب پارامترها را جایگذاری خواهد کرد .

متد format() برای مبتدی ها ممکن است گیج‌کننده باشد . در حقیقت فرمت دهی رشته‌ها می‌تواند بیش از این پیچیده باشند ولی آنچه در اینجا گفتیم در اکثر موارد و بیشتر اهداف کارساز خواهد بود . برای درک بهتر از متد format() برنامه زیر را امتحان کنید :

```
message1 = '{0} is easier than
{1}'.format('Python', 'Java')
message2 = '{1} is easier than
{0}'.format('Python', 'Java')
message3 = '{:10.2f} and
{:d}'.format(1.234234234, 12)
message4 = '{}'.format(1.234234234)

print (message1)
#You will get 'Python is easier tha Java'

print (message2)
#You will get 'Java is easier Python'

print (message3)
#You will get '       1.23 and 12'
#You dont need to indicate the positions of
the parameters.

print (message4)
#You will get '1.234234234. no formatting is
done.'
```

Python 3.4.0: format-example.py - /home/shariatimehr/Desktop/format-example.py

File Edit Format Run Options Windows Help

```
message1 = '{0} is easier than {1}'.format('Python', 'Java')
message2 = '{1} is easier than {0}'.format('Python', 'Java')
message3 = '{:10.2f} and {:d}'.format(1.234234234, 12)
message4 = '{}'.format(1.234234234)

print (message1)
#You will get 'Python is easier tha Java'

print (message2)
#You will get 'Java is easier Python'

print (message3)
#You will get '      1.23 and 12'
#You dont need to indicate the positions of the parameters.

print (message4)
#You will get '1.234234234. no formatting is done.'
```

Ln: 3 Col: 5

خروجی به صورت زیر خواهد بود :

Python 3.4.0 Shell

File Edit Shell Debug Options Windows Help

```
Python 3.4.0 (default, Jun 19 2015, 14:20:21)
[GCC 4.8.2] on linux
Type "copyright", "credits" or "license()" for more information.
>>> ================================ RESTART ================================
============
>>>
Python is easier than Java
Java is easier than Python
      1.23 and 12
1.234234234
>>>
```

Ln: 10 Col: 4