۲-۷- رشته ها در پایتون

رشته ها در پایتون عبارت است از مجموعه کاراکترهای همجوار که در علامت نقل قول نمایش داده می شوند .
پایتون از هر دو شکل تک کوتیشن و دابل کوتیشن پشتیبانی می کند .می توان بابهره گیری از عملگر برش یا ([ ] [:]) slice operator که اندیس آن در آغاز رشته با اندیس 0 شروع شده و تا - 1 در انتها ادامه می یابد، بخش هایی از یک رشته را استخراج کرد.
علامت (+) یک عملگر اتصال است که دو رشته را به هم پیوند می دهد .
علامت (*) درواقع یک repetition operator است که دستوری را تکرار می کند(برای مثال یک رشته را دوبار چاپ مینمایند.)

مثال:

```
str = 'Hello World!'
print (str)        # Prints complete string
print (str[0])     # Prints first character of the string
print (str[2:5])   # Prints characters starting from 3rd to 5th
print (str[2:])    # Prints string starting from 3rd character
print (str * 2)    # Prints string two times
print (str + "TEST")    # Prints concatenated string
```

نتیجه ی زیر حاصل می گردد:

```
Hello World!
H
llo
llo World!
Hello World!Hello World!
Hello World!TEST
```

۳-۷- نوع داده ای List در پایتون

از میان نوع های داده ای پایتون، List ها تطبیق پذیرترین نوع داده ای هستند، بدین معنا که برای منظورهای مختلف می توان از آن ها بهره گرفت .یک لیست شامل مجموعه ای از آیتم ها است که توسط ویرگول از هم جدا شده و داخل [] محصور می شوند .تا حدی می توان گفت که List شبیه به نوع داده ای آرایه در زبان C است .یک تفاوت اساسی بین آرایه و لیست این است که آیتم های موجود در لیست می توانند از نوع داده های مختلف باشند(از نظر نوع با هم متفاوت باشند.)

مقادیر ذخیره شده در یک لیست را می توان با استفاده از عملگر برش ([ ] [:]) از طریق اندیس که در ابتدای لیست ار صفر آغاز شده و تا - 1 در انتهای لیست ادامه می یابد، مورد دسترسی قرارداد .علامت + به عنوان یک عملگر اتصال نقش ایفا کرده و عملگر * نیز صرفا یک دستور را تکرارمی کند

مثال:

```
list = [ 'abcd', 786 , 2.23, 'john', 70.2 ]
tinylist = [123, 'john']
print (list)            # Prints complete list
print (list[0])         # Prints first element of the list
print (list[1:3])       # Prints elements starting from 2nd till 3rd
print (list[2:])        # Prints elements starting from 3rd element
print (tinylist * 2)    # Prints list two times
print (list + tinylist) # Prints concatenated lists
```

کد فوق، نتیجه ی زیر را بدست می دهد:

```
['abcd', 786, 2.23, 'john', 70.2]
 abcd
 [786, 2.23]
 [2.23, 'john', 70.2]
 [123, 'john', 123, 'john']
 ['abcd', 786, 2.23, 'john', 70.2, 123, 'john']
```

٤-٧-نوع داده ای Tuple در پایتون

Tuple یا چندتایی ها نیز یک نوع داده ای متشکل از رشته یا مجموعه ای از آیتم هاست که مشابه نوع داده ای **List** می باشد .یک **Tuple** تعدادی مقادیر را در خوش دارد که این مقادیر توسط ویرگول از هم جدا می شوند .اما برخلاف **List** ، نوع داده ای **Tuple** داخل پرانتز محصور می شود.

بین نوع داده ای مذکور تفاوت هایی وجود دارد :در **List** مقادیر درون [] جای می گیرند، در حالی که این مقادر در **tuple** داخل پرانتز محصور می شوند .تفاوت دیگر این است که المان های **List** واندازه ی آن را می توان اصلاح نمود ولی این امکان برای **tuple** وجود ندارد .می توان به **tuple** به چشم **list** های فقط خواندنی (**read-only**) نیز نگریست . **tuple**را می توان لیست های فقط خواندنی نیز نام گذاشت.

مثال:

```
tuple = ( 'abcd', 786 , 2.23, 'john', 70.2 )
tinytuple = (123, 'john')
print (tuple)               # Prints complete list
```

```
print (tuple[0])          # Prints first element of the list
print (tuple[1:3])        # Prints elements starting from 2nd till 3rd
print (tuple[2:])         # Prints elements starting from 3rd element
print (tinytuple * 2)     # Prints list two times
print (tuple + tinytuple) # Prints concatenated lists
```

نتیجه ی زیر حاصل می گردد:

```
('abcd', 786, 2.23, 'john', 70.2)
abcd
(786, 2.23)
(2.23, 'john', 70.2)
(123, 'john', 123, 'john')
('abcd', 786, 2.23, 'john', 70.2, 123, 'john')
```

۵-۷-نوع داده ایDictionary

Dictionaryدر پایتون تا حدی شبیه به جداول hash table type هستند .این نوع داده ای علمکردی مشابه آرایه های شرکت پذیر - associative array - یا hash ها در Perl دارند و از جفت های کلید مقدار key-value pairs تشکیل می شوند .کلید می تواند از هر نوعی باشد، با این وجود اغلب از نوع اعداد و رشته ها هستند .اما مقادیر، می توانند از هر شی دلبخواه و اختیاری درپایتون باشند.

آیتم های Dictionary داخل {} محصور می شوند .جهت دسترسی و استخراج مقادیری ازdictionary می بایست از [] استفاده کرد .مثال:

```
dict = {}
dict['one'] = "This is one"
dict[2] = "This is two"
tinydict = {'name': 'john','code':6734, 'dept': 'sales'}
print (dict['one'])       # Prints value for 'one' key
print (dict[2])           # Prints value for 2 key
print (tinydict)          # Prints complete dictionary
print (tinydict.keys())   # Prints all the keys
print (tinydict.values()) # Prints all the values
```

نتیجه ی زیر را ارائه می دهد:

This is one
This is two
{'name': 'john', 'dept': 'sales', 'code': 6734}

dict_keys(['name', 'dept', 'code'])
dict_values(['john', 'sales', 6734])

نکته ۱: در Dictionary ، المان ها دارای ترتیب یا order مشخصی نیستند.

نکته ۲: برای فهمیدن اینکه هر کدام از متغییرهایی که تعریف کرده ایم از چه نوعی است می توانیم از نوع type() استفاده کنیم:

```
>>> a='2354.4'
>>> b=int(float(a))
>>> c=float(b)
>>> type(a)
<class 'str'>
>>> type(b)
<class 'int'>
>>> type(c)
<class 'float'>
```

۱– عملگرهای اصلی پایتون(Python operator)

عملگرها سازه هایی هستند که توسط آن ها می توان مقدار عملوندها را دستکاری کرد.

به این عبارت دقت کنید . 4 + 5 = 9 :در این عبارت، اعداد 4 و 5 عملوند (operand) خوانده شده وعلامت جمع، عملگر (operator) نامیده می شود.

انواع عملگرها

زبان پایتون از عملگرهای زیر پشتیبانی می کند.

۱- عملگرهای محاسباتی(arithmetic operators)

۲- عملگرهای مقایسه ای(comparison operators)

۳- عملگرهای انتساب(assignment operator)

۴- عملگرهای منطقی(logical operator)

۵- عملگرهای بیتی(bitwise operators)

در زیر به شرح تک تک این عملگرها خواهیم پرداخت.

۱– عملگرهای محاسباتی:

فرض کنید متغیر به نام **a** مقدار **10** و متغیر به نام **b** مقدار **20** را نگه می دارد:

عملگر	شرح	مثال
+ جمع	مقدار a را با مقدار b جمع می بندد.	a + b = 30
- تفریق	عملوند سمت راست را از عملوند سمت چپ کسرمیکند.	a – b = -10
*** ضرب**	مقادیر متغیرهای a و b را درهم ضرب می کند.	a * b = 200
/ تقسیم	عملوند سمت چپ را بر عملوند سمت راست تقسیم میکند.	b / a = 2
% باقی مانده تقسیم	عملوند سمت چپ را بر عملوند سمت راست تقسیم کرده و باقی مانده را بازمی گرداند.	b % a = 0
**** توان**	عملوندی را به توان عملوند دیگری می برد.	$a^{**}b = 10^{20}$
//تقسیم به کف	دو عدد را بر هم تقسیم کرده و نتیجه ی آن را به پایین گرد می کند.	9//2 = 4

۲–عملگرهای مقایسه ای پایتون :

این عملگرها مقادیر در دو طرف عملگر را با هم مقایسه کرده و رابطه ی بین آن ها را ارزیابی میکند .این عملگرها تحت عنوان **relational operators** نیز شناخته می شوند.

مثال : فرض کنید متغیر **a** مقدار **10** و متغیر **b** مقدار **20** را در خود ذخیره دارد:

عملگر	شرح	مثال
==	در صورت برابر بودن مقدار دو عملوند، شرط صحیح می باشد.	صحیح نمی باشد(a == b) .
!=	در صورت برابر نبودن مقدار عملوندها، شرط صحیح می شود.	صحیح است(a!=b)
<>	اگر مقادیر دو عملوند برابر نباشد، در آن صورت شرط صحیح می شود.	شرط (a <> b) صحیح می باشد .درست شبیه عملگر =!عمل می کند.
>	چنانچه مقدار عملوند سمت چپ بزرگتر از مقدار عملوند سمت راست باشد، شرط صحیح می باشد.	صحیح نمی باشد.(a > b)
<	اگر مقدار عملوند سمت چپ کمتر از مقدار عملوند سمت راست باشد، شرط صحیح می باشد.	صحیح می باشد(a < b) .
>=	اگر مقدار عملوند سمت چپ بزرگتر از یا برابر مقدار عملوند سمت راست باشد، در آن صورت شرط صحیح می شود.	صحیح نمی باشد(a >= b) .
<=	اگر مقدار عملوند سمت چپ کوچکتر از یا مساوی مقدار عملوند سمت چپ باشد، شرط صحیح می شود.	صحیح می باشد(a <= b) .

۳- عملگرهای انتساب:

عملگر	شرح	مثال
=	یک مقداری را به متغیر تخصیص می دهد.	جمع دو مقدار a و b را داخل متغیر c می ریزد.
+= Add AND	عملوند سمت راست را به عملوند سمت چپ اضافه کرده و نتیجه را در داخل عملوند سمت چپ می ریزد. cو a را جمع زده و نتیجه را داخل c می ریزد.	a =+ cدرواقع همان c = c + a می باشد.
-= Subtract AND	عملوند a را از c کسر می کند و نتیجه را به c تخصیص می دهد	a =- cدرحقیقت همان c = c – a می باشد.
/= Divide AND	متغیرهای c و a را بر هم تقسیم کرده و نتیجه را در عملوند سمت چپ می ریزد	a =/ cدر واقع همان c = c / a می باشد.
%= Modulus AND	باقی مانده تقسیم مقدار دو متغیر را محسابه کرده و آن را به عملگر سمت چپ تخصیص می دهد	a =% cدرواقع همان c = c % a می باشد.
**= Exponent AND	عملیات توان بر روی دو متغیر انجام داده و نتیجه را در متغیر سمت چپ عملگر می ریزد.	a =** cدرواقع همان c = c ** a می باشد.
//= تقسیم به کف	cو a را بر هم تقسیم کرده، نتیجه ی آن را به پایین گرد می کند، سپس آن را داخل c می ریزد.	a =// cدر واقع همان c = c // a می باشد.

۴- عملگرهای بیتی در پایتون

عملگرهای bitwise با بیت ها سروکار داشته و مقادیر بیت ها را تغییر می دهد .اگر a = 60 باشد و b = 13، هر یک در فرمت دودیی بدین صورت خواهند بود:

a = 0011 1100
b = 0000 1101
---
a&b = 0000 1100
a|b = 0011 1101

a^b = 0011 0001
~a = 1100 0011

زبان پایتون از عملگرهای بیتی زیر پشتیبانی می کند:

عملگر	شرح	مثال
& (Binary AND)	در صورتی که هر دوبیت 1 باشد، 1 را در نتیجه جای گذاری می کند، در غیر این صورت صفر را در نتیجه کپی می کند	0000 1100 معادل (a & b)
| (Binary OR)	اگر در یکی از دو عملوند، 1 وجود داشته باشد، 1 در نتیجه کپی می شود.	0011 1101معادل (a | b)
^ (Binary XOR)	اگر هر دو عملوند یکی باشند صفر و در غیر این صورت 1 را در نتیجه کپی می کند.	0011 0001معادل (a ^ b)
~ (Binary Ones Complement)	یک عملگر یگانی نقیض است که جای بیت ها را با هم عوض می کند .هرجا 1 است 0 و هرجا 1 است صفر می گذارد	چون که در فرمت دودویی عدد علامت دار نداریم، مکمل 2 عدد 0011 1100را نمایش دادیم که معادل 1100 0011است .
<< (Binary Left Shift)	مقدار عملوند سمت چپ به تعداد بیت های مشخص شده توسط عملوند سمت راست به چپ رانده می شوند.	<< a معادل 1111 0000
>> (Binary Right Shift)	مقدار عملوندهای سمت چپ به تعداد بیت های مشخص شده توسط عملوند سمت راست، به راست shiftمی شوند	معادل a >> . 0000 111

۵- عملگرهای منطقی پایتون

زبان پایتون از عملگرهای منطقی زیر پشتیبانی می کند .فرض بگیرید متغیر a دارای مقدار 10 ومتغیر b دارای مقدار 20 می باشد:

عملگر	شرح	مثال
and Logical AND	اگر هر دو علوند صحیح باشند، شرط برقرار و صحیح می باشد.	(a and b)صحیح می باشد.
or Logical OR	چنانچه یکی از دو عملوند صفر نباشد، شرط برقرار می شود(مقدار true	(a or b)صحیح می باشد.

	برگردانده می شود.)	
Not(a) می شود	به منظور معکوس کردن وضعیت منطقی عملوند بکار می رود.	**not Logical NOT**

❖ اولویت عملگرها در پایتون

جدول زیر تمامی عملگرها را به ترتیب اولویت فهرست کرده:

شرح	عملگر	ترتیب
توان	**	۱
منفی , مثبت ,نقیض	~ + -	۲
کف به تقسیم , باقی مانده, تقسیم، ضرب،	* / % //	۳
جمع و تفریق	+ -	٤
شیفت به چپ , شیفت به راست	>> <<	٥
Bitwise 'AND'	&	٦
Bitwise exclusive `OR' and regular `OR'	^ |	۷
عملگرهای مقایسه ای	<= < > >=	۸
عملگرهای برابری	<> == !=	۹
عملگرهای تخصیص یا انتساب	= %= /= //= -= += *= **=	۱۰