


نکات اضافی درباره لیست ها :


گفتیم که `append` و `extend` چه تفاوتی دارند. `Append` میومد آن آیتم های اضافه شده را به عنوان یک لیست اضافه میکرد و `extend` میومد آیتم هارا مستقیم اضافه میکرد :

```
test2.py > ...
Click here to ask Blackbox to h
1 a = [1,2,3,4]
2 b = [3,4,5,6]
3
4 a.append(b)
5
6 print(a)
```



```
[1, 2, 3, 4, [3, 4, 5, 6]]
PS C:\Users\hamid\Desktop\N
```


```
test2.py > ...
Click here to ask B
1 a = [1,2,3,4]
2 b = [3,4,5,6]
3
4 a.extend(b)
5
6 print(a)
```



```
[1, 2, 3, 4, 3, 4, 5, 6]
```

خب ؟ تا اینجا شو گرفتی ؟ حالا فرض کن ما `append` زدیم . میدونید که خود اون لیست میشه یدونه ایندکس که خودش باز ایندکس داره ببین اینو :

```
['a', 'b', 'c', 'd', ['e', 'a', 'b']]
0 1 2 3 4
```



خود اون لیست شده ایندکس شماره 4 یعنی اگر پیام پرینتش بگیرم برام برمیگردونتش :

```
test2.py > ...
Click here to ask Blackbox to help you
1 a = ["a" , "b" , "c" , "d"]
2 b = ["e" , "a" , "b"]
3
4 a.append(b)
5 print(a)
6 print(a[4])
7 print(type(a[4]))
```

```
/test2.py
['a', 'b', 'c', 'd', ['e', 'a', 'b']]
['e', 'a', 'b']
<class 'list'>
```

نیگا : وقتی ایندکس 4 رو خروجی گرفتم خود اون لیست توی اون لیست اولیه رو برام چاپ کرد . نوعش خروجی گرفتم نوشت لیست پس شد این یه لیست تو در تو .

حالا چجور به آیتم های تو اون لیست اولیه دسترسی داشته باشیم ؟ خب کاری نداره :

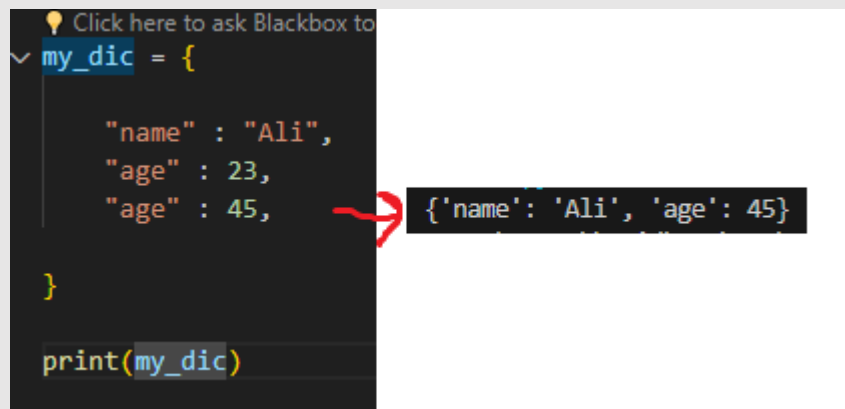
```
test2.py > ...
Click here to ask Blackbox to help you
1 a = ["a" , "b" , "c" , "d"]
2 b = ["e" , "a" , "b"]
3
4 a.append(b)
5
6 print(a)
7 print(a[4][1])
```

```
/test2.py
['a', 'b', 'c', 'd', ['e', 'a', 'b']]
a
```

گفتم `a[4][1]` : برو از `a` برو سراغ ایندکس 4 امی ( اون لیسته توی لیست اولیه ) بعد برو سراغ ایندکس شماره 1 اش یعنی `a` .

کار با دیکشنری ها :

گفتیم که دیکشنری ها به صورت `key : value` هستند و `key` ها به قول معروفی `unique` باید باشند و اگر تکرار بشن موقع خورجی گرفتن آخری شو نشون میده و اینکه باید `key` ها استرینگ باشن :



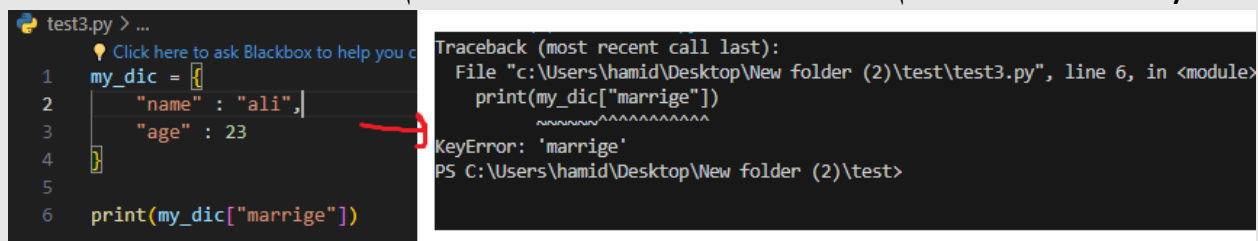
```
Click here to ask Blackbox to help you c
my_dic = {
    "name" : "Ali",
    "age" : 23,
    "age" : 45,
}

print(my_dic)
```

→ {'name': 'Ali', 'age': 45}

بعدش گفتیم :

اگر `key` ای را برگردانیم که وجود نداشته باشه ارور میگیریم :



```
test3.py > ...
Click here to ask Blackbox to help you c
1 my_dic = {}
2   "name" : "ali",
3   "age" : 23
4 }
5
6 print(my_dic["marriage"])
```

```
Traceback (most recent call last):
  File "c:\Users\hamid\Desktop\New folder (2)\test\test3.py", line 6, in <module>
    print(my_dic["marriage"])
    ~~~~~^~~~~~
KeyError: 'marriage'
PS C:\Users\hamid\Desktop\New folder (2)\test>
```

حال اگر نخواهیم که ارور بگیریم و اگر وجود نداشت `none` برگردونه :

```
test3.py > ...  
Click here to ask Blackbox to help you code faster  
1 my_dic = {  
2     "name" : "ali",  
3     "age" : 23  
4 }  
5  
6 print(my_dic.get("marriage"))
```

None

میشود مسائل قبلی را با اینها هم ترکیب کرد مثلا :

```
test3.py > ...  
Click here to ask Blackbox to help you code faster  
1 my_dic = {  
2     "name" : "ali",  
3     "age" : 23  
4 }  
5  
6 print(my_dic["name"].upper())
```

ALI

```
test3.py > ...  
Click here to ask Blackbox to help you code faster  
1 my_dic = {  
2     "name" : "ali",  
3     "age" : 23  
4 }  
5  
6 print(type(my_dic["age"]))
```

<class 'int'>

```
test3.py > ...  
Click here to ask Blackbox to help you code faster  
1 my_dic = {  
2     "name" : "ali",  
3     "age" : 23  
4 }  
5  
6 print(my_dic["name"].isdigit()) # False
```

False

دیکشنری ها هم میتوانند تو در تو باشند :

```
test3.py > ...  
Click here to ask Blackbox to help you code faster  
1 my_dic = {  
2  
3     "dic1" : {"name" : "ali" , "age" : 30},  
4     # key          value  
5     "dic2" : {"marriage" : False , "death" : True},  
6  
7 }  
8  
9 print(my_dic)
```

```
{'dic1': {'name': 'ali', 'age': 30}, 'dic2': {'marriage': False, 'death': True}}
```

key

key

حال برای دسترسی به المان های آن باید مثل لیست های تو در تو عمل کنیم :

```
test3.py > ...  
Click here to ask Blackbox to help you code faster  
1 my_dic = {  
2  
3     "dic1" : {"name" : "ali" , "age" : 30},  
4     # key          value  
5     "dic2" : {"marriage" : False , "death" : True},  
6  
7 }  
8  
9 print(my_dic['dic1']["name"])  
10 print(my_dic["dic1"]["age"])  
11 print(my_dic['dic2']["marriage"])  
12 print(my_dic["dic2"]["death"])
```

ali  
30  
False  
True

تمام متد های دیکشنری هارو میشه برای این ها هم استفاده کرد :

```
test3.py > ...
Click here to ask Blackbox to help you code faster
1 my_dic = {
2
3     "dic1" : {"name" : "ali" , "age" : 30},
4     # key      value
5     "dic2" : {"marriage" : False , "death" : True},
6 }
7
8
9 print(my_dic['dic1'].get("name"))
10 my_dic['dic2'].update({"age" : 30})
11 print(my_dic)
```

The output of the code is shown on the right:

```
{'dic1': {'name': 'ali', 'age': 30}, 'dic2': {'marriage': False, 'death': True, 'age': 30}}
```

Red and blue arrows indicate the flow of data: a red arrow points from the `name` value in the first dictionary to the `ali` output, and a blue arrow points from the `age` key in the second dictionary to the `age` key in the final output dictionary.

در خط اول اومدیم داخل "dic1" که خود key دیکشنری اصلی هست از متد `get` استفاده کردیم که key رو دادیم و value رو برگردونده برامون

در خط دوم هم داخل "dic2" رو آپدیت کردیم و {"age" : 30} را وارد آن کردیم

شرط ها :

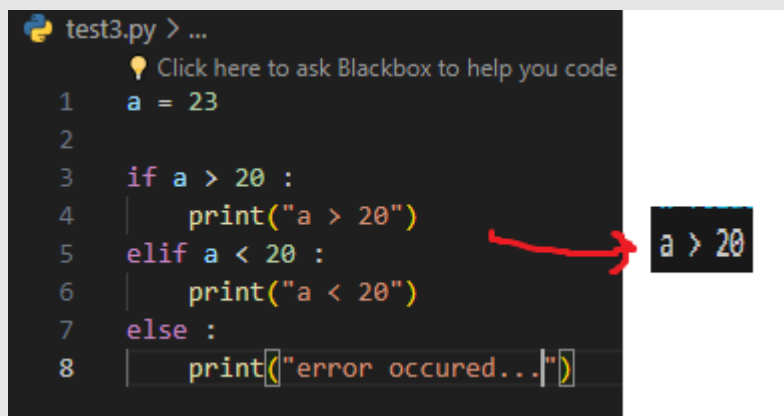
If condition :  
statement

elif condition :  
statement

elif condition :  
statement

else:  
statement

حواستون به indent گذاری ها باشه statement ها باید 4 تا اسپیس یا یک تب فاصله داشته باشند نیازی هم نیست که 4 تا اسپیس بزنید وقتی بعد از 2 نقطه بعد از condition اینتر میزنید و صدای کیبرد رو میشنوید :) خودش 4 تا فاصله میندازه اگر ننداخت با زدن دکمه تب خودش 4 تا فاصله میندازه .



```
test3.py > ...  
Click here to ask Blackbox to help you code  
1 a = 23  
2  
3 if a > 20 :  
4     print("a > 20")  
5 elif a < 20 :  
6     print("a < 20")  
7 else :  
8     print("error occured...")
```

یک نکته خیلی مهم توی شرط ها این است که وقتی یک شرط درست در بیاد دیگه شرط های بعدی اجرا نمیشوند :

```
test3.py > ...  
Click here to ask Blackbox to help you code  
1 a = 2  
2  
3 if a > 20 :  
4     print("a > 20")  
5 elif type(a) == str :  
6     print("a is string")  
7 elif a == 2:  
8     print("a is 2")  
9 else :  
10    print("error eccured")
```

a is string

در اینجا شرط دومی که گفته اگر نوعش استرینگ بود چون درست در اومده با اینکه شرط بعدی که میگه اگر 2 بود درسته ولی باز هم اجرا نشده !

یک مثال :

```
test3.py > ...  
Click here to ask Blackbox to help you code faster  
1 a = ["a", "b", "c", "d"]  
2  
3 if type(a[0]) == int :  
4     print(f"the first item of the {a} list was an integer")  
5 elif type(a[0]) == str :  
6     print(f"the first item of the {a} list was a string")  
7 else :  
8     print("the type was something else")
```

the first item of the ['a', 'b', 'c', 'd'] list was a string

این کد رو با کمک تابع همیشه کوچیکش کرد که ما چون تابع رو نگفتیم این بحث رو وسط نمیکشم و فقط کد شو براتون میزارم که بعدا با تابع کار کردیم یادمون نره چون این یگ مثال خیلی خوبیه برای تابع :



```
test3.py > ...  
  Click here to ask Blackbox to help you code faster  
1  a = ["a" , "b" , "c" , "d"]  
2  
3  def check(list , index):  
4      return type(a[index])  
5  
6  
7  if check(a , 0) == int :  
8      print(f"the first item of the {a} list was an integer ")  
9  elif check(a , 0) == str :  
10     print(f"the first item of the {a} list was a string")  
11 else :  
12     print("the type was something else")
```

the first item of the ['a', 'b', 'c', 'd'] list was a string