Tuples

چندتایی‌ها

چندتایی‌ها مانند لیست‌ها هستند، اما یک تفاوت اساسی دارند، و آن این است که نمی‌توانیم عنصری را درون آن تغییر دهیم. شاید بخواهید از چندتایی‌ها جایی استفاده کنید که عناصرش نباد تغییر داده شوند. مثلا نام‌های هفته، مناسبت‌ها، لیست کلاس و ... در این بخش به طور خلاصه محتوای زیر را خواهیم دید:

- ایجاد یک چندتای
- چند متد پایه‌ای
- خاصیت تغییرناپذیری
- !کی از چندتایی استفاده کنیم

شما حتما به ذهن‌تان رسیده که چطور از چندتایی‌ها استفاده کنید، چون آن‌ها شبیه لیست‌ها هستند. یعنی دقیقا با چندتایی همانطور برخورد می‌کنید که با لیست برخورد می‌کنید، بجز تفاوت اساسیشون که همان تغییر ناپذیر بودن هست!

ساخت یک چندتایی

برای ایجاد یک چندتایی از دو پرانتز () استفاده می‌کنیم و عناصر درون آن را با ویرگول جدا می‌کنیم.

In [1]:
```
a = [1,2,3,4]
```

In [2]:
```
a[2] = 5
a
```

Out[2]:
```
[1, 2, 5, 4]
```

In [3]:
```
# Create a tuple
t = (1,2,3)
```

In [4]:
```
# Check len just like a list
len(t)
```

Out[4]:
```
3
```

In [5]:
```
# Can also mix object types
t = ('one',2)

# Show
t
```

Out[5]:
```
('one', 2)
```

In [6]:
```
# Use indexing just like we did in lists
t[0]
```

Out[6]:
```
'one'
```

In [7]:
```
# Slicing just like a list
t[-1]
```

Out[7]:
```
2
```

In [8]:
```
t[1] = 3
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-8-91448c5f771b> in <module>()
----> 1 t[1] = 3

TypeError: 'tuple' object does not support item assignment
```

In [9]:
```
a = [2]
a
```

Out[9]:
```
[2]
```

In [14]:
```
b = (2,)
type(b)
```

Out[14]:
```
tuple
```

Basic Tuple Methods

Tuples have built-in methods, but not as many as lists do. Let's look at two of them:

In [ ]:
```
# Use .index to enter a value and return the index
t.index('one')
```

In [15]:
```
# Use .count to count the number of times a value appears
t.count('one')
```

Out[15]:
```
1
```

Immutability

It can't be stressed enough that tuples are immutable. To drive that point home:

In [ ]:
```
t[0]= 'change'
```

Because of this immutability, tuples can't grow. Once a tuple is made we can not add to it.

In [ ]:
```
t.append('nope')
```

When to use Tuples

You may be wondering, "Why bother using tuples when they have fewer available methods?" To be honest, tuples are not used as often as lists in programming, but are used when immutability is necessary. If in your program you are passing around an object and need to make sure it does not get changed, then a tuple becomes your solution. It provides a convenient source of data integrity.

You should now be able to create and use tuples in your programming as well as have an understanding of their immutability.

Up next Sets and Booleans!!