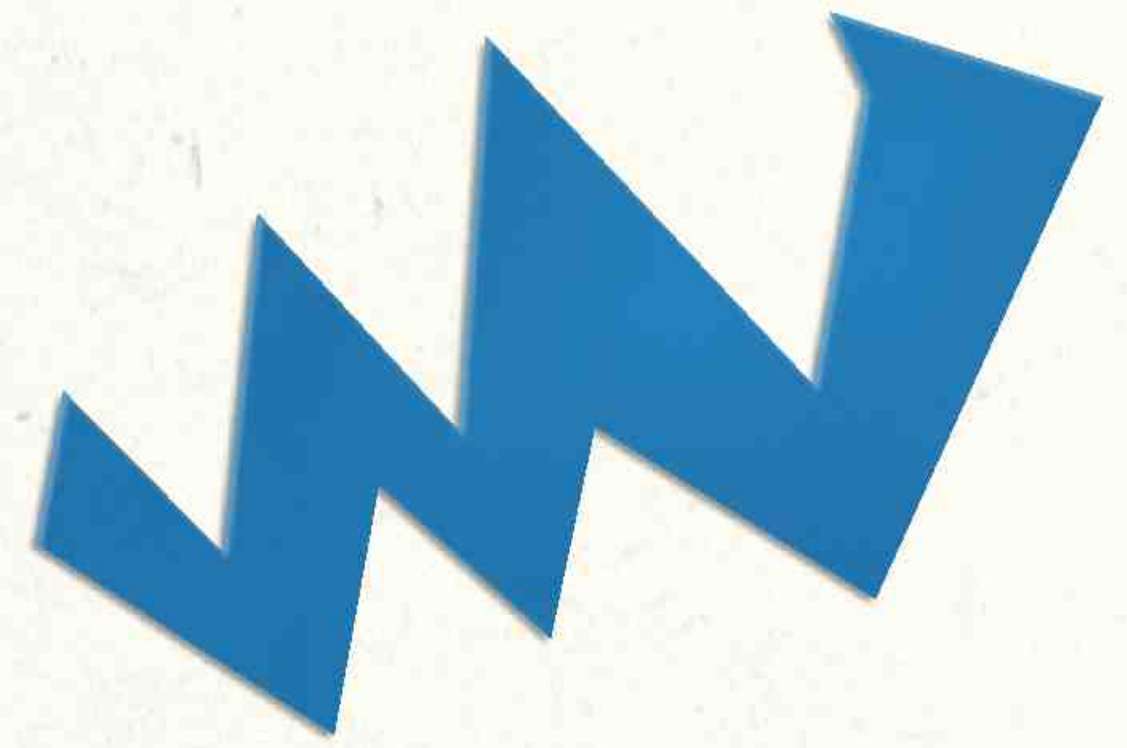
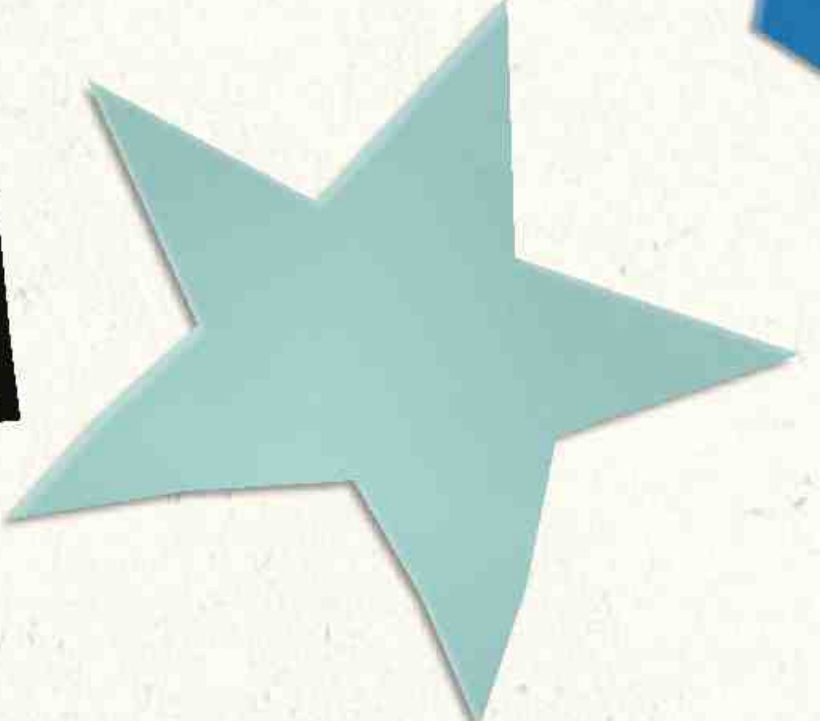
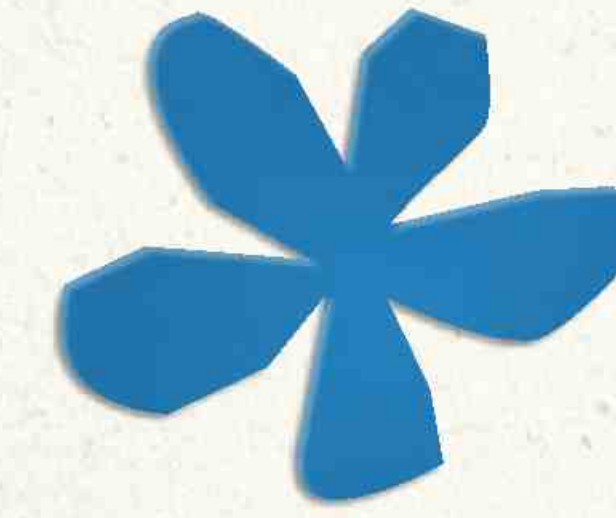
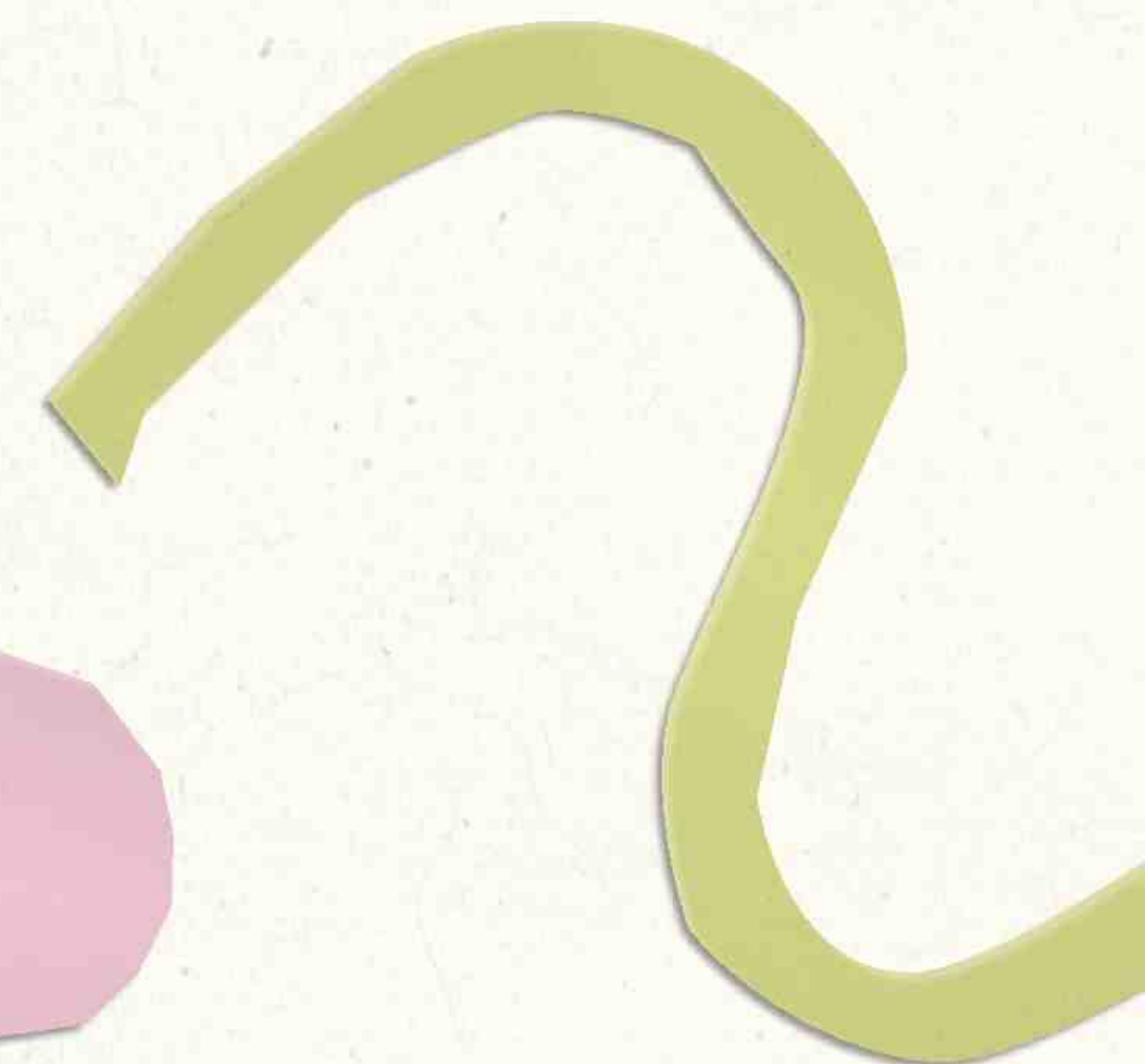
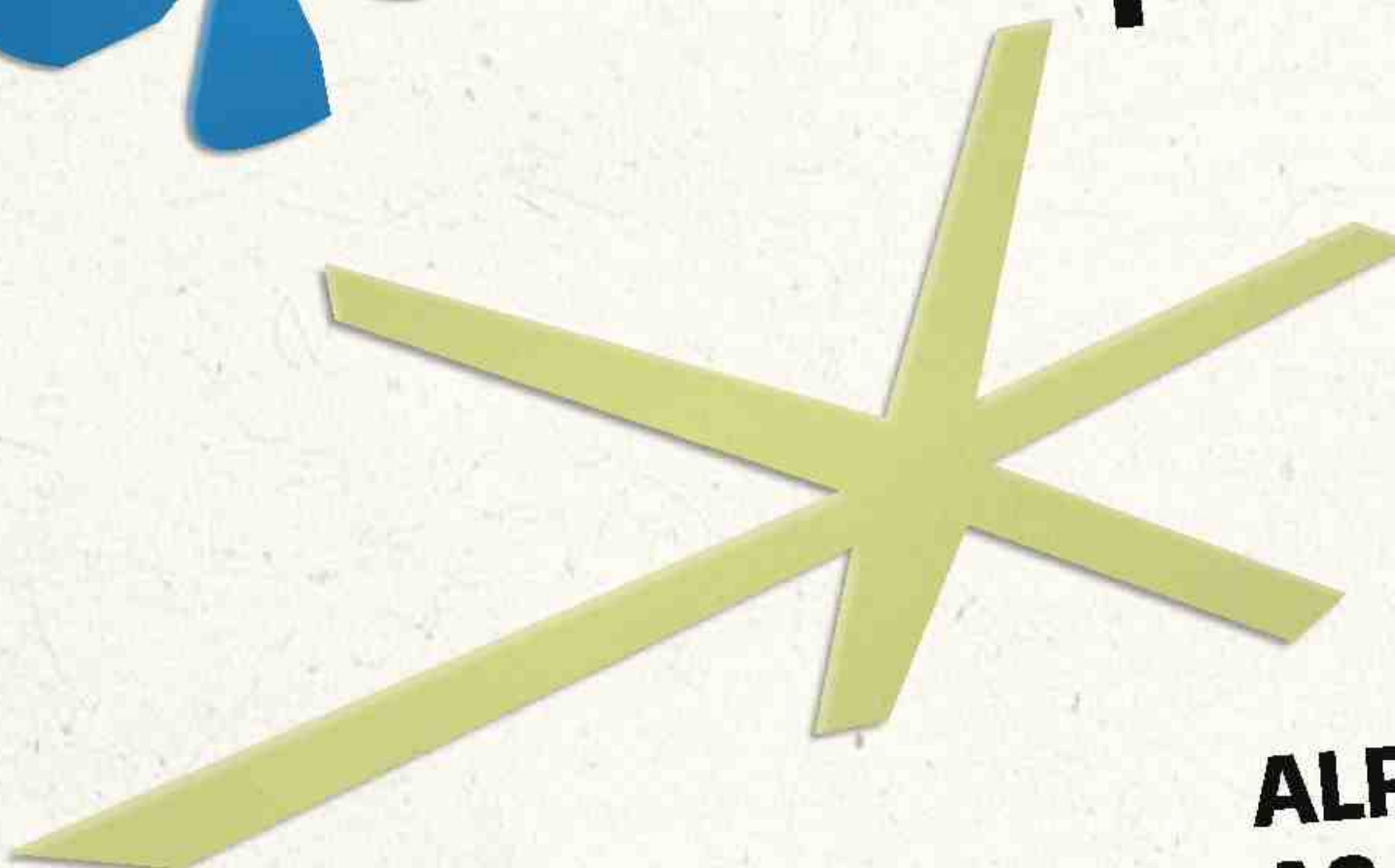




python

class



ALPAN ACADEMY
ASAL RASOULI





تاپل (Tuple) چیست؟

تاپل (Tuple) یکی از انواع داده‌های مرکب در پایتون است که برای نگهداری مجموعه‌ای از داده‌ها استفاده می‌شود. از نظر ظاهری، تاپل شباهت زیادی به لیست دارد؛ با این تفاوت که بعد از ایجاد، امکان تغییر عناصر آن وجود ندارد.

به عبارت دیگر، اگر یک تاپل ایجاد کنید، دیگر نمی‌توانید:

- مقدار یکی از عناصر را تغییر دهید.
- عنصر جدیدی به آن اضافه کنید.
- عنصری را حذف کنید.
-

به همین دلیل به تاپل یک نوع داده **غیر قابل تغییر (Immutable)** گفته می‌شود.

چرا از Tuple استفاده می‌کنیم؟

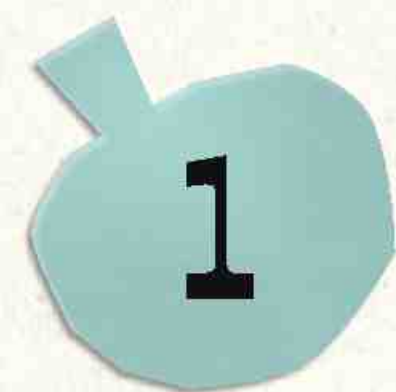
ممکن است این سؤال برایتان پیش بیاید:

وقتی List هم وجود دارد، چرا باید از Tuple استفاده کنیم؟

پاسخ این است که همیشه قرار نیست اطلاعات تغییر کنند. فرض کنید برنامه‌ای برای ذخیره روزهای هفته نوشته‌اید. روزهای هفته همیشه ثابت هستند و هیچ‌وقت قرار نیست روز جدیدی اضافه یا حذف شود. در چنین شرایطی استفاده از Tuple انتخاب مناسب‌تری است. نمونه‌های دیگر:

- مختصات جغرافیایی یک مکان
- رنگ‌های اصلی
- ماه‌های سال
- اطلاعات ثابت یک محصول
- اطلاعات ثابت یک دانشجو که نباید تغییر کند

ویژگی‌های tuple



مرتب (Ordered)



غیر قابل تغییر
(Immutable)



امکان وجود عناصر
تکراری



امکان ذخیره انواع
مختلف داده



سرعت بیشتر
نسبت به List

مقایسه اولیه List و Tuple

ویژگی	List	Tuple
قابل تغییر	✓	✗
مرتب	✓	✓
عناصر تکراری	✓	✓
سرعت	کمتر	بیشتر
مصرف حافظه	بیشتر	کمتر



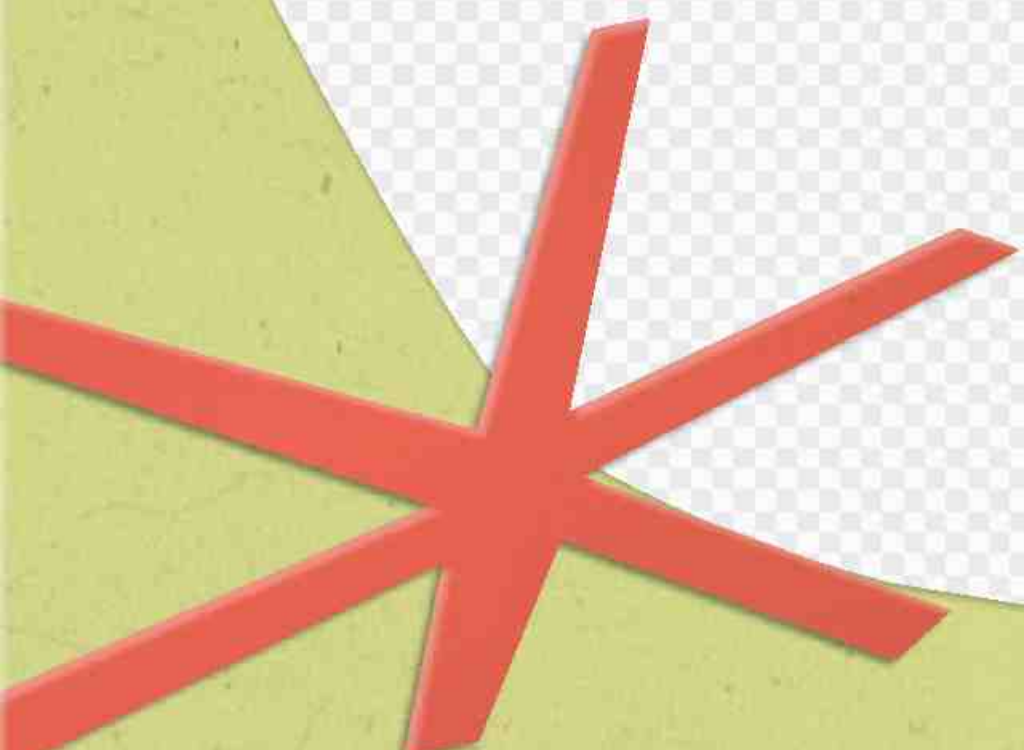
ساخت تاپل (Creating Tuple)

```
fruits = ("Apple", "Banana", "Orange")
```

```
print(fruits)
```

```
student = ("Sara", 18, "Yazd")
```

```
print(student)
```



```
tuple_name = (item1, item2, item3)
```

برای ایجاد یک تاپل از پرانتز () استفاده می‌کنیم.

تاپل تک‌عضوی (Single Element Tuple)

```
number = (10,)
```

```
print(type(number))
```

تاپل خالی (Empty Tuple)

```
my_tuple = ()
```

```
print(my_tuple)
```



انديس (Index)

Index: 0 1 2

fruits = ("سیب", "موز", "پرتقال")

هر عنصر داخل تاپل یک شماره دارد.
به این شماره **اندیس (Index)** گفته می شود.
اندیس ها از **صفر** شروع می شوند.

دسترسی به عناصر
fruits = ("سیب", "موز", "پرتقال")

print(fruits[0])

print(fruits[2])

اندیس منفی

پایتون از اندیس منفی نیز پشتیبانی می‌کند.
در این حالت شمارش از انتهای لیست آغاز
می‌شود.

-1 -2 -3

دسترسی به عناصر

```
fruits = ("سیب", "موز", "پرتقال")
```

```
print(fruits[-1])
```

```
print(fruits[-2])
```



برش تاپل (Slicing)

```
tuple[start:end]
```

```
numbers = (1,2,3,4,5)
```

```
print(numbers[1:4])
```

```
print(numbers[:3])
```

```
print(numbers[2:])
```

```
print(numbers[:])
```

گاهی فقط بخشی از تاپل را نیاز داریم.
برای این کار از **Slicing** استفاده می‌کنیم.
ساختار کلی:

عنصر start در خروجی قرار می‌گیرد، اما
عنصر end در خروجی قرار نمی‌گیرد.

list[start : end : step]

استفاده از Step

numbers = (0,1,2,3,4,5,6,7)

print(numbers[:2])

print(numbers[::-1])



عملیات روی تاپل‌ها (Tuple Operations)

الحاق (Concatenation)

گاهی لازم است دو یا چند تاپل را به هم متصل کنیم. برای این کار از عملگر + استفاده می‌کنیم.

عملگر + یک تاپل جدید ایجاد می‌کند و عناصر هر دو تاپل را در کنار هم قرار می‌دهد.

دقت کنید که هیچ‌یک از تاپل‌های اصلی تغییر نمی‌کنند.

```
tuple3 = tuple1 + tuple2
```

```
tuple1 = ("Python", "Java")
```

```
tuple2 = ("C++", "C#")
```

```
languages = tuple1 + tuple2
```

```
print(languages)
```

تکرار (Repetition)

```
new_tuple = old_tuple * number
```

```
numbers = (1, 2)
```

```
print(numbers * 3)
```

برای تکرار عناصر یک تاپل از عملگر *
استفاده می‌کنیم.

جستجو در تاپل (Membership Operators)

عملگر in

اگر مقدار داخل تاپل وجود داشته باشد،
نتیجه True خواهد بود.

```
numbers = (10, 20, 30, 40)
```

```
print(20 in numbers)
```

```
print(50 in numbers)
```

جستجو در تاپل (Membership Operators)

عملگر not in

اگر مقدار داخل تاپل وجود نداشته باشد،
نتیجه True خواهد بود.

```
numbers = (10, 20, 30)
```

```
print(50 not in numbers)
```

```
print(20 not in numbers)
```



توابع کاربردی

توابع کاربردی



len

تعداد عناصر



sum

مجموع



max

بزرگ‌ترین مقدار



min

کوچک‌ترین مقدار



sorted

مرتب کردن

```
numbers = (10, 20, 30, 40)
```

```
print(len(numbers))
```

```
print(sum(numbers))
```

```
print(max(numbers))
```

```
print(min(numbers))
```

```
print(sorted(numbers))
```



متدهای پرکاربرد تایل

count

شمارش تعداد تکرار

```
numbers = (1, 2, 3, 2, 2, 5)
```

```
print(numbers.count(2))
```

index

پیدا کردن محل اولین عنصر

```
numbers = (10, 20, 30, 40)
```

```
print(numbers.index(30))
```



بسته‌بندی (Packing) باز کردن (Unpacking)

Unpacking

در این حالت عناصر تاپل را داخل چند متغیر جداگانه قرار می‌دهیم.

```
student = ("Ali", 20, "Yazd")
```

```
name, age, city = student
```

```
print(name)
```

```
print(age)
```

```
print(city)
```

Packing

وقتی چند مقدار را داخل یک تاپل قرار می‌دهیم، به این کار Packing گفته می‌شود.

```
student = ("Ali", 20, "Yazd")
```



تبدیل بین List و Tuple

```
numbers = (10,20,30)
```

```
my_list = list(numbers)
```

```
print(my_list)
```

گاهی لازم است یکی از عناصر یک تاپل را
تغییر دهیم.

از آنجایی که تاپل قابل تغییر نیست، ابتدا آن
را به لیست تبدیل می‌کنیم.



تاپل‌های تو در تو (Nested Tuples)

```
matrix = (  
    (1,2,3),  
    (4,5,6),  
    (7,8,9)  
)
```

یک تاپل می‌تواند شامل تاپل‌های دیگر باشد.

```
print(matrix[1])
```

دسترسی به ردیف دوم

```
print(matrix[1][2])
```

دسترسی به عدد ۶

```
matrix[2][0] = 10
```

تغییر مقدار

```
print(matrix)
```

TypeError



نمونه برنامه کامل

در این مثال، اطلاعات چند درس و نمرات یک دانش‌آموز داخل Tuple ذخیره شده است. سپس با استفاده از توابع مختلف، اطلاعات پردازش و نمایش داده می‌شوند.

```
subjects = ("ریاضی", "علوم", "برنامه نویسی", "زبان انگلیسی")  
scores = (18, 17, 20, 19)
```

```
count_scores = len(scores)
```

```
average_score = sum(scores) / len(scores)
```

```
highest_score = max(scores)
```

```
lowest_score = min(scores)
```

```
print("آیا درس ریاضی وجود دارد؟", "ریاضی" in subjects)
```

در این مثال، اطلاعات چند درس و نمرات یک دانش‌آموز داخل Tuple ذخیره شده است. سپس با استفاده از توابع مختلف، اطلاعات پردازش و نمایش داده می‌شوند.

```
print("تعداد درس‌ها:", len(subjects))  
print("بیشترین نمره:", highest_score)  
print("کمترین نمره:", lowest_score)  
print("میانگین نمرات", average_score)
```

```
lesson1, lesson2, lesson3, lesson4 = subjects
```

```
print("درس اول:", lesson1)  
print("درس دوم:", lesson2)
```

ویژگی	List	Tuple
نحوه تعریف	[]	()
قابل تغییر (Mutable)	✓	✗
امکان اضافه کردن عنصر	✓	✗
امکان حذف عنصر	✓	✗
امکان تغییر عنصر	✓	✗
ترتیب عناصر حفظ می‌شود	✓	✓
امکان وجود عناصر تکراری	✓	✓
پشتیبانی از Index	✓	✓
پشتیبانی از Slicing	✓	✓
سرعت	کمتر	بیشتر
مصرف حافظه	بیشتر	کمتر

چه زمانی از Tuple استفاده کنیم؟

اگر اطلاعات قرار نیست تغییر کنند، Tuple انتخاب مناسب‌تری است.
مثال:

- روزهای هفته
- ماه‌های سال
- مختصات GPS
- رنگ‌های اصلی

چه زمانی از List استفاده کنیم؟

اگر اطلاعات برنامه ممکن است تغییر کنند، از List استفاده می‌کنیم.
مثال:

- لیست خرید
- سبد خرید فروشگاه
- لیست دانش‌آموزان کلاس
- پیام‌های یک شبکه اجتماعی

در این بخش یاد گرفتیم:

- ✓ تاپل مجموعه‌ای مرتب از داده‌هاست.
- ✓ بعد از ایجاد، امکان تغییر عناصر آن وجود ندارد.
- ✓ از اندیس‌گذاری و برش (Slicing) پشتیبانی می‌کند.
- ✓ می‌توان دو Tuple را با هم ترکیب یا تکرار کرد.
- ✓ عملگرهای in و not in برای جستجوی عناصر استفاده می‌شوند.
- ✓ از توابعی مانند min، sum()، len() و max() می‌توان روی آن استفاده کرد.
- ✓ تنها دو متد count() و index() را دارد.
- ✓ می‌توان با Packing و Unpacking داده‌ها را مدیریت کرد.
- ✓ در صورت نیاز، می‌توان آن را به List تبدیل کرد و دوباره به Tuple برگرداند.



تمرین کلاسی

66

یک Tuple شامل نام ۵ شهر ایران ایجاد کنید و آن
را چاپ کنید.

66

66

تاپل زیر را ایجاد کنید:

`cities = ("یزد", "تهران", "شیراز", "اصفهان", "تبریز")`

موارد زیر را چاپ کنید:

- اولین شهر
- سومین شهر
- آخرین شهر

67

66

تاپل زیر را ایجاد کنید:

```
numbers = (10, 20, 30, 40, 50, 60)
```

موارد زیر را چاپ کنید:

- سه عنصر اول
- سه عنصر آخر
- عناصر یک در میان
- تاپل برعکس

66

66

برنامه‌ای بنویسید که بررسی کند آیا "Python" داخل Tuple
زیر وجود دارد یا خیر.

```
languages = ("C++", "Java", "Python", "C#")
```

67

66

تاپل زیر را ایجاد کنید:

```
student = ("Ali",20,"Yazd")
```

با استفاده از Unpacking، اطلاعات را در سه متغیر جداگانه ذخیره کنید و هر کدام را چاپ کنید.

66

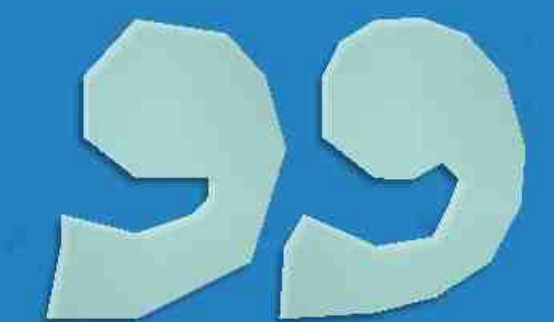


تاپل زیر را ایجاد کنید:

```
numbers = (10, 20, 30, 40)
```

سپس:

- آن را به List تبدیل کنید.
- عدد ۲۰ را به ۲۵ تغییر دهید.
- دوباره به Tuple تبدیل کنید.
- نتیجه را چاپ کنید.





تمرین‌های خانگی

66

یک Tuple شامل نام ۷ روز هفته ایجاد کنید و
موارد زیر را نمایش دهید:

- اولین روز
- آخرین روز
- تعداد روزها

66

66

تاپل زیر را ایجاد کنید:

`grades = (12,18,20,15,18,17)`

موارد زیر را چاپ کنید:

- تعداد تکرار عدد ۱۸
- اندیس اولین عدد ۲۰
- میانگین نمرات

99

66

دو Tuple زیر را ایجاد کنید:

```
tuple1 = ("HTML", "CSS")
```

```
tuple2 = ("Python", "JavaScript")
```

آن‌ها را با هم ترکیب کنید و نتیجه را نمایش دهید.

66

66

تاپل زیر را ایجاد کنید:

```
numbers = ( 1 , 2 , 3 )
```

آن را سه بار تکرار کنید و نتیجه را چاپ کنید.

99

66

برنامه‌ای بنویسید که بررسی کند آیا عدد ۵۰ در Tuple زیر وجود دارد یا خیر.

```
numbers = (10, 20, 30, 40, 50, 60)
```

99

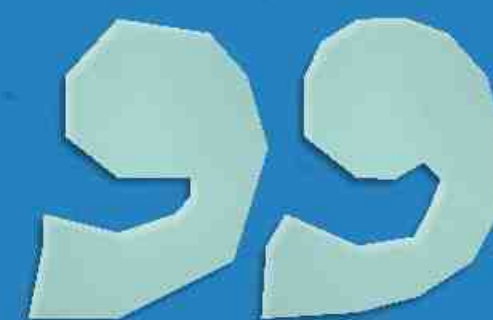


تاپل زیر را ایجاد کنید:

```
products = ("Mouse", "Keyboard", "Monitor")
```

سپس:

1. آن را به List تبدیل کند.
2. محصول "Printer" را به انتهای لیست اضافه کند.
3. محصول "Keyboard" را حذف کند.
4. دوباره آن را به Tuple تبدیل کند.
5. نهایی را چاپ کند Tuple



THANKS!

ALPAN ACADEMY
ASAL RASOULI