Lists

لیست ها

- مجموعه ای از مقادیر مرتب که در آن هر عضو به وسیله اندیس مشخصی نشان داده می شود.
- به مقداری که در لیست ذخیره می شود، عضو یا عنصر گفته می شود

```
>>> first_list = [1,2,3,4]
```

- لیست در پایتون می تواند شامل عناصری با نوع داده ای مختلف باشد.

```
>>> mixed = [2, 4.765, True, “good”]
```

عملیات پایه در لیست ها

- ایجاد یک لیست جدید

```
>>> empty = []
>>> letters = [‘a’, ‘b’, ‘c’, ‘d’]
>>> numbers = [2, 3, 5]
```

- اضافه کردن یک عنصر جدید به انتهای لیست

```
>>> numbers.append(7)
Output: [2, 3, 5, 7]
```

- تعیین تعداد عناصر لیست

```
>>> len (letters)
Output: 4
```

دسترسی به عناصر لیست

- دسترسی به یک عنصر در یک ایندکس خاص

```
>>> numbers[0]
Output: 2
>>> numbers[-1]
Output: 7
```

- برش قسمتی از یک لیست

```
>>> letters [ :3]
Output: [‘a’, ‘b’, ‘c’]
>>> numbers [1 : -1]
Output: [3, 5]
```

لیست های تودرتو

- هر نوع عنصری می تواند در یک لیست قرار گیرد حتی یک لیست دیگر !

```
>>> x = [letters, numbers]
>>> x
Output: [ [‘a’, ‘b’, ‘c’, ‘d’] , [2, 3, 5, 7] ]
>>> x[0]
Output: [‘a’, ‘b’, ‘c’, ‘d’]
>>> x[0][1]
Output: ‘b’
 >>> x[1][2: ]
Output: [5, 7]
```

List Methods

- اضافه کردن یک object قبل از یک index مشخص

```
>>> my_list.insert(index, object)
```

- مثال:

```
>>> a = [1,2,3]
>>> a.insert(1, 8)
Output: [1, 8, 2, 3]
```

- حذف عنصر value از لیست (اولین رخداد)

```
>>> my_list.remove(value)
```

- حذف تمام آیتم های موجود در لیست

```
>>> my_list.clear()
```

List Methods - ادامه

- برگرداندن تعداد تکرار یک value در لیست

```
>>> my_list.count(value)
```

- تعیین اولین index یک value

```
>>> my_list.index(value)
```

- حذف یک عنصر در index

```
>>> my_list.pop(index)
```

- مرتب سازی یک لیست

```
>>> my_list.sort()
```

- برعکس کردن ترتیب عناصر در لیست

```
>>> my_list.reverse()
```

تابع range(x,y)

- برای تولید لیستی از اعداد صحیح متوالی در یک بازه خاص، از تابع range استفاده می شود.
- عموما این تابع شامل دو آرگومان می باشد.
- آرگومان x ابتدای بازه و آرگومان y انتهای بازه را مشخص می کند.

```
>>> range(1, 5)
Output: [1, 2, 3, 4]
```

- این تابع می تواند شامل ۱ و یا ۳ آرگومان هم باشد.
- مثال

```
>>> range(5)
Output: [0, 1, 2, 3, 4]
>>> range(1, 10, 2)
Output: [1, 3, 5, 7, 9]
```

عضویت در لیست – عملگر in

- عملگر in یک عملگر بولی می باشد که عضویت یک عنصر در یک دنباله را تعیین می کند.
- علاوه بر لیست در سایر دنباله ها هم استفاده می شود.
- مثال

```
>>> name = [‘Ali’, ‘Reza’, ‘Zahra’]
>>> ‘Ali’ in name
Output: True (or 1)
>>> ‘Akbar’ in name
Output: False (or 0)
```

- از عملگر not in هم می توان استفاده کرد.

عملگرهای لیست

- عملگر + دو لیست را به هم پیوند می زند.

```
>>> a = [1, 2, 3]
>>> b = [4, 5, 6]
>>> c = a + b
>>> print (c)
Output: [1, 2, 3, 4, 5, 6]
```

- عملگر * یک لیست را به تعداد مشخصی تکرار می کند.

```
>>> [1, 2] * 3
Output: [1, 2, 1, 2, 1, 2]
```

Mutable

- بر خلاف رشته ها، لیست ها تغییر پذیر (mutable) هستند. یعنی عناصر آنها قابل جایگزینی می باشد.

```
>>> a = [1, 2, 3]
>>> a[0] = 6
Output: [6, 2, 3]
```

- به نظر شما نتیجه اجرای کد زیر چی می تواند باشد؟

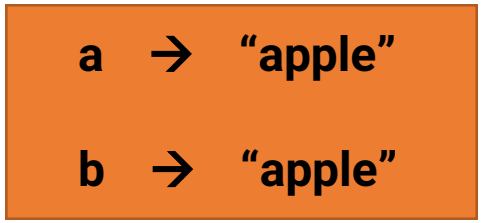
```
>>> name = “Salam”
>>> name[1] = “o”
Output: ?
```

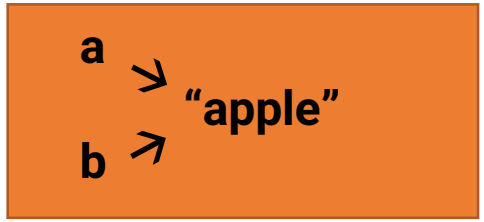
اشیا و مقادیر

- به دو دستور زیر توجه کنید:

```
>>> a = “apple”
>>> b = “apple”
```

- دو حالت ممکن است اتفاق بیافتد.





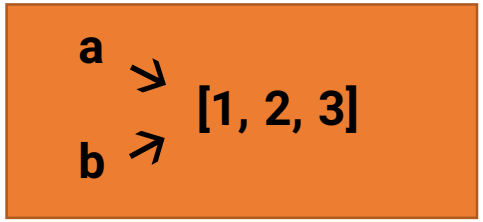
- شی، چیزیست که متغیر بتواند به آن اشاره کند.
- هر شی یک شناسه منحصر بفرد دارد که با تابع id قابل بازیابی می باشد.
- آیا می توان گفت id(a) == id(b) ←

بدل سازی - aliases

- از آنجا که متغیرها به اشیا اشاره می کنند، اگر ما یک متغیر را به متغیر دیگری نسبت دهیم، هر دو به یک چیز اشاره خواهند کرد.

```
>>> a = [1, 2, 3]
>>> b = a
```

- در این حالت نمودار حالت به صورت زیر می شود



- چون در اینجا دو متغیر به یک لیست اشاره می کنند، می گوییم آن لیست بدل شده است.

```
>>> b[0] = 5
>>> print (a)
Output: [5, 2, 3]
```

تکثیر لیست - clone

- اگر بخواهیم لیستی را تغییر دهیم باید یک کپی از آن نگهداری کنیم. به این فرآیند تکثیر گفته می شود.
- ساده ترین روش تکثیر استفاده از عملگر برش است.

```
>>> a = [1, 2, 3]
>>> b = a[:]
```

- گرفتن هر برش از a یک لیست جدید می سازد.
- از متد copy هم می توان استفاده کرد.

```
>>> a = [1, 2, 3]
>>> b = a.copy()
```

ماتریس ها

- لیست های تودرتو اغلب برای نمایش ماتریس استفاده می شوند.

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

- ماتریسی با دو سطر و سه ستون

```
>>> matrix = [[1, 2, 3], [4, 5, 6]]
```

- در اینجا matrix دارای دو عضو می باشد که هر کدام یک سطر ماتریس را نمایش می دهند.
- بازیابی یک سطر کامل

```
>>> matrix [1] → Output: [4, 5, 6]
```

- بازیابی یک عنصر یا درایه

```
>>> matrix [1][1] → Output: 5
```

تمرین

- تمرینات می بایست در تاریخ مشخص به TA در قالب یک فایل PDF ایمیل شود.
- در فایل ارسالی حتما نام و نام خانوادگی خود را بنویسید.