INTRO TO PYTHON FOR DATA SCIENCE

Python Lists

Python Data Types

- `float` - real numbers
- `int` - integer numbers
- `str` - string, text
- `bool` - `True`, `False`

```
In [1]: height = 1.73

In [2]: tall = True
```

- Each variable represents single value

Problem

- Data Science: many data points
- Height of entire family

```
In [3]: height1 = 1.73

In [4]: height2 = 1.68

In [5]: height3 = 1.71

In [6]: height4 = 1.89
```

- Inconvenient

Python List [a, b, c]

```
In [7]: [1.73, 1.68, 1.71, 1.89]
Out[7]: [1.73, 1.68, 1.71, 1.89]

In [8]: fam = [1.73, 1.68, 1.71, 1.89]

In [9]: fam
Out[9]: [1.73, 1.68, 1.71, 1.89]
```

- Name a collection of values
- Contain any type
- Contain different types

Python List [a, b, c]

```
In [10]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [11]: fam
Out[11]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
```

```
["liz", 1.73]
["emma", 1.68]
["mom", 1.71]
["dad", 1.89]
```

Python List [a, b, c]

```
In [10]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [11]: fam
Out[11]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]


In [11]: fam2 = [["liz", 1.73],
                 ["emma", 1.68],
                 ["mom", 1.71],
                 ["dad", 1.89]]

In [12]: fam2
Out[12]: [['liz', 1.73], ['emma', 1.68],
               ['mom', 1.71], ['dad', 1.89]]
```

List type

```
In [13]: type(fam)
Out[13]: list

In [14]: type(fam2)
Out[14]: list
```

- Specific functionality
- Specific behavior

INTRO TO PYTHON FOR DATA SCIENCE

Let’s practice!

INTRO TO PYTHON FOR DATA SCIENCE

Subsetting Lists

Subsetting lists

```
In [1]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [2]: fam
Out[2]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
```

```
    index:   0      1      2      3      4      5      6      7
```

"zero-based indexing"

Subsetting lists

```
In [1]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [2]: fam
Out[2]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
    index:   0      1       2      3      4      5      6      7
```

```
In [3]: fam[3]
Out[3]: 1.68
```

Subsetting lists

```
In [1]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [2]: fam
Out[2]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
    index:   0      1       2       3      4      5      6      7
```

```
In [3]: fam[3]
Out[3]: 1.68

In [4]: fam[6]
Out[4]: 'dad'
```

Subsetting lists

```
In [1]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [2]: fam
Out[2]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
```

index:	0	1	2	3	4	5	6	7
	-8	-7	-6	-5	-4	-3	-2	-1

```
In [3]: fam[3]
Out[3]: 1.68

In [4]: fam[6]
Out[4]: 'dad'

In [5]: fam[-1]
Out[5]: 1.89
```

Subsetting lists

```
In [1]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [2]: fam
Out[2]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
```

index:	0	1	2	3	4	5	6	7
	-8	-7	-6	-5	-4	-3	-2	-1

```
In [3]: fam[3]
Out[3]: 1.68

In [4]: fam[6]
Out[4]: 'dad'

In [5]: fam[-1]
Out[5]: 1.89

In [6]: fam[-2]
Out[6]: 'dad'
```

List slicing

```
In [7]: fam
Out[7]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
           0      1      2      3      4      5      6      7

In [8]: fam[3:5]
Out[8]: [1.68, 'mom']
```

```
[ start : end ]
```

inclusive exclusive

List slicing

```
In [7]: fam
Out[7]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
           0      1       2      3     4     5      6      7

In [8]: fam[3:5]
Out[8]: [1.68, 'mom']

In [9]: fam[1:4]
Out[9]: [1.73, 'emma', 1.68]
```

[start : end]

inclusive **exclusive**

List slicing

```
In [7]: fam
Out[7]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
          0      1      2      3      4      5      6      7

In [8]: fam[3:5]
Out[8]: [1.68, 'mom']

In [9]: fam[1:4]
Out[9]: [1.73, 'emma', 1.68]

In [10]: fam[:4]
Out[10]: ['liz', 1.73, 'emma', 1.68]
```

List slicing

```
In [7]: fam
Out[7]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]
          0     1      2      3      4     5      6      7

In [8]: fam[3:5]
Out[8]: [1.68, 'mom']

In [9]: fam[1:4]
Out[9]: [1.73, 'emma', 1.68]

In [10]: fam[:4]
Out[10]: ['liz', 1.73, 'emma', 1.68]

In [11]: fam[5:]
Out[11]: [1.71, 'dad', 1.89]
```

INTRO TO PYTHON FOR DATA SCIENCE

Let’s practice!

INTRO TO PYTHON FOR DATA SCIENCE

Manipulating Lists

List Manipulation

- Change list elements
- Add list elements
- Remove list elements

Changing list elements

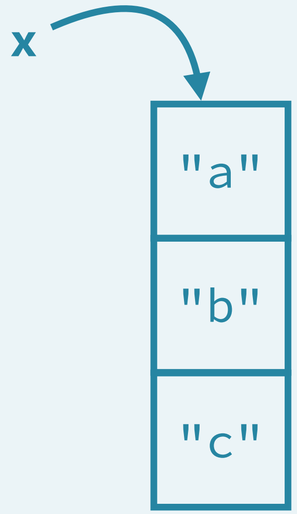
```
In [1]: fam = ["liz", 1.73, "emma", 1.68, "mom", 1.71, "dad", 1.89]

In [2]: fam
Out[2]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.89]

In [3]: fam[7] = 1.86

In [4]: fam
Out[4]: ['liz', 1.73, 'emma', 1.68, 'mom', 1.71, 'dad', 1.86]

In [5]: fam[0:2] = ["lisa", 1.74]

In [6]: fam
Out[6]: ['lisa', 1.74, 'emma', 1.68, 'mom', 1.71, 'dad', 1.86]
```

Adding and removing elements

```
In [7]: fam + ["me", 1.79]
Out[7]: ['lisa', 1.74,'emma', 1.68,
                            'mom', 1.71, 'dad', 1.86, 'me', 1.79]

In [8]: fam_ext = fam + ["me", 1.79]



In [9]: del(fam[2])

In [10]: fam
Out[10]: ['lisa', 1.74, 1.68, 'mom', 1.71, 'dad', 1.86]

In [11]: del(fam[2])

In [12]: fam
Out[12]: ['lisa', 1.74, 'mom', 1.71, 'dad', 1.86]
```

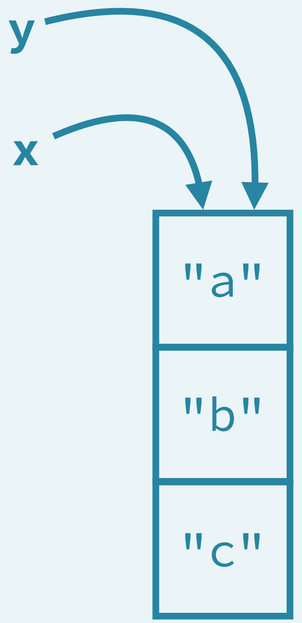
Behind the scenes (1)

```
In [13]: x = ["a", "b", "c"]

In [14]: y = x
```



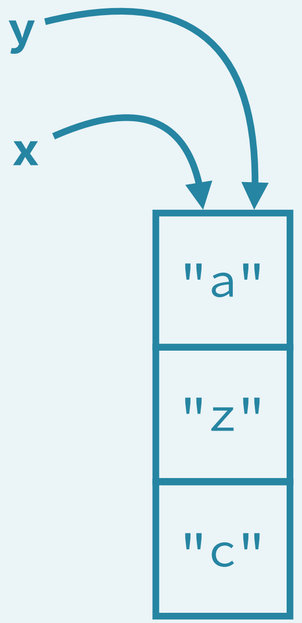
Behind the scenes (1)

```
In [13]: x = ["a", "b", "c"]

In [14]: y = x

In [15]: y[1] = "z"

In [16]: y
Out[16]: ['a', 'z', 'c']

In [17]: x
Out[17]: ['a', 'z', 'c']
```



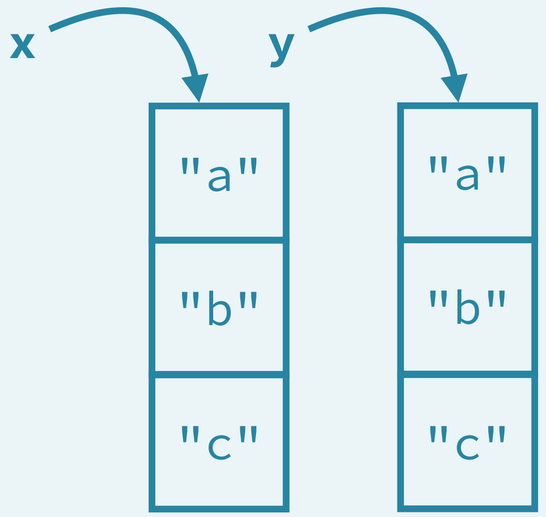
Behind the scenes (1)

```
In [13]: x = ["a", "b", "c"]

In [14]: y = x

In [15]: y[1] = "z"

In [16]: y
Out[16]: ['a', 'z', 'c']

In [17]: x
Out[17]: ['a', 'z', 'c']
```



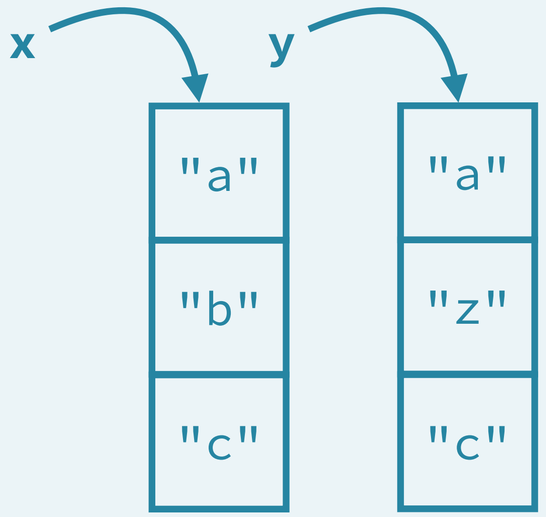
Behind the scenes (2)

```
In [18]: x = ["a", "b", "c"]

In [19]: y = list(x)

In [20]: y = x[:]

In [21]: y[1] = "z"
```



Behind the scenes (2)

```
In [18]: x = ["a", "b", "c"]

In [19]: y = list(x)

In [20]: y = x[:]

In [21]: y[1] = "z"

In [22]: x
Out[22]: ['a', 'b', 'c']
```



INTRO TO PYTHON FOR DATA SCIENCE

Let’s practice!