



Queen Mary  
University of London

**QHP4701**

# **Introduction to Data Science Programming**

Collection(s) of Data: List/Set/Dictionary

Lecturer: Nikesh Bajaj, PhD

School of Physical and Chemical Sciences

<http://nikeshbajaj.in>

# Lecture Outline

- Collection(s) of Data
- List and List of Lists
- Tuple
- Sets
- Dictionary
- Referencing and Mutability in Python
- Mixing the collections

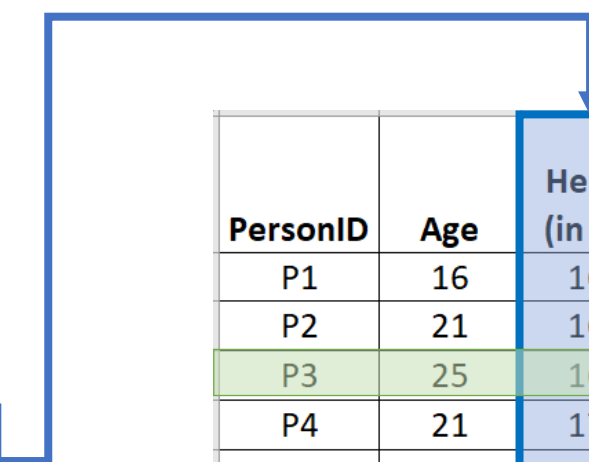
# Collection of Data

- Basic Data Types hold a single value
- For recording sequence or collection of single values we need an efficient representation

Let's check an example from Table 1.1

H = 168, 160, 169, 170, 168, 165, 165, 170, 171, 168

P3 = 25, 169, 69.01, None, UG



PersonID	Age	Height (in cm)	Weight (in Kg)	Address	Education Level
P1	16	168	60.50	E16 3LW, Lon	Highschool
P2	21	160	61.30	E16 3LW, Lon	UG
P3	25	169	69.01	None	UG
P4	21	170	70.60	E16 3LW, Lon	UG
P5	23	168	59.10	E16 3LW, Lon	PG
P6	32	165	55.89	None	PhD
P7	None	165	59.00	E16 3LW, Lon	Highschool
P8	42	170	65.00	E16 3LW, Lon	Phd
P9	28	171	76.60	E16 3LW, Lon	PG
P10	27	168	79.90	E16 3LW, Lon	PG

Think about:

Text – sequence of words, Image – grid of pixels, Speech – sequence of values

# Collection of Data

Following Data Types in Python hold collection of values/objects

- List
- Tuple
- Set
- Dictionary

$$X = \left[ \textit{Object} \ \textit{Object} \ \textit{Object} \ \textit{Object} \right]$$

*Objects*

• **Numerical:** *int, float*   • **Strings:** *str*   • **Boolean:** *bool*   • **NoneType:** *None*

***Collection of objects***

*Each of them have different structure, purpose and operations, we will discuss each one by one*

# Lecture Outline

- Collection(s) of Data
- List and List of Lists
- Tuple
- Sets
- Dictionary
- Referencing and Mutability in Python
- Mixing the collections

# List

## List

- Collection of elements (**objects** or items)
- Most versatile and efficient data type
- Could be all of same data types or different

[ *Object* *Object* *Object* *Object* ]

[ ● ■ ■ ● ]

- Examples

[1, 2, 4, 10, -10]

['Apple', 'Oranges', 'Banana']

['A', 1, 'B', -19, None]

['P3', 25, 169, 69.01, '145, South London, SW12 LQ2, UK', None]

*an element or an item*

# Operations on List

## List

```
c = [-45, 6, 0, 72, 1543]
```

### Creating a List

- Length of a List *len(c)*
- List of sequential numbers (integers)

```
c = list(range(5))
```

```
c = [0, 1, 2, 3, 4]
```

```
c = list(range(1,10,2))
```

```
c = [1,3,5,7,9]
```

```
range(end)
```

```
range(start, end)
```

```
range(start, end, step)
```

*By default **start** is set to 0 and **step** is set to 1*

# Operations on List

## Selecting Element(s)

- Indexing

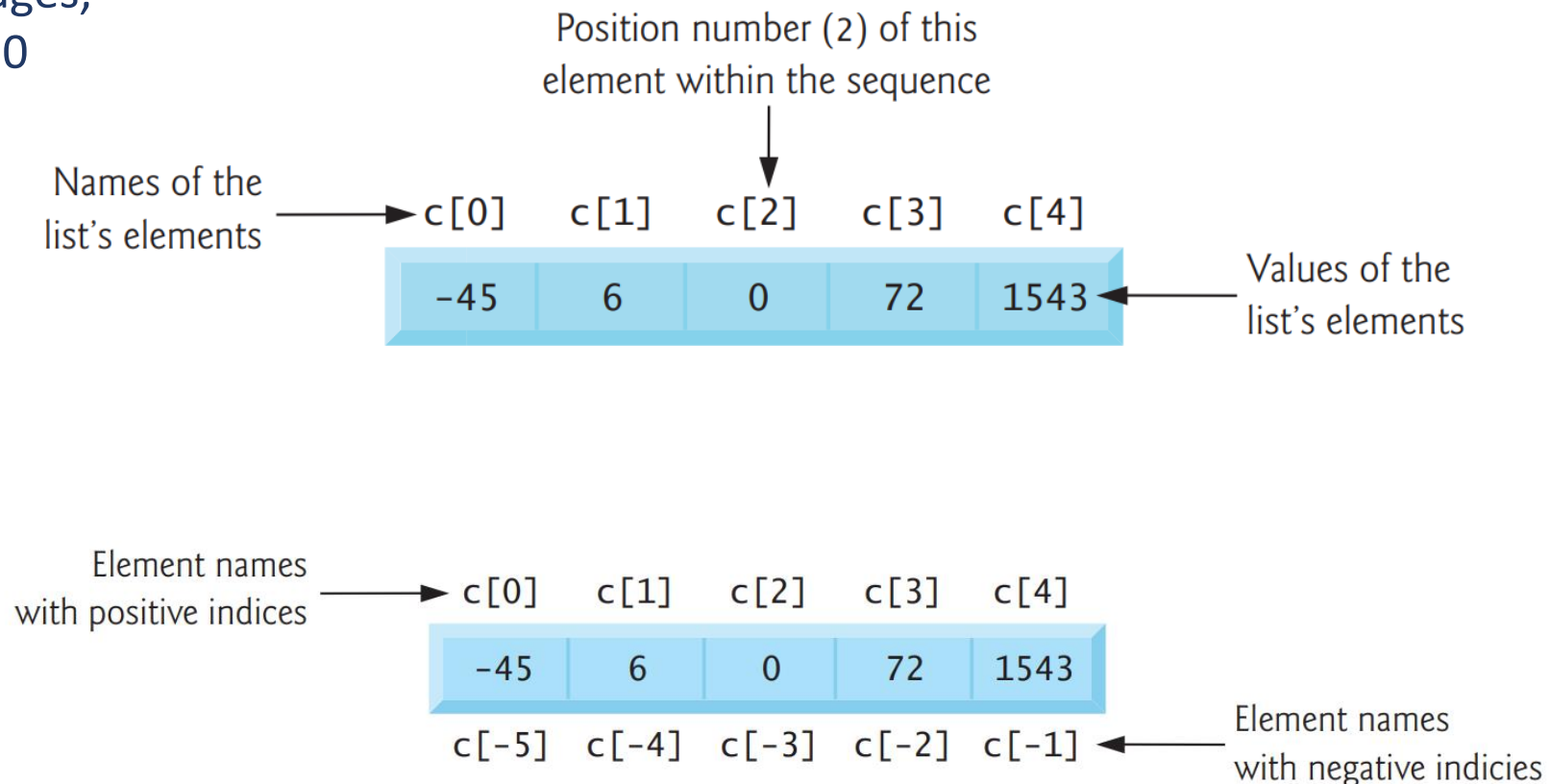
As most programming languages,  
Indexing in python starts with 0

```
c = [-45, 6, 0, 72, 1543]
```

C[0]

C[3]

C[-1]





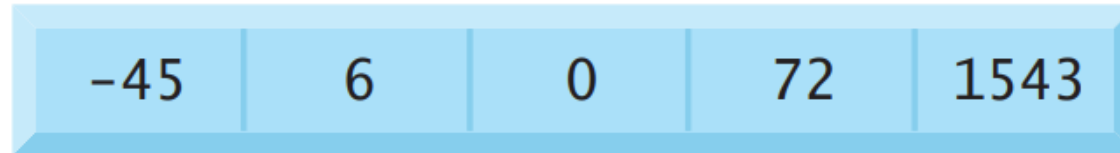
# Operations on List

```
c = [-45, 6, 0, 72, 1543]
```

Selecting element(s):

- Slicing

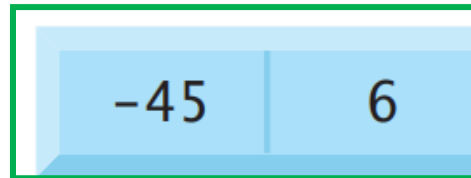
c =



c[1:3]



c[:2]



c[2:]



*Excluding*

c[starts: end]  
c[start: end: **step**]  
c[:: **step**]

Try: c[-2:], c[::2], c[::-1]

# Operations on List

## Adding element(s) to a list

- Append
  - One element at a time
- Extend
  - Multiple elements
- +
  - Concatenation of two lists



c.append(5)

c.extend([5,10])

c.extend(c)

x = c + d

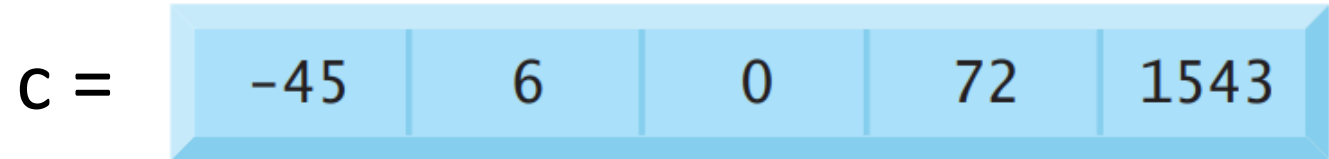
In python,  
'op\_name()' is called method

# Operations on List

Removing element(s) to a list

- Remove

- One element at a time



c.remove(0)

- Pop

- Remove and return

z = c.pop(1)

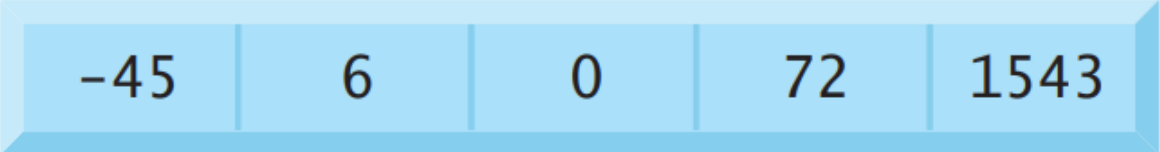
- Clear

- Clear everything

c.clear()

# Operations on List

Change/replace element(s) of list

c = 

- One element at a time

$c[0] = 10$

- Multiple

$c[1:3] = [0,0]$

## Operations on List: Try

```
x = [1, 1, 0, 3, 2]
```

```
len(x)
```

```
y = x*2
```



























































```
y = x + x
```

```
x[10]
```

```
x[-5]
```

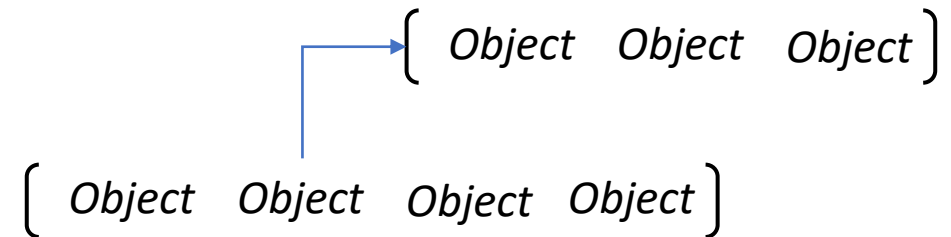
```
x[::-2]
```

## 'List' of methods

[     ]	.append(  )	[      ]
[    ]	.count(  )	2
[    ]	.insert(1,  )	[     ]
[    ]	.remove(  )	[   ]
[    ]	.index(  )	2
[    ]	.pop(1)	[   ]
[    ]	.sort( )	[    ]
[    ]	.clear( )	[ ]
[    ]	.copy( )	[    ]
[    ]	.reverse( )	[    ]

# List of Lists

- List of Lists
  - A list can have an element that is also a list



- $C = [ [1, 2, 4, 5, 7, 2],$   
     $[4, 5],$   
     $[6, 8, 4]]$

# List of Lists: as matrix

- List of lists

```
a = [[77, 68, 86, 73],  
     [96, 87, 89, 81],  
     [70, 90, 86, 81]]
```

	Column 0	Column 1	Column 2	Column 3
Row 0	77	68	86	73
Row 1	96	87	89	81
Row 2	70	90	86	81

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Column index  
Row index  
List name



# Matrix Multiplication

- $A = \begin{bmatrix} [1, 2, 3], \\ [2, 0, 1], \\ [1, 0, 1] \end{bmatrix}$

- $B = \begin{bmatrix} [2, 2, 3], \\ [1, 1, 0], \\ [2, 1, 0] \end{bmatrix}$

compute

- $C = A \times B$

# Lecture Outline

- Collection(s) of Data
- List and List of Lists
- Tuple
- Sets
- Dictionary
- Referencing and Mutability in Python
- Mixing the collections

## Tuple and Immutability

Tuple and List share same structure. The key difference in tuple are immutable.

- `a = (1,2,3)`  
`a[0] = 10`
- `b = (1,2,4, [1,2,4,5],None, 'A')`  
`b[3][0]=10`

List has more methods than tuple

# Lecture Outline

- Collection(s) of Data
- List and List of Lists
- Tuple
- Sets
- Dictionary
- Referencing and Mutability in Python
- Mixing the collections

# Sets

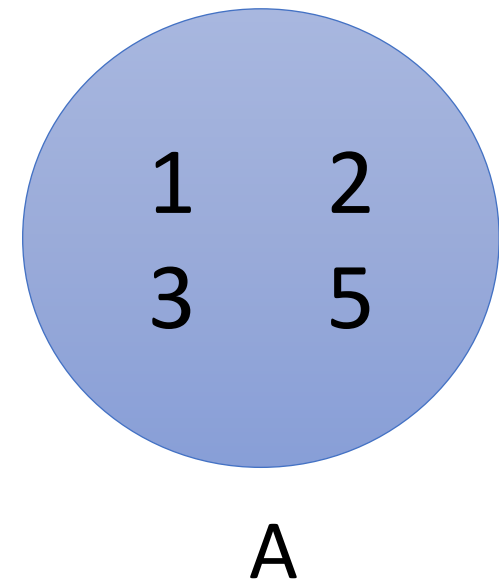
As defined in mathematics, a set is a collection of unique elements

## Creating a set

- `A = set([1, 2, 3, 3, 5])`
- `B = {1,2,3,3,5}`

## Adding /removing an element

- `A.add(5)`
- `A.add(8)`
- `A.remove(1)`
- `A.discard(10)`



# Operation on Sets

- Union  $A \cup B$

- $A \cup B$
- `A.union(B)`

- Intersection  $A \cap B$

- $A \cap B$
- `A.intersection(B)`

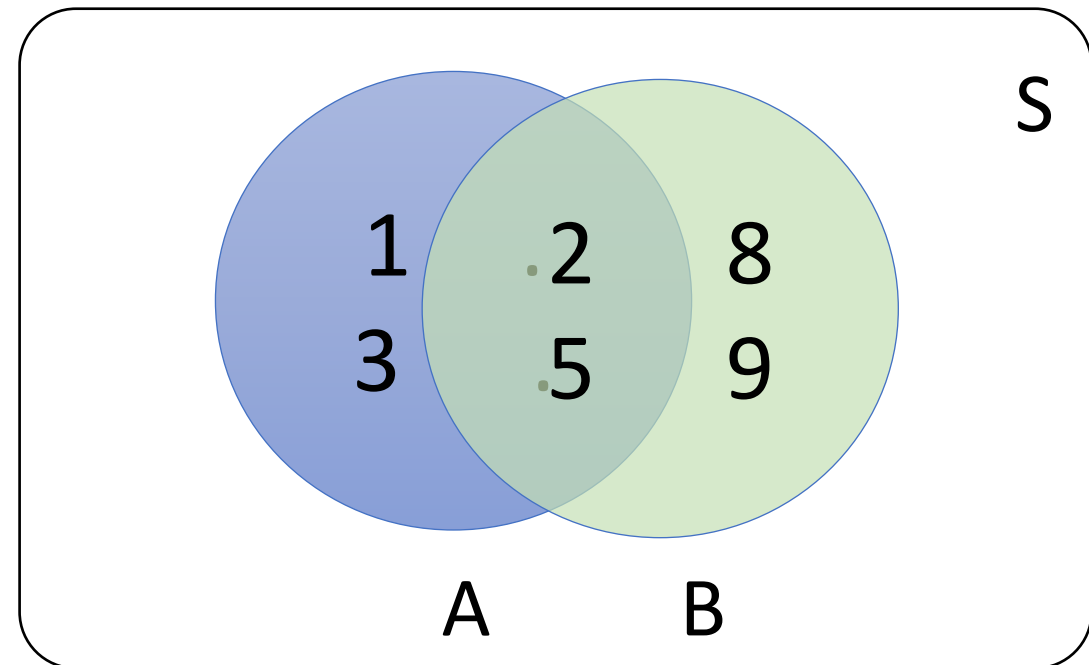
- Differences  $A - B$

- $A - B$
- `A.difference(B)`

$$S = \{1, 2, \dots 9, 10\}$$

$$A = \{1, 2, 3, 5\}$$

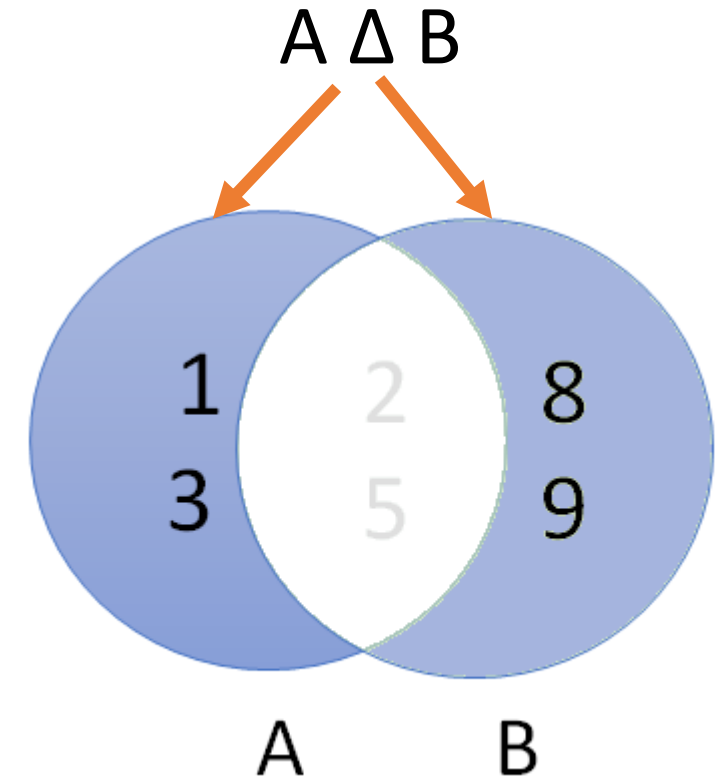
$$B = \{2, 5, 8, 9\}$$



*Q: Where can we use it in Data Analysis?*

## Sets: more operations

- `A.symmetric_difference(B)`  $A \Delta B$   $A \Delta B$
- `A == B`
- `A.issubset(B)`  $A \leq B$   $x1 < B$
- `A.issuperset(B)`  $A \geq B$   $x1 > B$
- `A.update(B[, C ...])`  $A |= B$
- `A.isdisjoint(B)`
- `A.pop()`
- `A.clear()`



$A = \{1, 2, 3, 5\}$

$C = \{5, 1, 2, 3\}$

# Lecture Outline

- Collection(s) of Data
- List and List of Lists
- Tuple
- Sets
- Dictionary
- Referencing and Mutability in Python
- Mixing the collections



## Dictionary : Mapping Type

- Dictionary in Python are great way to store data or any mapping pairs.
- Dictionary has keys, values pair.

A → 1

B → 2

C → 3

### Three ways to create dictionary

- X = { 'A': 1, 'B':2, 'C':3}
- X = `dict`([('A',1), ('B',2), ('C',3)])
- X = `dict`(A=1, B=2,C=3)

*Keys in dictionary have to be unique*

Data → [1,2,3,5,3,2,4]

photo → Image

Grades → ['A', 'A', 'B', 'A']

# Operation on Dictionary

- Accessing values by keys
  - `X['A']`
- All the keys:
  - `X.keys()`
- All the values:
  - `X.values()`
- Changing value
  - `X['A']=10`
- Deleting key-value pair
  - `del X['A']`

`X = { 'A': 1, 'B':2, 'C':3 }`

# Lecture Outline

- Collection(s) of Data
- List and List of Lists
- Tuple
- Sets
- Dictionary
- Referencing and Mutability in Python
- Mixing the collections



# Mutability in Python

- Mutable types

- List is mutable type: an element can be changed

```
A = [1,2,4]
```

```
A[0] = 10
```

- Immutable

- Tuple is immutable type : an element can NOT be changed

```
A = (1,2,4)
```

```
A[0]=10
```

- String is immutable type

```
X = "Hello!"
```

```
X[0]='e'
```

*String in python is one of special data type that has wide variety of functionalities*

# Lecture Outline

- Collection(s) of Data
- List and List of Lists
- Tuple
- Sets
- Dictionaries
- Referencing and Mutability in Python
- Mixing the collections

# Mixing the collections

- List of Lists
- List of any type
  - List of tuples, dictionaries, sets or mix of them
- Tuple of Tuples
- Tuple of any type
  - tuple of lists, dictionaries, sets or mix of them
- Dictionary
  - keys, a list can not a key, but tuples can be
  - Anything can be in values
- Sets
  - Element of a set can be tuple but not list or set

*Mixing of **dict** and **set** is not always straightforward*

- Next !!!
  - 1.4: Hands on Collection of Data
    - List
    - Tuples
    - Sets
    - Dictionary
  - 2.1: Numpy Array





Queen Mary

**University of London**