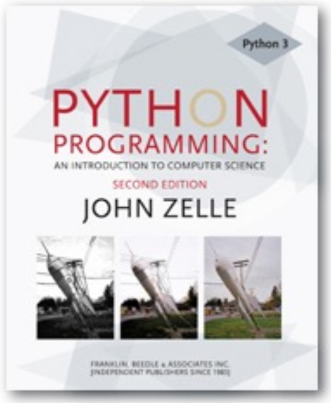
CS 177



Data Collections: Sets and Dictionaries

Objectives for Today

- Overview of Python Data Collections
- Discuss:
  - Sets
  - Dictionaries

Multiple elements stored together are called a sequence

- *Range* - Integers generated using a mathematical sequence and stored in order
- *String* - Characters stored in order
- *List* – Elements stored in order
- These are all structured sequences which can be accessed by indexing

```
>>> r=range(1,10)
>>> r[3]
4
```

```
>>> s='boiler'
>>> s[3]
'l'
```

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

Sequences are flexible and useful in many ways

```
myList = [ 5, 8, 2, 4, 1, 5 ]
```

- We can slice them…

```
>>> myList[3:7]
```

- iterate over them in loops…

```
>>> for n in myList:
```

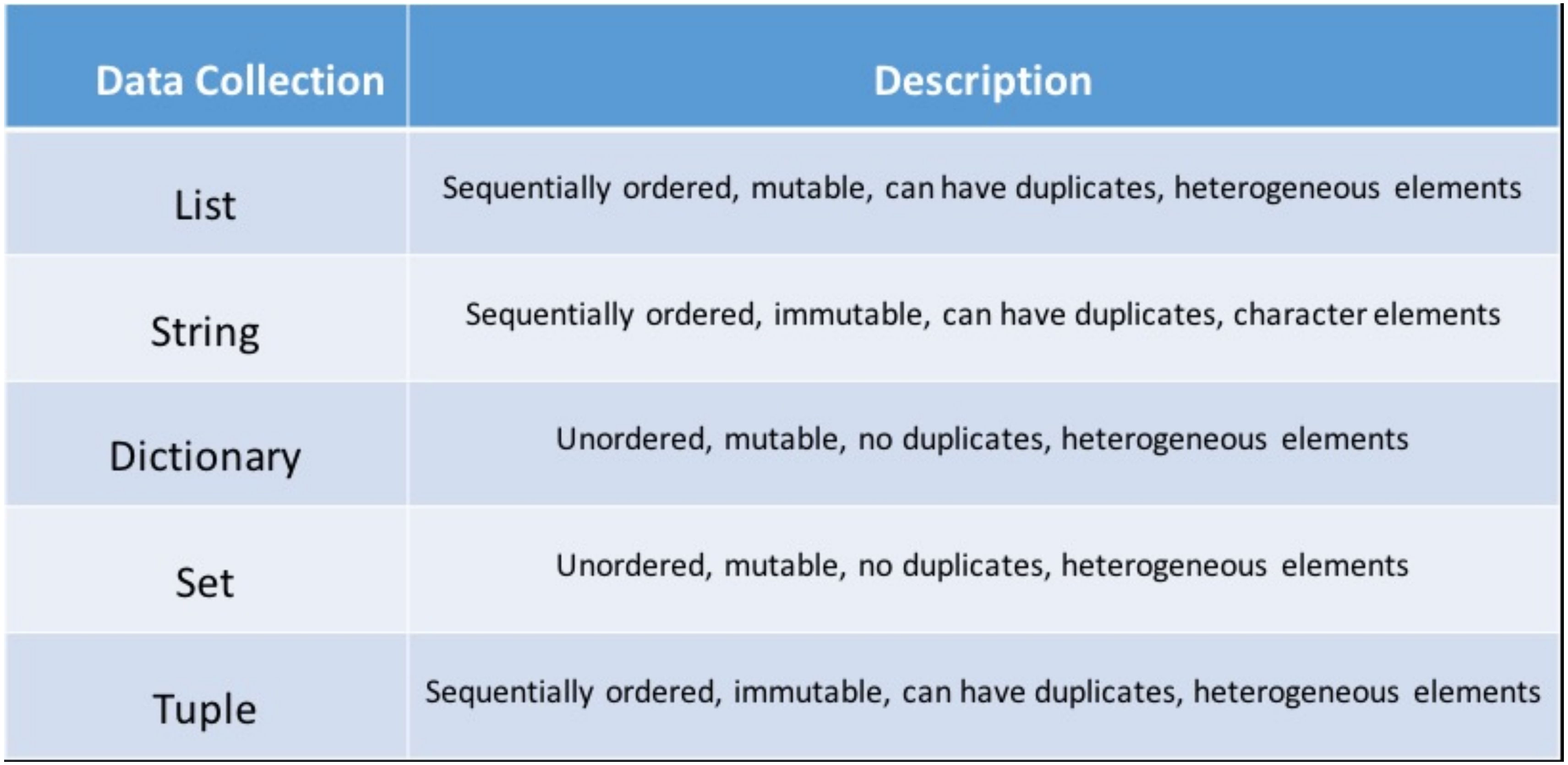
- and check them for membership:

```
>>> if 4 in myList:
```

Python has some special sequence types too

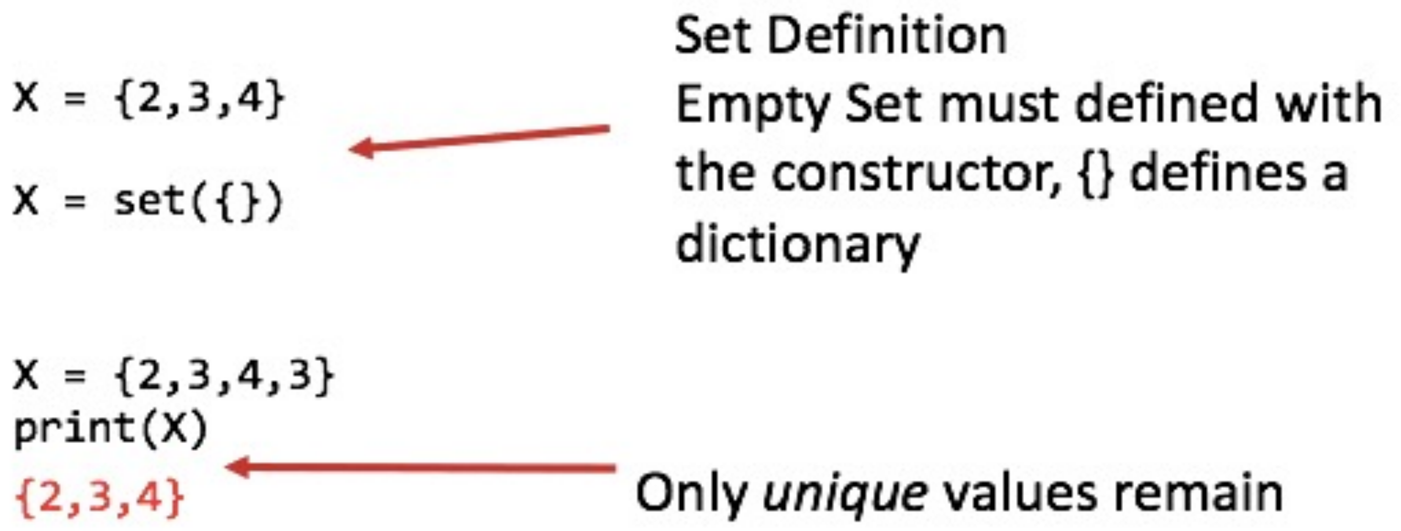
- *Tuple* – <u>Ordered</u> and <u>immutable</u> elements
- *Set* – <u>Unordered</u> and non-duplicated values
- *Dictionary* – <u>Unordered</u> values accessed by key-value pairs
- <u>Immutable</u> sequences *can not* be changed
- <u>Unordered</u> sequences *are not* accessible using indexing

Data Collection Differences



Data Collection	Description
List	Sequentially ordered, mutable, can have duplicates, heterogeneous elements
String	Sequentially ordered, immutable, can have duplicates, character elements
Dictionary	Unordered, mutable, no duplicates, heterogeneous elements
Set	Unordered, mutable, no duplicates, heterogeneous elements
Tuple	Sequentially ordered, immutable, can have duplicates, heterogeneous elements

Sets are a unique data type using { } in their definition



```
X = {2,3,4}

X = set({})
```

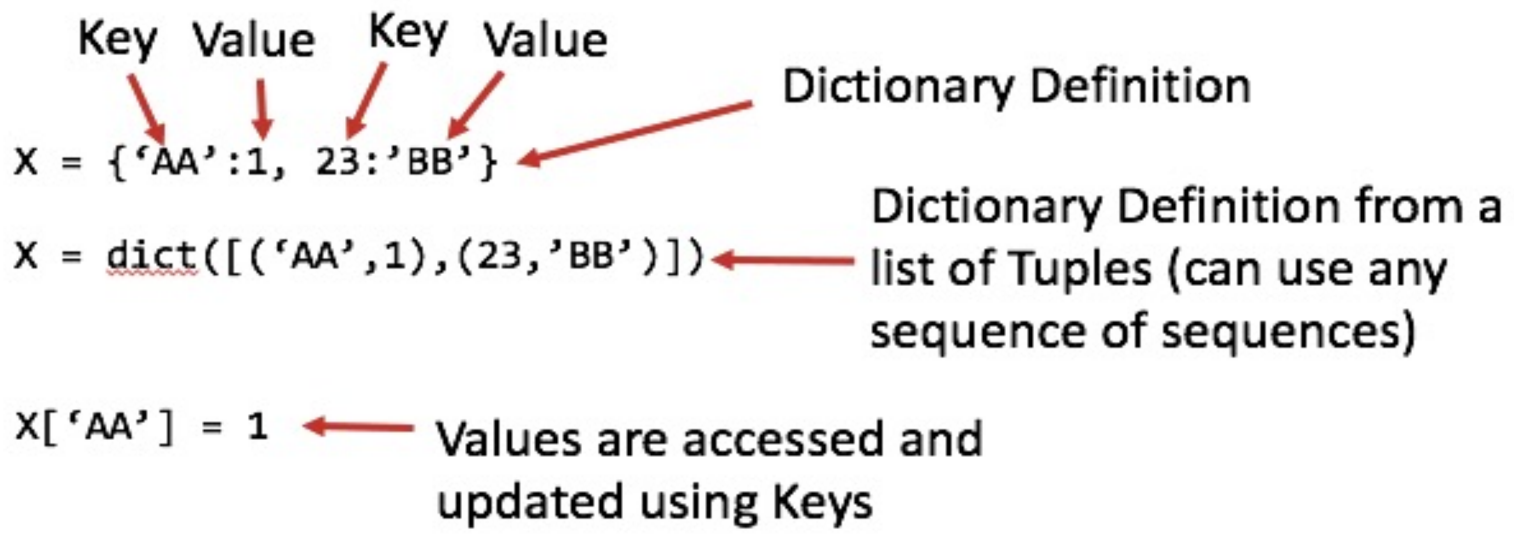
Set Definition
Empty Set must defined with the constructor, {} defines a dictionary

```
X = {2,3,4,3}
print(X)
{2,3,4}
```

Only *unique* values remain

A *Set* can be used to eliminate duplicate values from another sequence type

Dictionaries are unordered and accessed using keys



```
X = {‘AA’:1, 23:’BB’}
```

Dictionary Definition

```
X = dict([(‘AA’,1),(23,’BB’)])
```

Dictionary Definition from a list of Tuples (can use any sequence of sequences)

```
X[‘AA’] = 1
```

Values are accessed and updated using Keys

Think of a *Dictionary* as a collection of (key,value) pairs with a value associated with each key.

Dictionary values can be any data type

```
>>> myDict = {'sky':'blue', 'grass':'green'}
>>> quizScores = {'rgeorges':(10,10,10),
                  'janderson':(10,8,10)}
>>> circles = {'red':[CircleObject,5,-3],
               'blue':[CircleObject,3,2],
               'green':[CircleObject,-6,4]}
```

This gives the *Dictionary* (key, value) pairs the flexibility to store an almost unlimited amount of data!

Check your understanding: *Write the Python code to…*

1. Separate the words in a paragraph (stored as a single *String*) into a *List* named `words`
2. Create a *Set* named `unique` containing only the unique entries from the *List* `words` (<u>no duplicates</u>)
3. Define a *Dictionary* where the keys are the entries in the *Set* `unique` and the corresponding values are a count of times they occur in the *List* `words`

String: `'a red balloon and a blue balloon are joined into a red and blue balloon'`

List: `['a', 'red', 'balloon', 'and', 'a', 'blue', 'balloon', 'are', 'joined', 'into', 'a', 'red', 'and', 'blue', 'balloon']`

Dictionary: `{'are': 1, 'blue': 2, 'joined': 1, 'balloon': 3, 'red': 2, 'a': 3, 'and': 2, 'into': 1}`

One way to solve…

```
def main():
    # take the input
    para = input('Enter the paragraph:')
    # use .split() method to extract the words
    words = para.split()
    # use set() to find the unique words
    unique_words = set(words)
    # initialize an empty dictionary
    words_dict = {}
    # use a for loop to iterate over the keys of
    # the dictionary
    for word in unique_words:
        # use .count() method to find number of occurences
        words_dict[word] = words.count(word)
    print(words_dict)

main()
```