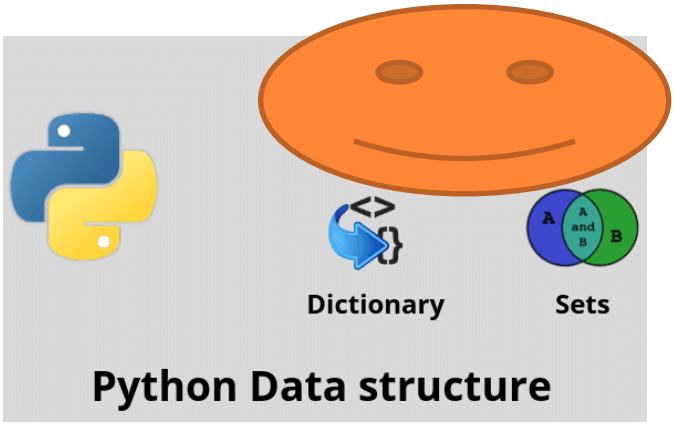
Data structures in Python

Week 8: Dictionary

Learning objectives

- Creating Python Dictionaries and sets
- Keys and values in Dictionaries
- Sets properties
- Creating and using modules in Python
- Importing user created modules in Python

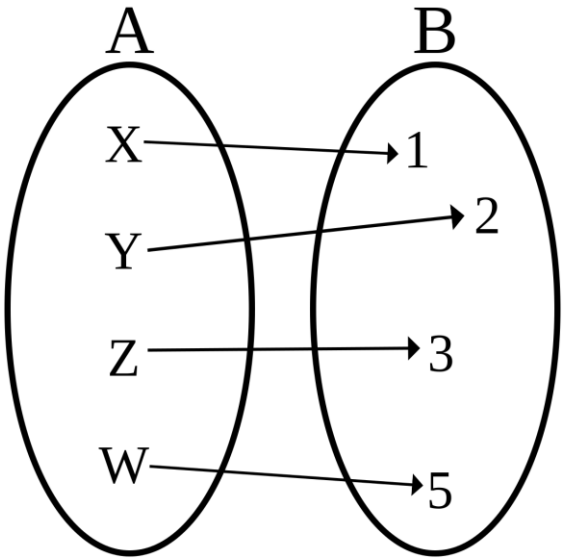
At the conclusion of this lecture, students will be able to understand the use of basic data structures and able to write programs using Sets and Dictionaries. In addition, student learn how to import and use user defined modules in their programs.



Dictionary

What is dictionary?

- It is **defined general purpose data structure** to store group of objects.
- It has set of keys and each key is associated with single value.
- Dictionary in Python contains collection of ordered values. The values of dictionary are accessed by keys.



- **How to define a dictionary?**
- Suppose we want to store student id and their respective quiz scores. Dictionary can be handy here to define student id as a key and quiz score is an associated value of it.

example 1.py *

```
1 Quizscores = {"x123" : 17, # here x123 is a key and 17 is a value..
2               "x124" : 54,
3               "x125" : 62,
4               "x126" : 75,
5               "x127" : 92,
6               "x128" : 49}
7 print(Quizscores)
```

Shell

```
Python 3.7.9 (bundled)
>>> %cd 'Z:\Python 2021_Fall\Fall 2021_CT60A0203\Week 8\Lecture slides'
>>> %Run 'example 1.py'
>>> %Run 'example 1.py'
  {'x123': 17, 'x124': 54, 'x125': 62, 'x126': 75, 'x127': 92, 'x128': 49}
```

- Dictionary elements are defined inside { }. Names of key must be unique, but values can be duplicated. That is , Dictionaries cannot have two values with the same key.
- What will happen if we use the name of the key more than once?

Thonny - Z:\Python 2021_Fall\Fall 2021_CT60A0203\Week 8\Lecture slides\example 1.py @ 6 : 24

File Edit View Run Tools Help

example 1.py *

```
1 Quizscores = {"x123" : 17, # here x123 is a key and 17 is a value..
2               "x124" : 54,
3               "x125" : 62,
4               "x126" : 75,
5               "x129" : 92,
6               "x124" : 49}
7 print(Quizscores)
```

```
>>> %Run 'example 1.py'
  {'x123': 17, 'x124': 49, 'x125': 62, 'x126': 75, 'x127': 92}
>>>
```

Dictionary does not allow duplicates, so the value of it will be overwritten.

- **How to access the elements of Dictionary?**

example 1.py

```
1  Quizscores = {"x123" : 17, # here x123 is a key and 17 is a value..
2                "x124" : 54,
3                "x125" : 62,
4                "x126" : 75,
5                "x129" : 92,
6                "x127" : 62}
7  print(Quizscores)
8  print(Quizscores.keys()) # displays all keys
9  print(Quizscores.values()) # displays all values
10 d1 = {} # creating an empty dictionary
11 print(d1.keys())
```

So, keys(), values(), items() are for?

Shell

```
Python 3.7.9 (bundled)
>>> %Run 'example 1.py'

  {'x123': 17, 'x124': 54, 'x125': 62, 'x126': 75, 'x129': 92, 'x127': 62}
  dict_keys(['x123', 'x124', 'x125', 'x126', 'x129', 'x127'])
  dict_values([17, 54, 62, 75, 92, 62])
  dict_keys([])
```

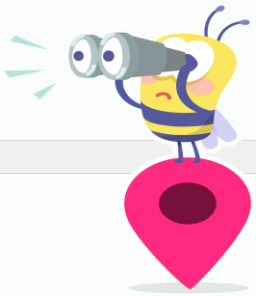
example 2 .py

```
1  Quizscores = {"x123" : 17, # here x123 is a key and 17 is a value.
2                "x124" : 54,
3                "x125" : 62,
4                "x126" : 75,
5                "x129" : 92,
6                "x127" : 62}
7  #iterating
8  for key, value in Quizscores.items(): # iterating
9      print (key, value)
10
11 # displaying quiz scores above 75
12 print("---------------------------------")
13 for x, y in Quizscores.items():
14     if y>75:
15         print (x,y)
16
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run 'example 2 .py'

  x123 17
  x124 54
  x125 62
  x126 75
  x129 92
  x127 62
  ---------------------------------
  x129 92
>>>
```



What is the difference between print(Quizscores) and print(Quizscores.items())?

Dictionary.. Iterating in dictionary items (keys and values)

example 3 .py

```
Quizscores = {"x123" : 17, # here x123 is a key and 17 is a value..
              "x124" : 54,
              "x125" : 62,
              "x126" : 75,
              "x129" : 92,
              "x127" : 62}

#iterating
for k in Quizscores: # iterating
    print (k, Quizscores[k]) # here key is a index
print("----------------------")
#iterating and print the key that in index 3
print([key for key in Quizscores.keys()][3])
print("----------------------")
#similarly print value that in index  4
print([value for value in Quizscores.values()][4])
print("----------------------")
#how about printing both fetching from specific index key and value
print(([key for key in Quizscores.keys()][2], [value for value in Quizscores.values()][2]))
print("----------------------")
#Is it possible to fetch different index and value. Why not?
print(([key for key in Quizscores.keys()][2], [value for value in Quizscores.values()][1]))
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run 'example 3 .py'

  x123 17
  x124 54
  x125 62
  x126 75
  x129 92
  x127 62
  ----------------------
  x126
  ----------------------
  92
  ----------------------
  ('x125', 62)
  ----------------------
  ('x125', 54)
```

- **Dictionary.. Some more**
- Suppose you want to add, update, and delete items in/of dictionary

example 4 .py

```
1  Quizscores = {"x123" : 17, # here x123 is a key and 17 is a value..
2              "x124" : 54,
3              "x125" : 62,
4              "x126" : 75,
5              "x129" : 92,
6              "x127" : 62}
7
8  #adding updating item (add or update)
9  Quizscores.update({"x128":31}) #adding
10 print(Quizscores.items())
11 print("-----------------")
12 # updating "x128"'s value and adding "x130"
13 Quizscores.update({"x128":41,"x130":84 })
14 print(Quizscores.items())
15 #one more way to add/update
16 x = {"x131":25,"x132":72,"x133":0,"x126":79}
17 Quizscores.update(x)
18 print("-----------------")
19 print(Quizscores.items())
20 #deleting a specific key value
21 del Quizscores["x133"]
22 print("-----------------")
23 print(Quizscores.items())
```

del Quizscores[key] remove specific key and its value
del Quizscores[] removes entire dictionary
Then what Quizscores.pop[key] does?
What is the difference between pop(), popitem() and del?

Shell

```
Python 3.7.9 (bundled)
>>> %Run 'example 4 .py'

  dict_items([('x123', 17), ('x124', 54), ('x125', 62), ('x126', 75), ('x129', 92), ('x127', 62), ('x128', 31)])
  -----------------
  dict_items([('x123', 17), ('x124', 54), ('x125', 62), ('x126', 75), ('x129', 92), ('x127', 62), ('x128', 41), ('x130', 84)])
  -----------------
  dict_items([('x123', 17), ('x124', 54), ('x125', 62), ('x126', 79), ('x129', 92), ('x127', 62), ('x128', 41), ('x130', 84), ('x1
  31', 25), ('x132', 72), ('x133', 0)])
  -----------------
  dict_items([('x123', 17), ('x124', 54), ('x125', 62), ('x126', 79), ('x129', 92), ('x127', 62), ('x128', 41), ('x130', 84), ('x1
  31', 25), ('x132', 72)])

>>>
```

- **OK. let us solve some tasks for dictionary**

```
cellDict = {
  'Apple': 100,
  'Samsung': 200,
  'Nokia': 400,
  'Sony':600,
  'Huawei':350
}
```

(i) The dictionary **cellDict {}** contains cell phones and their stock value. Search for cell phone "**Nokia**" and delete if it exists. Then print the elements of **cellDict {}**

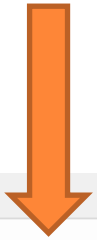
(ii) Update the stock value for **"Sony"** → **350** and add a new item "**LG**" with value **150**.

(iii) List the cellphones that with stock values below 250.

(iv) Create an empty dictionary and copy Samsung details from **cellDict {}** to it.

(i) The dictionary cellDict {} contains cell phones and stock value. Search for "Nokia" and delete if it exists. Then print the elements of cellDict {}

```
cellDict = {
  'Apple': 100,
  'Samsung': 200,
  'Nokia': 400,
  'Sony':600,
  'Huawei':350
}
```



example 5 .py

```
1  cellDict = {
2    'Apple': 100,
3    'Samsung': 200,
4    'Nokia': 400,
5    'Sony':600,
6    'Huawei':350
7  }
8
9  print(cellDict)
10 #Searching for Nokia and remove it from dictonary if exists
11 if 'Nokia' in cellDict:
12     del cellDict['Nokia']
13
14 print('Dict after deleting key =',cellDict)
```

Shell

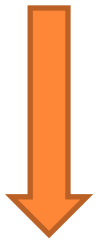
```
Python 3.7.9 (bundled)
>>> %Run 'example 5 .py'

  {'Apple': 100, 'Samsung': 200, 'Nokia': 400, 'Sony': 600, 'Huawei': 350}
  Dict after deleting key = {'Apple': 100, 'Samsung': 200, 'Sony': 600, 'Huawei': 350}

>>>
```

```
cellDict = {
  'Apple': 100,
  'Samsung': 200,
  'Sony':600,
  'Huawei':350
}
```

(ii) Update the stock value for “Sony” → 350 and add a new item “LG” with value 150. (refer slide no. 7)



example 6 .py

```
1  cellDict = {
2    'Apple': 100,
3    'Samsung': 200,
4     'Sony':600,
5    'Huawei':350
6  }
7  print (cellDict)
8  cellDict.update ({'Sony':350, 'LG':150})
9
10 #if 'Sony' in cellDict:
11 #    cellDict.update ({"Sony":350})
12
13 print ("After sony updation + addition of LG:",cellDict)
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run 'example 6 .py'

  {'Apple': 100, 'Samsung': 200, 'Sony': 600, 'Huawei': 350}
  After sony updation + addition of LG: {'Apple': 100, 'Samsung': 200, 'Sony': 350, 'Huawei': 350, 'LG': 150}

>>>
```

```
cellDict = {
  'Apple': 100,
  'Samsung': 200,
  'Sony':350,
  'Huawei':350,
  'LG':150

}
```

(iii) List the cellphone with stock values that are below 250 from cellDict {} (refer slide no. 5)

example 7.py

```
1  cellDict = {
2     'Apple': 100,
3     'Samsung': 200,
4     'Sony':350,
5     'Huawei':350,
6     'LG':150
7
8  }
9  for k, v in cellDict.items():
10       if v<250:
11            print(k,v)
12
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run 'example 7.py'

  Apple 100
  Samsung 200
  LG 150
```

```
cellDict = {
  'Apple': 100,
  'Samsung': 200,
  'Sony':350,
  'Huawei':350,
  'LG':150

}
```

(iv) Create an empty dictionary and copy Samsung details from cellDict {} to it

example ´8.py

```
1  cellDict = {
2    'Apple': 100,
3    'Samsung': 200,
4    'Sony':350,
5    'Huawei':350,
6    'LG':150
7
8  }
9  dict1 = {}
10 for key, value in cellDict.items():
11     if key == 'Samsung':
12         dict1 ={key:value}
13 print (dict1)
14
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run 'example ´8.py'

  {'Samsung': 200}

>>>
```

Reading data from a file and reference put those in a dictionary

*quiz1.txt - Notepad

File Edit Format View Help

```
Abdur,75
Wali,89
Chen,50
Mikko,34
Anita,95
Shah,85
Fatimah,70
Xiaboo,0
Dominik,29
Wang,63
Joy,55
```

filetodict.py

```
1  quizdict = dict()
2  f1 = open("quiz1.txt")
3
4  for quiz in f1:
5      c = quiz.split(",") #split data separated by comma
6      quizdict.update({c[0]:c[1].strip()})
7
8  for k, v in quizdict.items():
9      print(k,v)
```

Shell

```
Python 3.7.9 (bundled)
>>> %Run filetodict.py

 Abdur 75
 Wali 89
 Chen 50
 Mikko 34
 Anita 95
 Shah 85
 Fatimah 70
 Xiaboo 0
 Dominik 29
 Wang 63
 Joy 55

>>>
```

Reading data from file for sorting via dictionary and writing into another file.

person.txt - Notepad

File Edit Format View Help

```
Ali:56
Wali:28
Ram:16
Vijay:56
Abdur:26
Chen:18
Wang:34
Joy:21
Fatimah:22
Anita:37
Fritz:19
```

Ex3d_W8.py

```
1  f1 = open("person.txt")
2  personDict = {}
3  #transferring data from file to dictionary
4  for person in f1:
5      p = person.split(":")
6      personDict.update({p[0]:int(p[1].strip())})
7  #line 6--> name as key and age as value
8  #sorting done by x[0] that is key
9  list1 = sorted(personDict.items(), key=lambda x: x[0], reverse=True)
10 print(list1)
11
12 f2 = open("personsorted.txt","w")
13 for line in list1:
14     f2.write(str(line[0]+","+str(line[1])+"\n"))
15
16 f1.close()
17 f2.close()
```

personsorted.txt - Notepad

File Edit Format View Help

```
Wang,34
Wali,28
Vijay,56
Ram,16
Joy,21
Fritz,19
Fatimah,22
Chen,18
Anita,37
Ali,56
Abdur,26
```

To sort by age(value) in ascending order for example Line no 9 will be:

list1 = sorted(personDict.items(), key=lambda x: x[1])