



Hacettepe University

Computer Engineering Department

Programming in python

BBM103 Introduction to Programming Lab 1
Week 6

Fall 2019

Collections

- **A Collection Groups Similar Things**

List: ordered

Set: unordered, no duplicates

Tuple: unmodifiable list

Dictionary: maps from keys to values

Dictionaries

- Dictionary as an unordered set of *key: value* pairs, with the requirement that the keys are **unique** (within one dictionary).
- Unlike sequences, which are indexed by a range of numbers, dictionaries are indexed by *keys*, which can be any immutable type.

Creating a dictionary:

```
dict = {'Name': 'Zara', 'Age': 7, 'Class': 'First'}
```

Accessing Values in a Dictionary:

```
dict = { 'Name' : 'Zara' , 'Age' : 7 , 'Class' : 'First' }  
print ( "dict['Name'] : " , dict['Name'] )  
print ( "dict['Age'] : " , dict['Age'] )
```

Output:

```
dict['Name'] :   Zara  
dict['Age'] :    7
```

Updating a Dictionary

```
dict = { 'Name' : 'Zara' , 'Age' : 7 , 'Class' : 'First' }  
dict['Age'] = 8  
dict['School'] = "DPS School"  
print ( "dict['Age'] : " , dict['Age'] )  
print ( "dict['School'] : " , dict['School'] )
```

Output:

```
dict['Age'] : 8  
dict['School'] : DPS School
```

SN	Methods with Description
1	dict.clear() : Removes all elements of dictionary dict
2	dict.copy() : Returns a shallow copy of dictionary dict
3	dict.fromkeys() : Create a new dictionary with keys from seq and values set to value.
4	dict.get(key, default=None) : For key key, returns value or default if key not in dictionary
5	dict.has_key(key) : Returns true if key in dictionary dict, false otherwise
6	dict.items() : Returns a list of dict's (key, value) tuple pairs
7	dict.keys() : Returns list of dictionary dict's keys
8	dict.setdefault(key, default=None) : Similar to get(), but will set dict[key]=default if key is not already in dict
9	dict.update(dict2) : Adds dictionary dict2's key-values pairs to dict
10	dict.values() : Returns list of dictionary dict's values

Example:

```
phone_book = {"ahmet öz" : "0532 532 32 32",
"mehmet su": "0543 543 42 42",
"seda naz" : "0533 533 33 33",
"eda ala" : "0212 212 12 12"}
person = input("Please enter a name of a person: ")
if person in phone_book:
    answer = "{} adlı kişinin telefon numarası: {}".format(person, phone_book [person])
    print(answer)
else:
    print("This name is not in this telephone book!")
```

Example:

```
names = ["ahmet", "mehmet", "fırat", "zeynep",  
"selma", "abdullah", "cem"]  
dict = {i: len(i) for i in names}
```

Create a dictionary from a list

File I/O

The open Function:

Example:

```
# Open a file
fo = open("foo.txt", "wb")
print ("Name of the file: ", fo.name)
print ("Closed or not : ", fo.closed)
print ("Opening mode : ", fo.mode)
```

Output:

```
Name of the file: foo.txt
Closed or not : False
Opening mode : wb
```

File I/O

<code>grades.txt</code>
98-86-100-54-63
54-89-78-90-85
0-95-70-69-87-55

Opening files to read:

```
my_file = open("grades.txt", "r")
first_line = my_file.readline()
grades = first_line.split('-')
print ("Grades from the first line: ", grades)
my_file.close()
```

Output:

```
Grades from the first line: ['98', '86', '100', '54', '63\n']
```

File I/O

Opening modes

Sr.No.	Modes & Description
1	r Opens a file for reading only. The file pointer is placed at the beginning of the file. This is the default mode.
2	rb Opens a file for reading only in binary format. The file pointer is placed at the beginning of the file. This is the default mode.
3	r+ Opens a file for both reading and writing. The file pointer placed at the beginning of the file.
4	rb+ Opens a file for both reading and writing in binary format. The file pointer placed at the beginning of the file.
5	w Opens a file for writing only. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.
6	wb Opens a file for writing only in binary format. Overwrites the file if the file exists. If the file does not exist, creates a new file for writing.
7	w+ Opens a file for both writing and reading. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing.
8	wb+ Opens a file for both writing and reading in binary format. Overwrites the existing file if the file exists. If the file does not exist, creates a new file for reading and writing.
9	a Opens a file for appending. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing.
10	ab Opens a file for appending in binary format. The file pointer is at the end of the file if the file exists. That is, the file is in the append mode. If the file does not exist, it creates a new file for writing.
11	a+ Opens a file for both appending and reading. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing.
12	ab+ Opens a file for both appending and reading in binary format. The file pointer is at the end of the file if the file exists. The file opens in the append mode. If the file does not exist, it creates a new file for reading and writing.

Opening files and reading all lines:

```
my_file = open("expenses.txt", "r")
total_expense = 0
for line in my_file.readlines():
    expenses_list = line.split('-')
    for expense in expenses_list:
        total_expense += int(expense)
print("Total expense was:", total_expense)
my_file.close()
```

expenses.txt
100-54-63
78-90-85
70-69-87-55

Output:

Total expense was: 751

Example:

```
file = open("input.txt", "r")
for aline in file.readlines():
    list = aline.split(':')
    print("name:", list[0], "phone number:", list[1])
file.close()
```

input.txt

```
Ahmet Özbudak : 0533 123 23 34
Mehmet Sülün : 0532 212 22 22
Sami Sam : 0542 333 34 34
```

Output:

```
name: Ahmet Özbudak   phone number: 0533 123 23 34
name: Mehmet Sülün   phone number: 0532 212 22 22
name: Sami Sam       phone number: 0542 333 34 34
```

Opening files to write (print output):

```
my_file = open("output.txt", "w")  
my_file.write("I am writing this output to a file")  
my_file.close()
```

Output: The sentence “I am writing this output to a file” will be written into a file named **output.txt**

New function:
`f.write(string)`

writes the contents of *string* to the file, returning the number of characters written.

Opening files to write (print output) cont.:

```
my_file = open("myage.txt", "w")
my_age = 20
my_file.write("I am " + str(my_age) + " years old.")
my_file.close()
```

Output: The sentence "I am 20 years old." will be written into a file named **myage.txt**

file.write(string) takes only one argument, so you need to change any other types into strings and concatenate (+) all parts before passing them as an argument.

Exercise

- Write a program that reads an input file `grades.txt` which stores student names and their grades separated by a colon (:), prints out the name of the student with the highest grade, the name of the student with the lowest grade, and the average grade for this class. Your program should also write the same output to an output file named `class_stats.txt`
- **Note: use a dictionary to store the information from grades.txt**

`grades.txt`

```
Ahmet Özbudak:87  
Mehmet Sülün:99  
Sami Sam:45  
Leyla Tan:93  
Emre Göz:32
```