Python

Agenda – 4

- Files Operations
- **Lab-6**

File Operations

- The **open**() function opens and returns a file handle that can be used to read or write a file.

```
f = open('MyFile.txt', 'r')
print f <open file 'MyFile', mode 'r' at 80a0960>
```

  - The first argument is a string containing the **filename**.
    - An absolute path to the required file could be used
      - **f=open(r'c:\NewFolder\MyFile.txt','r')**
    - Also a relative path to the required file could be used
      - **f=open(r'..\MyFile.txt','r')**

File Operations

- The **open**() function opens and returns a file handle that can be used to read or write a file.

```
f = open('MyFile.txt', 'r')
print f <open file 'MyFile', mode 'r' at 80a0960>
```

  - The second argument is a string containing the **mode**.
    - **'r'** => for read only mode, the default if the mode is omitted
    - **'w'** => for write only mode, an existing file with the same name will be erased and if not exists it will be created.
    - **'a'** => for appending only, if files not exists it will be created.
    - **'r+'** => for read and write mode
    - **'rU'** => Universal read, treats all different line endings as a '\n' . When using universal read, **f.newlines** can be used to know type of new lines chars used in a file.

File Operations

- To read a file's contents, call **f**.**read**(***size***):
  - which reads some quantity of data and returns it as a **string**.
  - ***size*** is an optional numeric argument. When ***size*** is omitted or negative, the entire contents of the file will be read and returned.

```
f.read()  # 'This is the entire file.\n'
f.read()  # ' ' returns an empty string at EOF
```

- **f**.**readline**() reads a single line from the file while **f**.**readlines**() returns a list containing all the file lines.

```
f.readline() # 'This is the first line of the file.\n'
f.readline() # 'Second line of the file.\n'

f.seek(0)

f.readlines() # ['This is the first line of the file.\n', 'Second line of the file.\n']
```

File Operations

- To split the new line character from the read string use:

```
s=f.read().splitlines()
```

```
s=f.readline().rstrip('\n')
```

- For reading lines from a file, you can **loop** over the file object. This is memory efficient, fast, and leads to simple code:

```
for line in f:
  print line,

#This is the first line of the file
#Second line of the file
```

File Operations

- **f**.**write**(***string***) writes the contents of ***string*** to the file:

```
f.write('This is a test\n')
```

  - To write something other than a string, it needs to be converted to a string first:

```
value = ('the answer', 42)
s = str(value)
f.write(s)
```

  - The write process will not be reflected to the file until **f**.**close**() or **f**.**flush**() followed by **os**.**fsync**(**f**) is used.

File Operations

- **f**.**tell**() returns an integer giving the file object's current position in the file, measured in bytes from the beginning of the file.

```
f = open('MyFile.txt', 'r')
f.tell() # 0L

f.readline() # 'This is the first line of the file.\n'
f.tell() #37L

f.readline() # 'Second line of the file.\n'
f.tell() #63L

f.readline()  # ' ' returns an empty string at EOF
f.tell() #63L
```

File Operations

- **f.seek(*offset*, *from_what*)** changes the file object's position
  - The position is computed from adding ***offset*** to a reference point.
  - The reference point is selected by the ***from_what*** value, the supported values are:
    - 0 => the beginning of the file, it's the default if ***from_what*** *is omitted.*
    - 1 => the current file position **(+/- offsets)**.
    - 2 => the end of the file **(- offsets)**.

```
f = open('MyFile.txt', 'r+') # Suppose MyFile.txt is an empty file
f.write('0123456789abcdef')

f.seek(5)      # Go to the 6th byte in the file
f.read(1)      # '5'

f.seek(-3, 2)  # Go to the 3rd byte before the end
f.read(1)      # 'd'
```

File Operations

- Call **f.close**() to close a file and free up any system resources taken up by the open file. After calling f.close(), attempts to use the file object will automatically **fail**.

```
f.close()
f.closed  #True
f.read()
Traceback (most recent call last):
  File "<stdin>", line 1, in ?
ValueError: I/O operation on closed file
```

File Operations

- **With Statement:**
  - It is good practice to use the **with** keyword when dealing with file objects.

```
with open('workfile', 'r') as f:
  read_data = f.read()
  #do any other operations on file

f.closed #True
```

File Operations

- **General Points:**
  - Take care when dealing with **binary** files (JPG, Object files and ..) to append '**b**' to the **mode** value to open the file in binary mode.
    - Also to handle file newline endings as it is without python automatic translation you can use binary mode
  - Take care when dealing with **large files** not to read it all at once, try to read it **line by line** or **chunk by chunk**. Use os.path.**getsize**('MyFile.txt') to get file size.
  - Take care of file encodings when reading files. The "**codecs**" module provides support for reading a unicode file.

```
import codecs
f = codecs.open('foo.txt', 'rU', 'utf-8')
```

  - For more information about file operations:
    - https://docs.python.org/2.7/library/stdtypes.html#bltin-file-objects
    - https://docs.python.org/2.7/library/functions.html#open

LAB – 6
FILES LAB

Files Lab

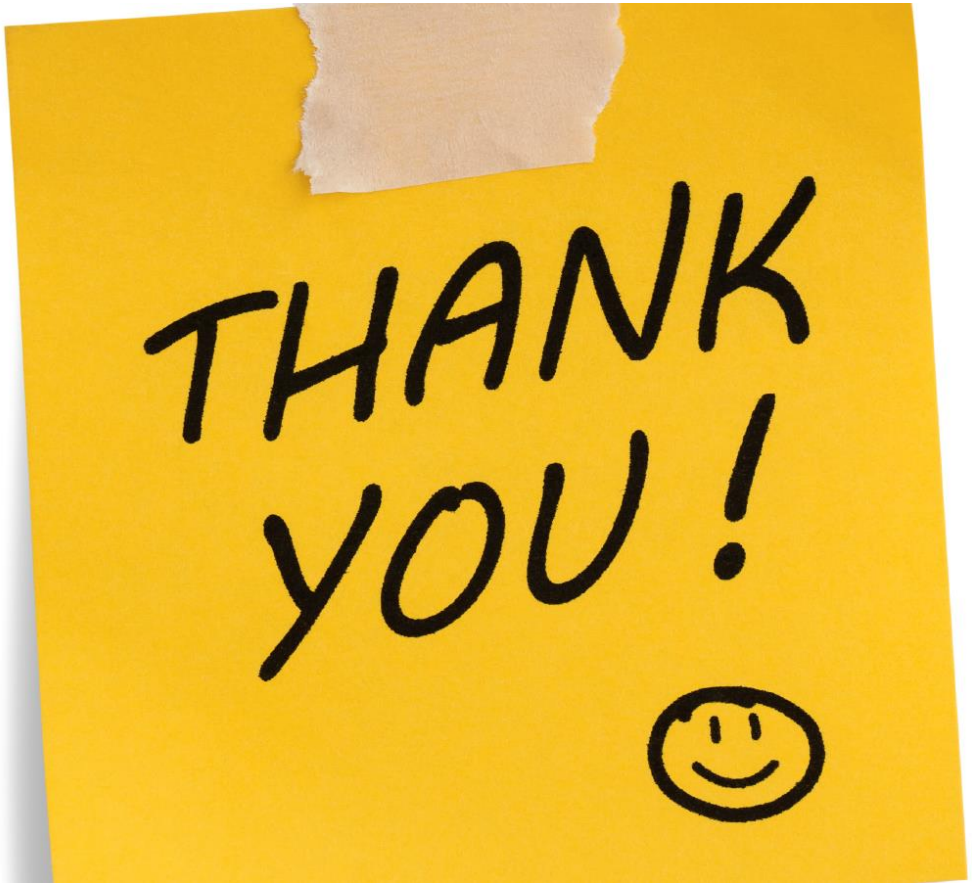
- Please download the lab from the following link:
  - https://drive.google.com/open?id=1ZBvmzVQgwSpe9wFfmPo8RevfngzH53fI
  - Complete the script **Files_lab.py** in **60** mins and send your solution on the following email:
  - Omar.Soliman@imtSchool.com with the following subject :
    - If you are from ITI-Smart track:
      - [ITI_SV_ES][PY-files]yourfullname
    - If you are from ITI-NasrCity track:
      - [ITI_NC_ES][PY-files]yourfullname

LAB – 6
FILES LAB

What's Next ?

- Get Certified With:
  - https://www.edx.org/course/learn-program-using-python-utarlingtonx-cse1309x
  - https://www.coursera.org/course/interactivepython1
  - https://www.coursera.org/course/interactivepython2

- More Interesting References:
  - Python Cookbook, 2nd Edition
  - https://automatetheboringstuff.com/
  - http://code.activestate.com/recipes/langs/



Eng. Mohammad A.Hekal: embeddedgeek.34@gmail.com
Omar Soliman: Omar.Soliman@imtSchool.com