MTA 98-381 LESSON 7 FILES

FILE OPEN MODE

- Use open() function to open a file.
- Syntax:

```
open (filename[, mode]
```

- 4 different modes for opening a file:
  - "r" – read file (Default). Error if the file does not exist.
  - "a" – append to the end of existing file. Create the file if it does not exist.
  - "w" – overwrite existing file. Creates the file if it does not exist.
  - "x" – create a new file. Error if the file already exists.

FILE OPEN MODE (CONT.)

- We can specify if the file should be handled as text or binary mode:
  - "t" - Text (Default).
  - "b" - Binary (image, video, etc.)
- We can open a file for both reading and writing:
  - "r+" – read and write a file. The file pointer is at the beginning of the file.
  - "a+" – append and read a file. The file pointer is at the end of the file.
  - "w+" – overwrite and read a file.

FILE OPEN MODE (CONT.)

- Examples:

```
open ('file.txt')         # read, text
open ('file.txt', 'rt')   # read, text
open ('file.txt', 'rb+')  # read and write, binary
open ('file.txt', 'wb+')  # write and read, binary
open ('file.txt', 'a+')   # append and read, text
open ('file.txt', 'ab')   # append, binary
```

FILE OPERATIONS

- Use `file.mode` attribute to check the open mode of an open file.
- Use `file.write(string)` function to write a string to an open file (does not automatically add a newline character ('\n') to the end of the string).
- Use `file.write(string + '\n')` to add '\n' to the end of the string.
- Use `file.read()` function to read a string from an open file.
- Use `file.close()` function to close an open file after use.

EXERCISE 1: WRITE TO A FILE

- 1. Run the following code.

```
file1 = open ("File1.txt", 'w')
print ('Opening mode:', file1.mode)
while True:
    data = input ('Enter your data: ')
    if data == '':
        break
    file1.write (data)
file1.close()
```

- 2. Enter several lines of input. Enter empty string to end.
- 3. Open file1.txt to check the file content. Your input is stored in one line although you input multiple lines.
- 4. Modify the program to store multiple lines input.

EXERCISE 2: APPEND TO A FILE

- In previous exercise, new data overwrites old data every time you run the program. Modify the program so that new data is appended to the old data.

READ FROM A FILE

1. Read all file contents into a string.

```
string = file1.read()
```

2. Read all file contents into a list.

```
lines = file1.readlines()
```

3. Read a line from file

```
line = file1.readline()
```

4. Read all file contents line by line.

```
for line in file1:
  print (line)
```

EXERCISE 3: READ FROM A FILE

- 1. Make sure your previous File1.txt exists.
- 2. Run the following code.

```
file1 = open ("File1.txt")
print ('Opening mode:', file1.mode)
string = file1.read()
print (string)
file1.close()
```

- 2. Modify the program to read the file contents as list.
- 3. Modify the program to read the file contents line-by-line.

CHECK IF A FILE EXISTS

- Use `os.path.isfile()` or `os.path.exists()`
- Example:

```
import os
if os.path.isfile ("File.txt"):
    print ("File.txt exists.")
else:
    print ("File.txt does not exist.")
```

DELETE A FILE

- Example:

```
import os
if os.path.isfile ("file.txt"):
    os.remove ("file.txt")
else:
    print ("File does not exist.")
```