Tutorial 5 Notes

String-Index Operator

```
# String-Index Operator
s = 'crocodile'
# s11 = s[9] # Error
s12 = s[6]  # s[-2] = 'l'
s13 = s[1:7]  # 'rocodi'
s14 = s[1:7:2]  # 'rcd'
s15 = s[1:-1]  # rocodil
s15 = s[1:7:len(s)]  # r
print(s15)
```

```
r
```

Manipulating String

```
# delete characters in a string
s = 'crocodile'
s21 = s[:5] + s[6:]  # if we want to delete 'd' (index 5)
s22 = s[:4] + s[-3:]  # if we want to delete 'od'
s23 = s[:3] + s[4] + s[6:]
```

```
crooile
```

```
# change an element
s = 'crocodile'
s31 = s[:6] + 'o' + s[7] + 'o'  # 'crocod' + 'o' + 'l' + 'o'
s32 = s[:5] + 'abcd' + s[7] + 'o'  # 'croco' + 'abcd' + 'l' + 'o'
ln33 = len(s) # 9
print(s32)
```

```
crocoabcdlo
```

```
s = 'crocodile'
s41 = 'cro' in s
s42 = 'crc' in s
s43 = 'cod' in s
s44 = '' in s
print(s41, s42, s43, s44)
```

```
True False True True
```

```
for charac in s:  # it iterates every character in the string
    print(charac, end=' > ')
```

```
c > r > o > c > o > d > i > l > e >
```

```
for charac in s:  # it iterates every character in the string
    if charac == s[-1]:
        print(charac, end='')
    else:
        print(charac, end=' > ')
```

```
c > r > o > c > o > d > i > l > e
```

Some Methods of Strings-1

```
# Judge the types of Characters in Strings
s = 'crocodile'
print(s.isalnum(), s.isalpha(), s.isdigit(), s.isupper(), s.islower())
```

```
True True False False True
```

```
s = 'CRO123*'
print(s.isalnum(), s.isalpha(), s.isdigit(), s.isupper(), s.islower())
```

```
False False False True False
```

```
# the endswith() and startswith() --> it is a boolean operator
s = 'crocodile'
s51 = s.endswith('le')
s52 = s.endswith('')
s53 = s.endswith('di')
print(s51, s52, s53)
s54 = s.startswith('cr')
s55 = s.startswith('')
s56 = s.startswith('crco')
print(s54, s55, s56)
```

```
True True False
True True False
```

```
# find(s1) and rfind(s1)
# find: find the LOWEST index that corresponds to that substring
# rfind: find the HIGHEST index that corresponds to that substring
s = 'crocodile'
s61 = s.find('oc')  # 2
s62 = s.find('l')  # 7
s63 = s.find('le')  # 7
s64 = s.find('ll')  # -1
s65 = s.find('c')  # 0
s66 = s.find('o')  # 2
print(s61, s62, s63, s66)
s67 = s.rfind('c')  # 3
s68 = s.rfind('o')  # 4

x = 'cococo'
```

```
x1 = x.find('co') # 4
x2 = x.rfind('co') # 0
```

```
2 7 7 2
4
```

```
# lower and upper -> modify to lowercase or uppercase letters
s = 'CRoCoDiLe90&*'
sUpper = s.upper()
sLower = s.lower()
print(sUpper, sLower)
# swapcase -> change lowercase to uppercase AND change uppercase to
lowercase
sSwap = s.swapcase()
print(sSwap)
```

```
CROCODILE90&* crocodile90&*
crOcOdIlE90&*
```

```
# count and replace
# count: count the non-overlapping occurences of the substring s1
# replace: replace the non-overlapping occurences of substring s1 with
another string
s = 'aaaaaaa'
s71 = s.count('a')
s72 = s.count('aa')
s73 = s.count('aaa')
print(s71, s72, s73)
# 7 'a', 3 'aa', 2 'aaa'. It is equivalent with 7//2 or 7//3
s74 = s.replace('a', 'rep')
s75 = s.replace('aa', 'rep')
s76 = s.replace('aaa', 'rep')
print(s74, s75, s76)  # notice the number of 'rep' in each string (see
the pattern?)
```

```
7 3 2
repreprepreprepreprep repreprepa reprepa
```

```
# split(delimiter) function
# delimiter: a character that separates a text string
# DEFAULT: split(' ')
# RESULT:  a list of strings
s = 'crocodile'
s81 = s.split('o')
s82 = s.split('l')
s83 = s.split()
print(s81, s82, s83)  # notice that the delimiter is deleted!
s2 = 'aaaaaa aaa'
s84 = s2.split()
s85 = s2.split('a')  # ['', '', '', '', '', '', ' ', '', '', '']
# ^ '(a)(a)(a)(a)(a)(a)( )(a)(a)(a)'
```

```
s86 = s2.split('aa')  # ['', '', '', ' ', 'a'] '(aa)(aa)(aa)( )(aa)
(a)'
s87 = s2.split('aaa')  # ['', '', ' ', ''] '(aaa)(aaa)( )(aaa)'
print(s84, s85, s86, s87)
```

```
['cr', 'c', 'dile'] ['crocodi', 'e'] ['crocodile']
['aaaaaa', 'aaa'] ['', '', '', '', '', '', ' ', '', '', ''] ['', '',
'', ' ', 'a'] ['', '', ' ', '']
```

File Handling

test.txt: Hello World JIAYOU_MIDTERMS\n\

note: there is another newline afte the last line 'JIAYOU_MIDTERMS\n'

Read File Functions

There are three main function that you can use to read file:

1. fileVariable.read() → a string
2. fileVariable.readline() → a string
3. fileVariable.readlines() → a list/

note: if the mode is 'w' or 'a', then these functions will output an ERROR

```
# fileVariable.read()
fileVariable = open('test.txt', 'r')
fVread = fileVariable.read()
print(fVread)  # it reads the whole content of the file into a string
```

```
Hello
World
JIAYOU_MIDTERMS\n
```

```
# Structure: fileVariable = open(filename, mode)
# \ -> escape operator : to insert character that are illegal in
string
fV = open("test.txt", 'r')
print(fV.readlines())
```

```
['Hello\n', 'World\n', 'JIAYOU_MIDTERMS\\n']
```

Pointer in a File

After you finish reading a text file to the end, the pointer will point to the end of the file.
If you want to read from the beginning again, you need to do one of the followings:

- Close and open the file again
- use fileVariable.seek(0)

```
# First method
fV.close()
fV = open('test.txt', 'r')

# second method
fV.seek(0)

0
```

FILE MODE

There are three file modes

- 'r' (read): you cannot modify the file. You just read and analyse the file
- 'w' (write): you write the file from the start! *** WARNING: If that file name already existed, the content of the old file will be deleted! ***
- 'a' (append): you edit the old version of the file by appending new sentences/strings at the end of the old file

```
fV11 = open('write1.txt', 'w')
fV11.write("Hate\n")  # \n denotes a NEW LINE
fV11.write("You")
fV11.close()
fV11 = open('write1.txt', 'r')
print(fV11.read())

Hate
You

# if we use write1.txt again, then the file will be automatically
deleted
fV21 = open('write1.txt', 'w')
fV21.write("Love\n")  # \n denotes a NEW LINE
fV21.write("You ")
fV21.close()
fV22 = open('write1.txt', 'r')
print(fV22.read())

Love
You

# if we use write1.txt again, then the file will be automatically
deleted
fV31 = open('write1.txt', 'a')
fV31.write("Forever\n")  # \n denotes a NEW LINE
fV31.write("My Love")
fV31.close()
fV32 = open('write1.txt', 'r')
print(fV32.read())
```

```
Love
You Forever
My Love

fV41 = open('write2.txt', 'a')
```

Exercises Answers

Q1 - Check Substrings

You can check whether a string is a substring of another string by using the find method in the str class. Write your own function to implement find. Write a program that prompts the user to enter two strings and then checks whether the first string is a substring of the second string.

```
def UsingFindFunction(first, second):
    if second.find(first) != -1:
        print(first + " is a substring of " + second)
    else:
        print(first + " is NOT a substring of " + second)

def selfDefinedFindFunction(first, second):
    foundSubstr = False
    for secondidx in range(len(second)):
        if second[secondidx:secondidx + len(first)] == first:
            print(first + " is a substring of " + second)
            foundSubstr = True

    if foundSubstr is False:
        print(first + " is NOT a substring of " + second)

selfDefinedFindFunction("llo", "hello")
UsingFindFunction("llo", "hello")

llo is a substring of hello
llo is a substring of hello
```

Q2 - Check Password

- A password must have at least eight characters → len()
- A password must consist of only letters and digits. → isalnum()
- A password must contain at least two digits. → isdigit()

```
def checkPassword(pwd):
    # consist of at least 8 characters and only consist of
alphanumeric
    if len(pwd) >= 8 and pwd.isalnum() is True:
        countdigit = 0
        # check if it has at least two digits
        for ch in pwd:
            if ch.isdigit():
                countdigit += 1
        if countdigit >= 2:
            return True
    return False

res = checkPassword("hello123")
if res is True:
    print("Your password is valid")
else:
    print("Your password is not valid. Please check the requirements
again")
```

```
Your password is valid
```

Q3 - Longest Common Prefix

Write a method that returns the longest common prefix of two strings. For example, the longest common prefix of distance and disinfection is dis. The header of the method is **def prefix(s1,s2)**: . If the two strings have no common prefix, the method returns an empty string. Write a main method that prompts the user to enter two strings and display their longest common prefix.

```
def prefix(s1, s2):
    leng = 1
    while True:
        if s1[:leng:1] == s2[:leng:1]:  # if s1.startswith(s2[:leng]):
<- you can also use this method
            leng += 1
        else:
            break
    return s1[:leng-1]
first = input("Enter the first string: ")
second = input("Enter the second string: ")
print("The longest common prefix of the two strings is", prefix(first,
second))
```

```
The longest common prefix of the two strings is dis
```

Q4.i) - Handling File

Write a program that will count the number of characters, words, and lines in a file. Words are separated by a whitespace character. Your program should prompt the user to enter a filename.

```
def fileHandling(fileName):
    try:
        fileVariable = open(fileName, 'r')  # we set the mode to be
'r' since we don't want to edit the file
        fileContent = fileVariable.read()
        numOfCharacters = len(fileContent)
        print("In the file "+ fileName + ', there are')
        print(numOfCharacters, 'characters,')
        fileVariable.close()
        fV = open(fileName, 'r')  # we need to close and open the file
again to make the pointer point to the beginning again
        numOfWords = 0
        for line in fV.readlines():
            numOfWords = numOfWords + len(line.split())
        print(numOfWords, 'words,')
        fV.close()
        fV2 = open(fileName, 'r')  # we need to close and open the
file again to make the pointer point to the beginning again
        numOfLines = len(fV2.readlines())
        print(numOfLines, 'lines.')
    except:
        print("the file name you inputted doesn't exist")

#fileName = input("Enter a filename:")
fileName = 'JaneEyre_utf8.txt'
fileHandling(fileName)
```

```
In the file JaneEyre_utf8.txt, there are
10893 characters,
1930 words,
46 lines.
```

Q4.ii) - Handling File

In the file “JaneEyre.txt”, there are some typos “amd” which should actually be “and”, find how many of them and replace all, then write the correct version into a new file.

```
def fileHandling2(fileName):
    # find the number of typo 'amd's
    fV = open(fileName, 'r')
    fileContent = fV.read()
    numberOfTypos = 0
    index = 0
    while index != len(fileContent):
        # we only want the word 'amd' and not 'amd' that is just part
of a word
        # for example, 'slAMDunks
        if fileContent[index: index+3] == 'amd' and fileContent[index-
1].isalnum() is False and fileContent[index+3].isalnum() is False:
            # change it into 'and'
            fileContent = fileContent[:index] + 'and' +
fileContent[index+3:]
            numberOfTypos += 1
            index += 4
        index+=1
    print("There are", numberOfTypos, "'amd' that was changed to and")
    # rewrite the revised text to a newly created file
    fV2 = open("JaneEyreRevised.txt", 'w')
    fV2.write(fileContent)

fileHandling2('JaneEyre_utf8.txt')
```

```
There are 80 'amd' that was changed to and
```