

Data Processing

Section 1
Chapter 5

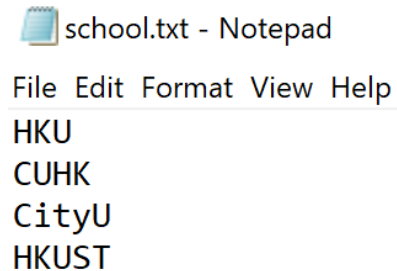
Quiz 11

Motivation

- Empirical and textual analyses often rely on data that are typically stored in **files**.
 - Need to connect Python programs with the files.
- So far, the output of our programs has been displayed on the screen and eventually lost. In real-life applications, we want the output to be preserved in **files** in permanent storage that will be available for later use.

Text Files

- Text files are simple files consisting of lines of text with no formatting
 - Text files can be created with any word processor, and accessed by Python programs



school.txt - Notepad

File Edit Format View Help

HKU
CUHK
CityU
HKUST

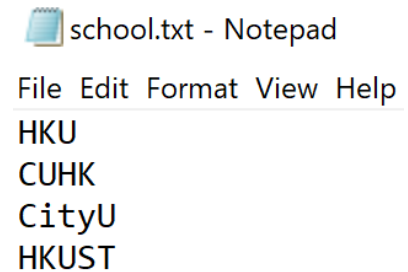
Connecting and disconnecting with text Files

- Establish connection between the program and the file
 - File is said to be opened for input

```
infile = open("school.txt", 'r')
```

- At any time, the connection from the program to the file can be terminated with the following statement

```
infile.close()
```



Some errors in opening files


```
>>> infile = open("school.txt", 'r') # correct
>>>
>>>
>>> infile = open("school", 'r') # Did not specify the file type
Traceback (most recent call last):
  File "<pyshell#4>", line 1, in <module>
    infile = open("school", 'r') # Did not specify the file type
FileNotFoundError: [Errno 2] No such file or directory: 'school'
>>>
>>>
>>> infile = open(school.txt, 'r') # Missing "" on the file name.
Traceback (most recent call last):
  File "<pyshell#7>", line 1, in <module>
    infile = open(school.txt, 'r') # Missing "" on the file name.
NameError: name 'school' is not defined
>>>
>>>
>>> infile = open("school.txt", r) # Missing ' on r.
Traceback (most recent call last):
  File "<pyshell#10>", line 1, in <module>
    infile = open("school.txt", r) # Missing ' on r.
NameError: name 'r' is not defined
```

Reading text files: the read method

- A file that is open can be accessed with the **read** and **readline** methods.
- The read method places the entire contents of the file (including the newline characters) into a single string.

```
strVar = infile.read()
```

Example 1: read

 school.txt - Notepad

File Edit Format View Help

HKU
CUHK
CityU
HKUST

```
#The entire txt file is read into a string.  
infile = open("school.txt", 'r')  
string = infile.read()  
print(string)  
infile.close()
```

HKU
CUHK
CityU
HKUST

```
>>> string[0]  
'H'  
>>> string[1]  
'K'  
>>> string[2]  
'U'  
>>> string[3] #note it is a change of line character.  
'\n'
```



Reading text files: the **readline** method

- When a text file is opened for input, a pointer is set to the beginning of the first line of the file.
- Each time a readline method is executed, the current line can be assigned to a string variable and the pointer advances to end of that line.

```
strVar = infile.readline()
```

- The readline method returns the empty string after all lines have been read.

Example 2: readline

 school.txt - Notepad

File Edit Format View Help

HKU
CUHK
CityU
HKUST

```
#Read each line into a string
infile = open("school.txt", 'r')
line = infile.readline()
while line != "":
    print(line, end="")
    line = infile.readline()
infile.close()
```

HKU
CUHK
CityU
HKUST

File handling using lists


- Each line of a text file can be read into a list using a for loop.

```
infile = open("fileName.txt", 'r')
for line in infile:
    indented block of statements
infile.close()
```

- Or simply using a list comprehension

```
listVar = [line.rstrip() for line in infile]
```

Example 3A: file handling using a list

 school.txt - Notepad


File Edit Format View Help

HKU
CUHK
CityU
HKUST

```
#Gather every line into a list.  
infile = open("school.txt", 'r')  
lst = list(infile)  # Or use lst = [line for line in infile]  
print(lst)  
infile.close()
```

```
['HKU\n', 'CUHK\n', 'CityU\n', 'HKUST\n']
```

Example 3B: file handling using a list

 school.txt - Notepad

File Edit Format View Help

HKU
CUHK
CityU
HKUST

```
#Gather every line into a list while getting rid of \n for each line.  
infile = open("school.txt", 'r')  
lst = [line.rstrip() for line in infile]  
print(lst)  
infile.close()
```

```
['HKU', 'CUHK', 'CityU', 'HKUST']
```

Creating Text Files

- Open with 'w' creates a new text file with the specified name. The file is opened for writing.

```
outfile = open(fileName, 'w')
```

- If list1 is a list of strings (each ending with “\n”), then the writelines method writes each item of the list into a file.

```
outfile.writelines(list1)
```


- Write adds the value of *strVar* as a line of the file

```
outfile.write(strVar)
```

- **Files must be closed** to guarantee that all data has been physically transferred to the disk.

Example 4: write

```
#Write a list into a txt file item by item.  
lst = ["CityU", "CUHK", "HKU", "HKUST"]  
outfile = open("school_1.txt", 'w')  
for x in lst:  
    outfile.write(x+"\n") # \n change line  
outfile.close()
```

 school_1.txt - Notepad

File Edit Format View Help

|CityU

CUHK

HKU

HKUST

Example 5: writelines

```
lst = ["CityU", "CUHK", "HKU", "HKUST"]
lst_n = [x+"\n" for x in lst]    #Create a list with \n added for each item.
outfile = open("school_1.txt", 'w')
outfile.writelines(lst_n)
outfile.close()
```


Example 6

- Sort the school names in the text file in the alphabetical order.

- Strategy
 - Put data in a list.
 - Sort the list.
 - Create a new file based on the list.

school.txt - Notepad
File Edit Format View Help
HKU
CUHK
CityU
HKUST




school_2.txt - Notepad
File Edit Format View Help
CUHK
CityU
HKU
HKUST

Adding lines to a text file

```
outfile = open(fileName, 'a')
```

- The above statement allows the program to add lines to the end of the specified file.
- Then the writelines and write methods can be used to add new lines. The file is said to be opened for append.


Example 7: Adding lines to a text file

 school.txt - Notepad

File Edit Format View Help

HKU
CUHK
CityU
HKUST

```
outfile = open("school.txt", 'a')  
lst = ["PolyU\n", "Lingnan\n"]  
outfile.writelines(lst) #add a list (two lines)  
outfile.write("OpenU\n") #add a string (one line)  
outfile.close()
```

 school.txt - Notepad

File Edit Format View Help

HKU
CUHK
CityU
HKUST
PolyU
Lingnan
OpenU

Sets

- A set is an **un**ordered collection of items
 - **No duplicates**
 - Delimited with {}
- Note that a list can have repetitive items, but sets cannot. Unlike lists, items in sets cannot be subscripted.

```
>>> [1, 1, 2]
[1, 1, 2]
>>> {1, 1, 2}
{1, 2}
```

```
>>> s = {2, 4, 'a'}
>>> s[0]
Traceback (most recent call last):
  File "<pyshell#7>", line 1, in <module>
    s[0]
TypeError: 'set' object is not subscriptable
```

Sets

```
>>> s = {2, 3}
>>> s.add(4)
>>> s
{2, 3, 4}
>>>
>>> s = {2, 3}
>>> s.add(2)
>>> s
{2, 3}
```

```
>>> s = {2, 3}
>>> s.discard(3)
>>> s
{2}
```

```
>>> set([2, 5, 5])
{2, 5}
>>> set((2, 5, 5)) #tuple -> set
{2, 5}
>>> set("ABBA")
{'B', 'A'}
```

Set Comprehension

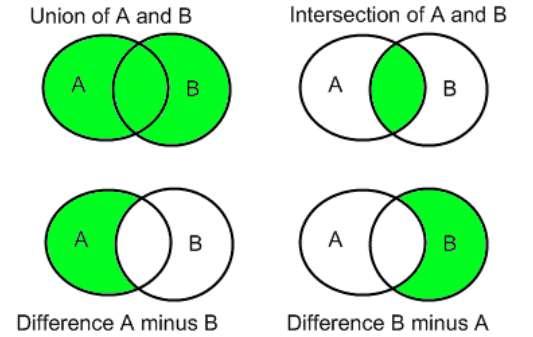
- Sets can be created with *comprehensions*.

```
>>> {x**2 for x in range(-3, 3)}  
{0, 9, 4, 1}
```

```
>>> {x**2 for x in [0, 1, 2, 3]}  
{0, 1, 4, 9}
```

Set-theoretic Methods

- Methods that create new sets from two existing sets



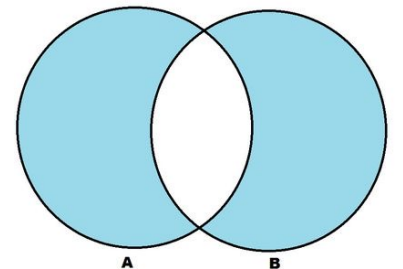
- Merge two sets

```
>>> {1, 2}.union({1, 2, 3})
{1, 2, 3}
```
- Intersection of two sets

```
>>> {1, 2, 5}.intersection({2, 3, 5})
{2, 5}
```
- Set difference

```
>>> {2, 3} - {2, 4, 5}
{3}
>>> {2, 4, 5} - {2, 3}
{4, 5}

>>> {2, 3}.symmetric_difference({2, 4, 5})
{3, 4, 5}
```




Using Set-theoretic Methods with Files

Steps for extracting information from two related text files


1. Create two sets; each containing the contents of one of the two text files.
2. Apply a set operation such as union, intersection, or difference to the sets.
3. Write the resulting set into a new text file.

Example 8: processing multiple files

 school.txt - Notepad

File Edit Format View Help

HKU
CUHK
CityU
HKUST


 school_others.txt - Notepad

File Edit Format View Help

HKU
CUHK
PolyU
Lingnan


```
# Put data from txt into sets.
infile = open("school.txt", 'r')
set1 = set(infile)
infile.close()
infile = open("school_others.txt", 'r')
set2 = set(infile)
infile.close()
```

```
# Set operations and save in txt.
outfile = open("union.txt", 'w')
outfile.writelines(set1.union(set2))
outfile.close()
outfile = open("intersection.txt", 'w')
outfile.writelines(set1.intersection(set2))
outfile.close()
outfile = open("difference.txt", 'w')
outfile.writelines(set1 - set2)
outfile.close()
```

 union.txt - Notepad


File Edit Format View Help

HKU
PolyU
CityU
Lingnan
CUHK
HKUST

 intersection.txt - Notepad

File Edit Format View Help

HKU
CUHK

 difference.txt - Notepad

File Edit Format View Help


CityU
HKUST

Processing Data, Part 2

Section 2
Chapter 5

CSV Files

- Text files we considered so far have a single piece of data per line (one column)
- Consider CSV formatted file
 - Several items of data on each line
 - Items are separated by commas.

 school_student.txt - Notepad

File Edit Format View Help

HKU,29791,8266


CUHK,20608,1697

CityU,18582,845

HKUST,16054,697

Example 9: average of a column

- The second column are student populations in each school. Find its average among the four schools.

 school_student.txt - Notepad

File Edit Format View Help

HKU,29791,8266

CUHK,20608,1697


CityU,18582,845

HKUST,16054,697

Average student population: 21259

Example 10: calculation for each row

- The third column is the number of faculties in each school. Find the student/faculty ratio for each school.

 school_student.txt - Notepad

File Edit Format View Help


HKU,29791,8266

CUHK,20608,1697

CityU,18582,845

HKUST,16054,697



 school_student_new.txt - Notepad

File Edit Format View Help

HKU,29791,8266,4

CUHK,20608,1697,12

CityU,18582,845,22

HKUST,16054,697,23

Classwork 11. Acrostic Poem

- Create a function `first_letters` that returns the first letter of each line in any given text file. The function should be capable of the following. The sample text file is available for download on Canvas. Upload the .py file and the output screenshot on Canvas.

```
>>> first_letters("acrostic.txt")
```

```
'CATS'
```



acrostic - Notepad

File Edit Format View Help

Cuddly and fluffy

Always cheeky

They make us laugh

Superb at climbing