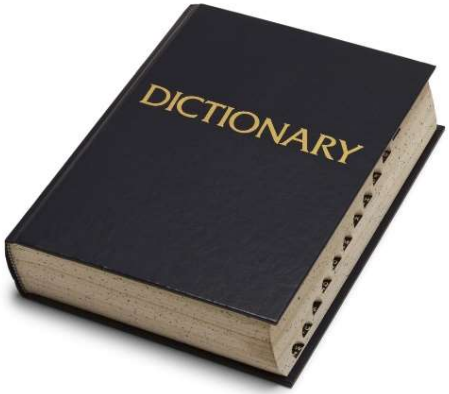
5/22 字典、檔案 (Dict and Files)



http://poet.ncnu.org/

Review

- String
  - Slicing
- Lists
  - Indexing
- Dictionary
  - Key-value pairs
- File
  - Multi-line strings stored in your storage device

File I/O

- Everything is gone after you turn off your computer.
- You would certainly like to save some data/programs for later use.
  - The cost of storage is extremely low.
  - 1GB costs less than NT$1.
    - ☞唐詩三百首 84KB
    - ☞四書（大學、中庸、論語、孟子）244KB
    - ☞五經（詩、書、易、禮、春秋）1394KB
- A *file* is a sequence of data that is stored in secondary memory (disk drive).
- Files can contain any data type, but the easiest to work with are text.
- A file usually contains more than one line of text.
- Python uses the standard newline character (\n) to mark line breaks.

Multi-Line Strings

- `Hello`

  `World`

- When stored in a file:
  `Hello\n\nWorld\n`
  - `\n` denotes a “newline” character.
  - The file size is 13 bytes.

Multi-Line Strings

- This is exactly the same thing as embedding \n in print statements.
  - Try the following statement
    print("Hello\n\nWorld\n")

- Remember, these special characters ('\n', '\t') only affect things when printed.

File Processing - Open

- The process of *opening* a file involves associating a file on disk with an object in memory.
- We can manipulate the file by manipulating this object.
  - Read from the file
  - Write to the file

File Processing - Close

- When done with the file, it needs to be *closed*. Closing the file guarantees any pending operations and other bookkeeping for the file to be completed.
- In some cases, failing to properly close a file could result in data loss.

File Processing - Write

- Opening a file for writing prepares the file to receive data
- If you open an existing file for writing, you wipe out the file's contents. If the named file does not exist, a new one is created.

- `f = open("mydata.out", "w")`
- `print(<expressions>, file=f)`

Showing a Poem on Screen

- `print("千山鳥飛絕，")`
- `print("萬徑人蹤滅；")`
- `print("孤舟蓑笠翁，")`
- `print("獨釣寒江雪。")`

Writing a Poem into a File

```
f = open("poem01.txt", "w")
print("千山鳥飛絕，", file=f)
print("萬徑人蹤滅；", file=f)
print("孤舟蓑笠翁，", file=f)
print("獨釣寒江雪。", file=f)
f.close()
```

- Listing files in current working directory:
  - `import glob`
  - `print( glob.glob("*") )`

File Processing - Read

- `<file>.read()` – returns the entire remaining contents of the file as a single (possibly large, multi-line) string
- `<file>.readline()` – returns the next line of the file. This is all text up to and including the next newline character
- `<file>.readlines()` – returns a list of the remaining lines in the file. Each list item is a single line including the newline characters.

read() vs. readlines()

- os.chdir('/content/gdrive/MyDrive/KGHS')
- infile = open("poem01.txt", "r")
- s = infile.read()
- infile.close()
- print(repr(s))
- print(s)

- infile = open("poem01.txt", "r")
- aList = infile.readlines()
- infile.close()
- print(aList)

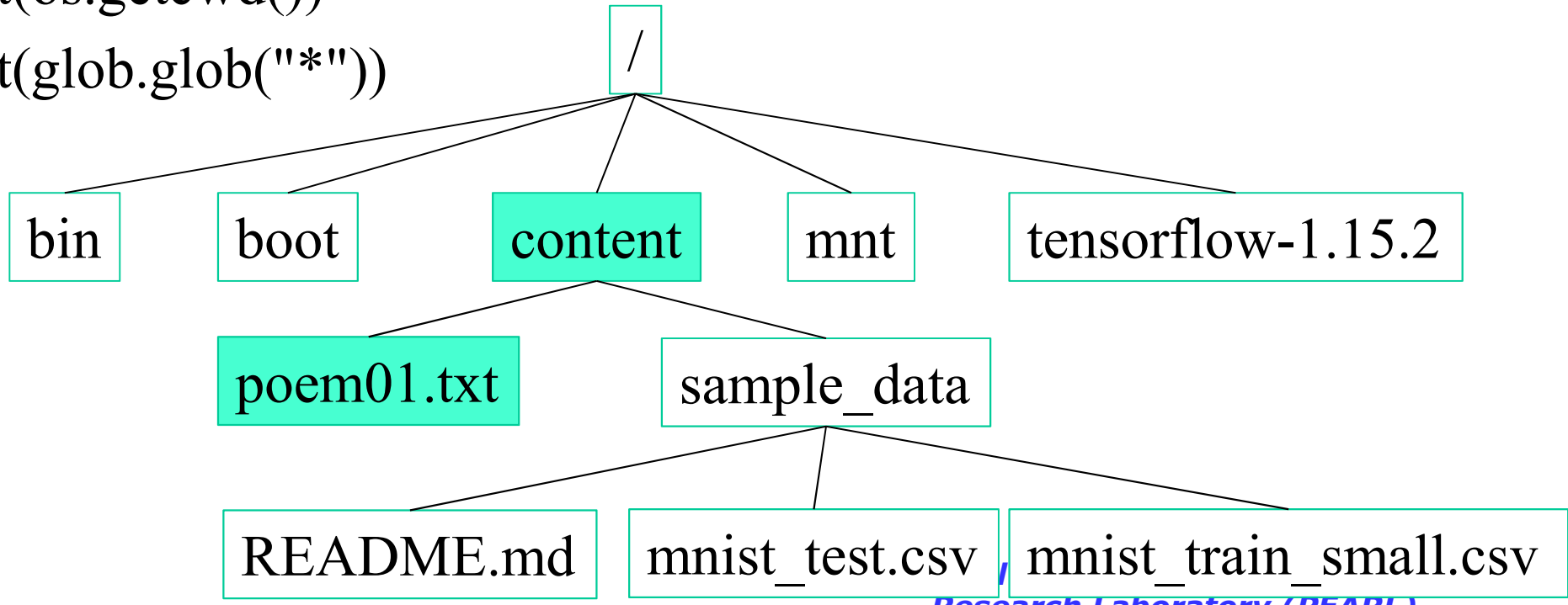
Iterate through a File

- Python treats the file itself as a sequence of lines!
-
```
infile = open(someFile, "r")
for line in infile:
    # process the line here
infile.close()
```

Current Working Directory

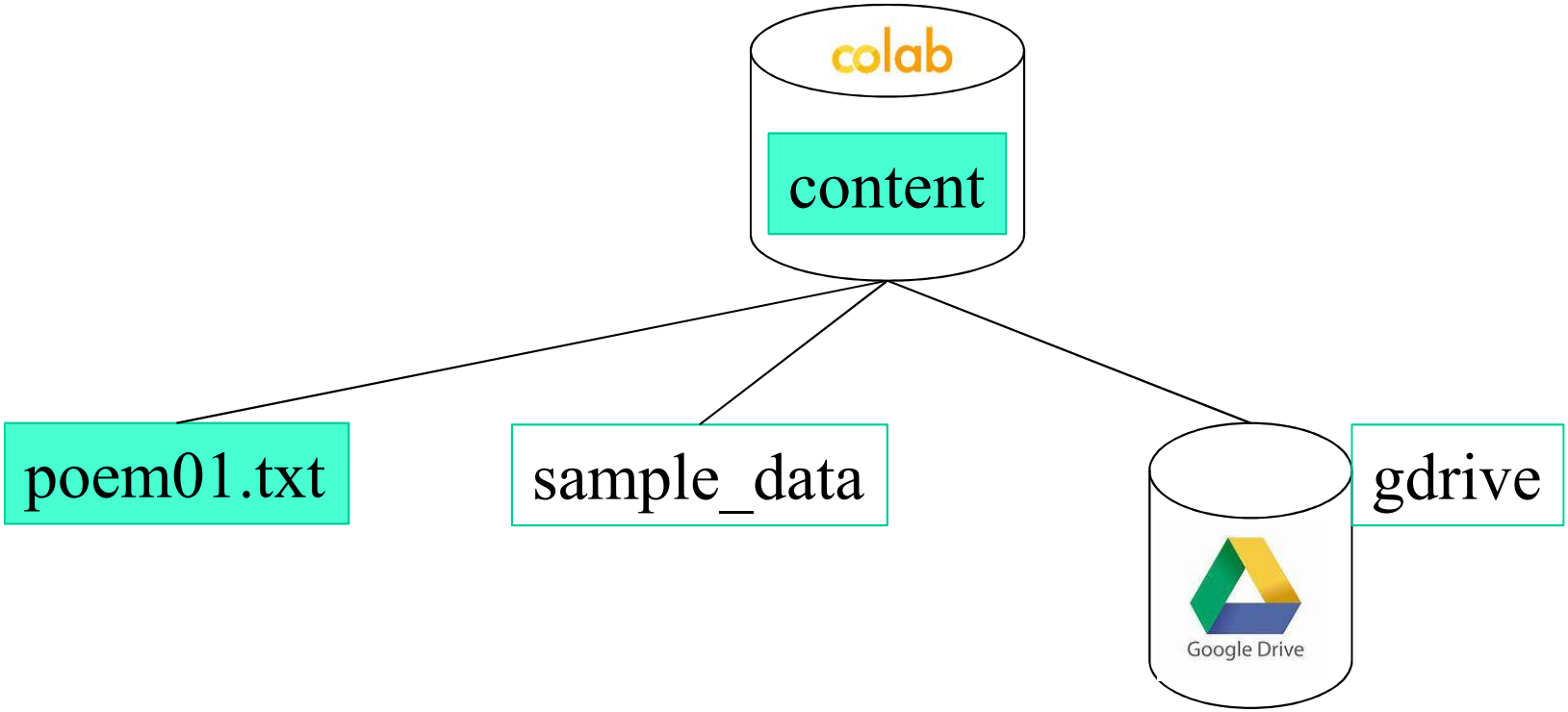
- import os, glob
- os.getcwd() # current working directory

- os.chdir('/') # Change Directory
- print(os.getcwd())
- print(glob.glob("*"))

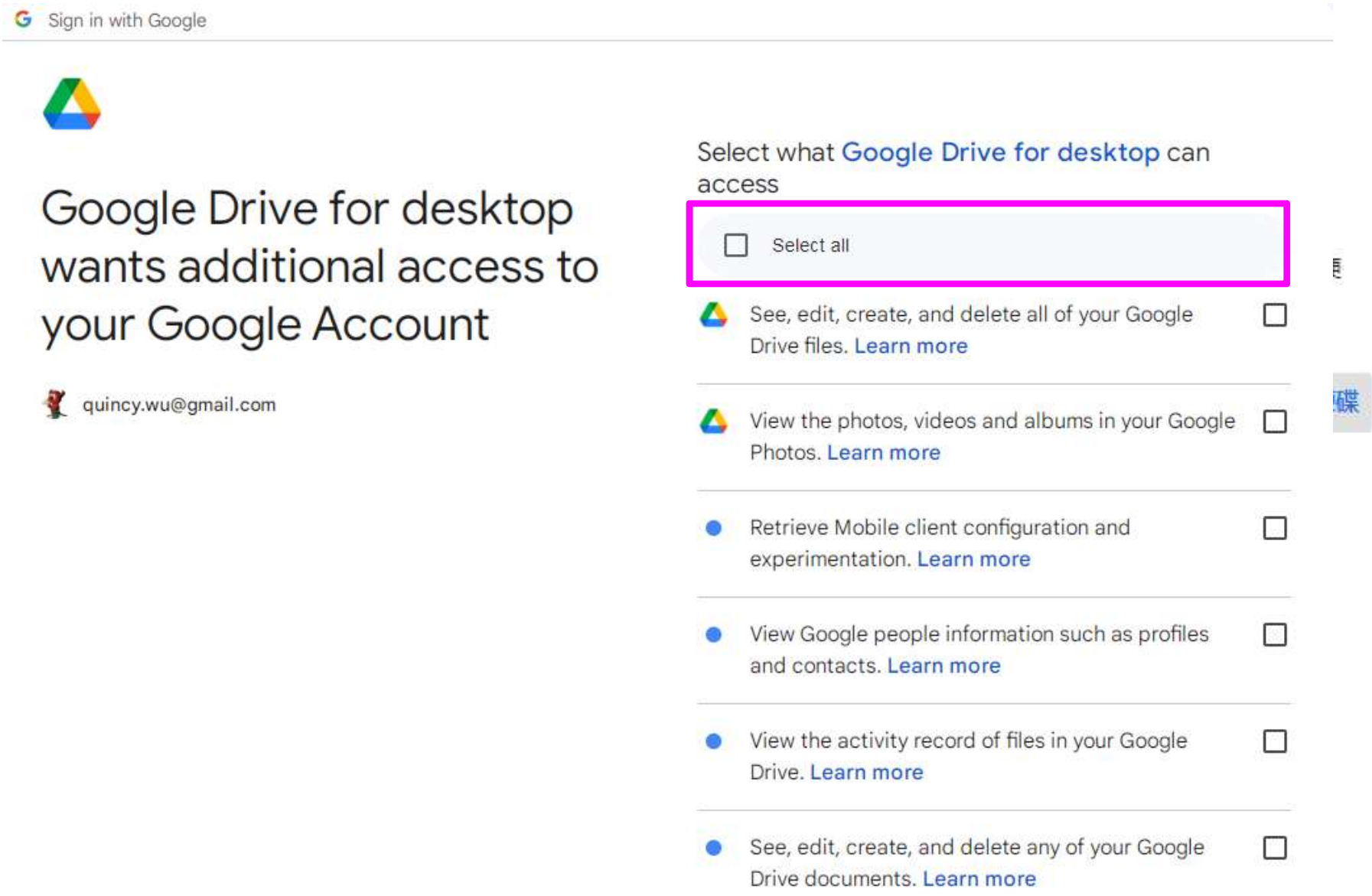


Mount gDrive

- 掛載網路硬碟
  - from google.colab import drive
  - drive.mount('/content/gdrive')



允許存取你的雲端碟硬

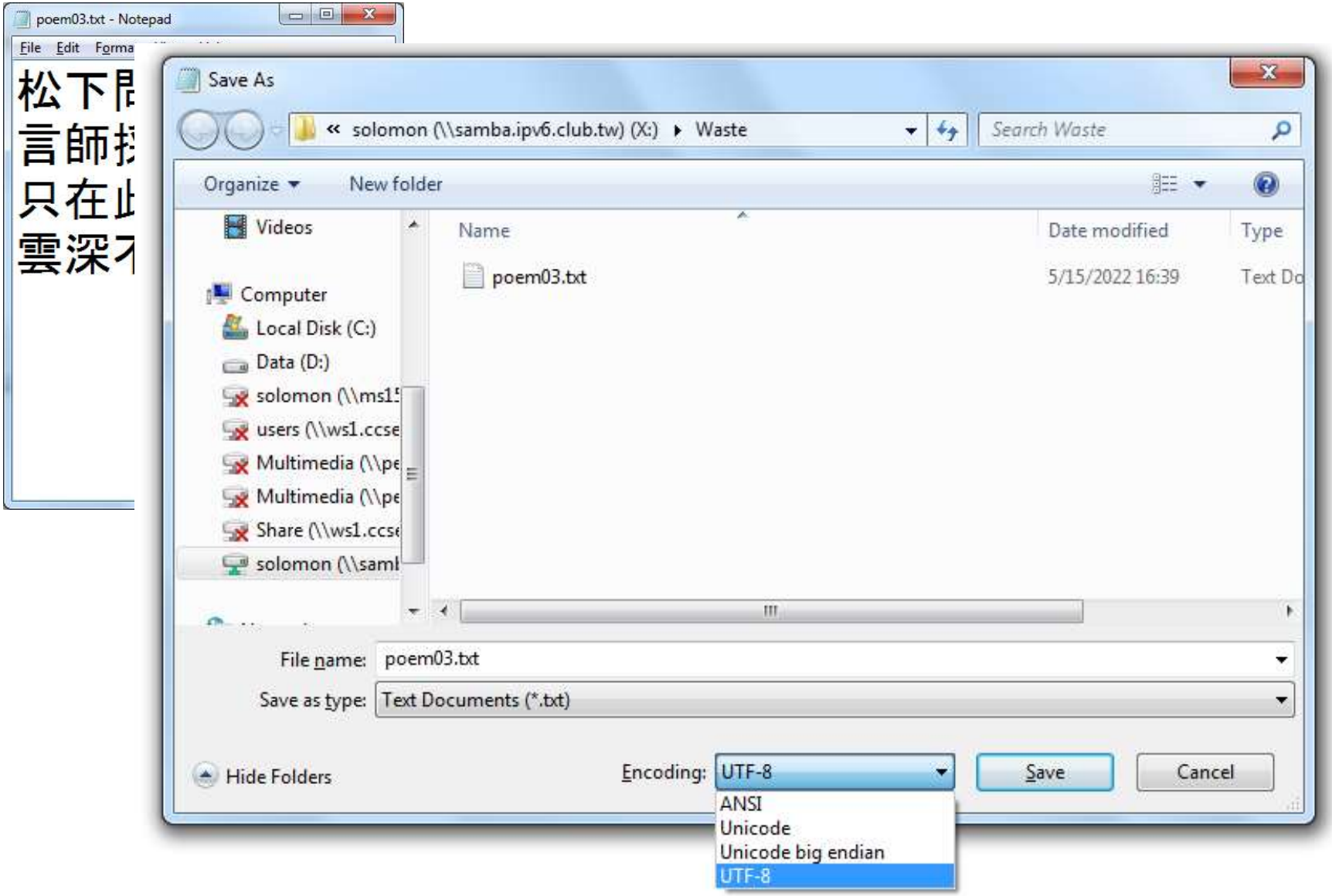

Sign in with Google
Google Drive for desktop wants additional access to your Google Account
quincy.wu@gmail.com
Select what Google Drive for desktop can access
Select all
See, edit, create, and delete all of your Google Drive files. Learn more
View the photos, videos and albums in your Google Photos. Learn more
Retrieve Mobile client configuration and experimentation. Learn more
View Google people information such as profiles and contacts. Learn more
View the activity record of files in your Google Drive. Learn more
See, edit, create, and delete any of your Google Drive documents. Learn more

Create/Remove Files in Google Drive

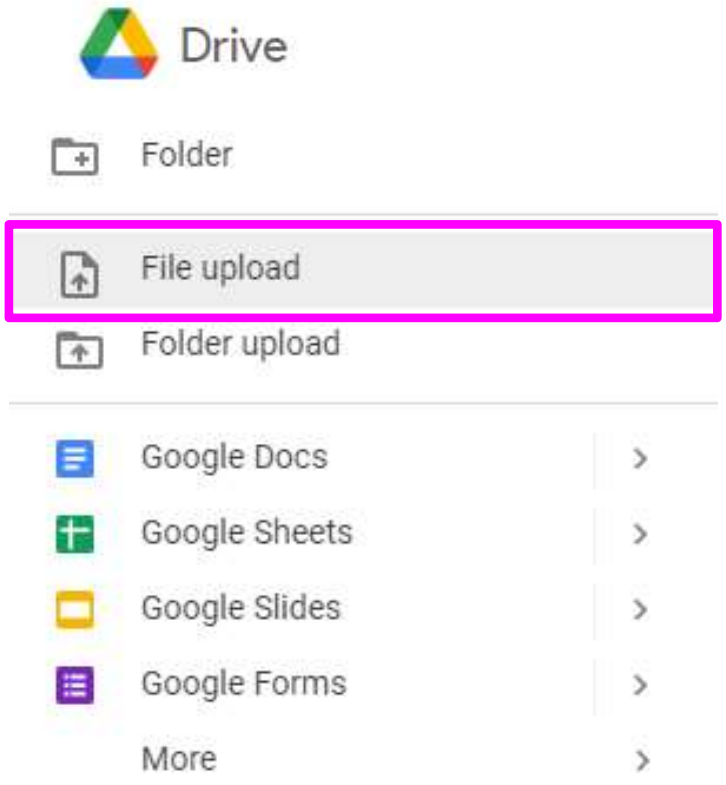
```
os.chdir('/content')
fn = 'gdrive/MyDrive/KGHS/poem02.txt'
s = '白日依山盡，\n黃河入海流；\n欲窮千里目，\n更上一層樓。\n'
outfile = open(fn, "w")
print(s, file=outfile, end='')
outfile.close()
```

- print(glob.glob('/content/gdrive/MyDrive/KGHS/poem*'))
- os.remove(fn)
- drive.flush_and_unmount()

Create a Plain Text File with Notepad



Upload Files to Google Drive, then Access from Colab




```
fn = 'gdrive/MyDrive/poem03.txt'
infile = open(fn)
s = infile.read()
print(s)
infile.close()
```

Exercise: Select a Poem

- Suppose you have 3 files in your current working directory:
  - poem01.txt 江雪
  - poem02.txt 登鸛鵲樓
  - poem03.txt 尋隱者不遇
- Write a program to prompt the user to select a number, and the corresponding poem will be printed.

```
0. Exit
1. 江雪
2. 登鸛鵲樓
3. 尋隱者不遇
Your choice --
```

Exercise: Select a Poem

- Suppose you have 3 files in your current working directory:
  - poem01.txt 江雪
  - poem02.txt 登鸛鵲樓
  - poem03.txt 尋隱者不遇
- Write a program to prompt the user to select a number, and the corresponding poem will be printed.

```
0. Exit
1. 江雪
2. 登鸛鵲樓
3. 尋隱者不遇
Your choice -- 3
松下問童子，
言師採藥去。
只在此山中，
雲深不知處。

0. Exit
1. 江雪
2. 登鸛鵲樓
3. 尋隱者不遇
Your choice --
```

Exercise: 隨機點名

- 之前旻勳老師是上網隨便找個亂數產生器，搖出一個數字（例如說12）後，還需要問「12號是誰？」
- 我們可以準備好一個文字檔，每行就是一位同學的學號和姓名。
- 把這檔案上傳到 Google Drive。
- 從 Colab 中把它用 readlines() 讀出來，存到一個 list 中。這個 list 中每個元素恰好就是一位同學的學號及姓名。
  - 把這個 list 命名為 students 好了。Python的習慣，list 的名字以 s 結束。
- 假設 n = len(students)，那麼 random.randrange(n) 就會得到 0..n-1 之間的一個隨機亂數。
  - 作用和 random.randint(0, n-1) 相同。
- 把對應於這個 index 的元素印出來吧！

Homework: Find 10+ Poems

- Suppose the filenames of poems are “poem01.txt”, “poem02.txt”, etc.
- The file contents consists of
  - Title (first line)
  - Author (second line)
  - (a blank line)
  - Poem

江雪
柳宗元

千山鳥飛絕，
萬徑人蹤滅；
孤舟蓑笠翁，
獨釣寒江雪。

- Your program should be able to automatically extract the title of each poem, and show a list for the user to select.

Question: Unknown Number of Items

- For two dices, we know the sum must be between 2 and 12, so there are only totally 11 possible values.
  - count = [0] * 13 # Index 0, 1 are wasted.
- For English characters, there are only 26 alphabets, so a list of size 26 is sufficient.
  - count = [0] * 26
- Most of the time we don't know the exact number of possible values.
  - For example, the hostname of email addresses.
    - ☞ gmail.com
    - ☞ hotmail.com
    - ☞ nsysu.edu.tw
    - ☞ mail.ncnu.edu.tw
    - ☞ ms38.hinet.net

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

Apple	1

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

Apple	1
Banana	1

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

Apple	1
Banana	2

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

Apple	2
Banana	2

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

Apple	2
Banana	2
Coconut	1

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

Apple	3
Banana	2
Coconut	1

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

Apple	3
Banana	2
Coconut	1
Guava	1

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Skill: Increase the size of your list as needed.

```
s = '''Apple
Banana
Banana
Apple
Coconut
Apple
Guava'''
```

```
fruit = []
count = []
for f in s.split():
    if f in fruit:
        i = fruit.index(f)
        count[i] = count[i] + 1
    else:
        fruit.append(f)
        count.append(1)

for i in range(len(fruit)):
    print(fruit[i], count[i])
```

Problem is solved, but we use two lists:

- fruit - name
- count - quantity

Can we combine these two related data in a single data structure?

Dictionary

- Python allows us to look up information associated with arbitrary keys.
  - In programming terminology, a *key-value pair*
  - C++ calls this *mapping*
  - Some other programming languages called *hashes* or *associative arrays*.
  - Python calls this *dictionary*.

dict

```
passwd = { "alice" : "11", "bob" : "12" }
print( passwd["alice"] )
passwd["carol"] = 13
```

Basically, dict is similar to list, since it can store lots of data.
The difference is that lists are indexed by integers,
while dicts are indexed by any data (often with strings).

```
print( passwd.get("daniel", 0) )
```

Code: Buying Fruit

- Suppose your classmates want to “group buying” fruit.
- You don’t know how many kind of choices in advance.
- You may want to create a list of fruit using the first character as the index, but unfortunately two fruits begin with ‘A’.
  - Apple
  - Apricot

- Now a dict which can be indexed by strings (compared with a list indexed by integers) is handy.

Sample Code:

```
print("Input fruit and amount.  Press Enter to finish.")
count = {}
while True:
    line = input()
    if line == '': break
    fruit, amount = line.split()
    amount = int(amount)
    if fruit in count:
        count[fruit] = count[fruit] + amount
    else:
        count[fruit] = amount

for key in count:
    print(key, count[key])
```

```
durian 5
apple 5
durian 17
apple 12
cherry 16
apple 16
apple 13
durian 6
apricot 15
apple 6
cherry 5
apricot 3
apple 14
apricot 17
durian 13
cherry 17
banana 11
cherry 5
apple 15
durian 2
```

Homework: 注音

- 將注音對照表上傳到你的 Google Drive。
  - 馮 ㄈㄥˊ,ㄆㄧㄥˊ
  - 陳 ㄔㄣˊ,ㄓㄣˋ
  - 晴 ㄑㄧㄥˊ
- 請撰寫一程式，將注音對照表讀入。
- 接著進入一個無窮迴圈，請使用者輸入姓名，接著印出其對應之注音。
- 並請使用者輸入下一個名字。

Name? 林佳佳
林(ㄌㄧㄣˊ) 佳(ㄐㄧㄚ) 佳(ㄐㄧㄚ)
Name? 馮佑熹
馮(ㄈㄥˊ) 佑(ㄧㄡˋ) 熹(不會唸)
Name?

Name? 林佳佳
林佳佳 - (ㄌㄧㄣˊ ㄐㄧㄚ ㄐㄧㄚ)
Name?

Homework: 英翻中

```
def main():
    chinese = { 'they': '他們', 'this': '這', 'that': '那',
              'it': '它',
              'is': '是', 'are': '是',
              'my': '我的', 'your': '你的',
              'brothers': '兄弟', 'fault': '錯'
            }
    sentence = input("Sentence? ")
    for c in '.,?!':
        sentence = sentence.replace(c, ' ')
    for word in sentence.lower().split():
        print(chinese[word], end='')
    print()
```

Methods for Dictionaries

Method	Meaning
<key> in <dict>	Returns true if dictionary contains the specified key, false if it doesn't.
<dict>.keys()	Return a sequence keys.
<dict>.values()	Returns a sequence of values.
<dict>.items()	Returns a sequence of tuples (key, value) representing the key-value pairs.
<dict>.get(<key>, <default>)	If dictionary has key returns its value; otherwise returns default.
Del <dict>[<key>]	Deletes the specified entry.
<dict>.clear()	Deletes all entries
for <var> in <dict>:	Loop over the keys

Getting Items in a Dictionary

- count.keys()
  - dict_keys(['durian', 'apple', 'cherry', 'apricot', 'banana'])
- count.values()
  - dict_values([43, 81, 43, 35, 11])
- count.items()
  - dict_items([('durian', 43), ('apple', 81), ('cherry', 43), ('apricot', 35), ('banana', 11)])

Sorting a List

- aList = list(count.items())
- aList.sort()
  - [('apple', 81), ('apricot', 35), ('banana', 11), ('cherry', 43), ('durian', 43)]
- aList.sort(reverse=True)
  - [('durian', 43), ('cherry', 43), ('banana', 11), ('apricot', 35), ('apple', 81)]

Can we sort a list by the second member of each element?

Define the Key for Sorting

- def f(x):
- return x[1]
- aList.sort(key=f)
  - [('banana', 11), ('apricot', 35), ('cherry', 43), ('durian', 43), ('apple', 81)]
- aList.sort(key=f, reverse=True)
  - [('apple', 81), ('cherry', 43), ('durian', 43), ('apricot', 35), ('banana', 11)]

Exercise: 琵琶行

- 統計白居易的《琵琶行》616個中，每個字出現的次數，並印出次數在5次以上的字。
  - 所謂「以上」「以下」者，俱連本數從之。

Homework: 琵琶行（二）

- 將《琵琶行》中，出現次數最多的16個字印出。
  - 依出現次數，由多到少排序。

全班名字誰最多

- 將全班姓名讀出。刪去學號的部份。
- 統計出哪些字最常被使用。
- 列出次數大於一的字，按次序由大排到小。
  - 王 3
  - 婷 3
  - 吳 3
  - 黃 3
  - 宇 3
  - 宥 2
  - 蓁 2
  - .
  - .

文字雲(Word Cloud)與關鍵字分析

- 和魯迅這個作者相關的詞
- 風雨都代表不如意，柳樹都代表離別, 流水代表消逝

Word Cloud

- from wordcloud import WordCloud
- text = 'a apple a day, keep doctors away'
- cloud = WordCloud().generate(text)
- fn = 'gdrive/MyDrive/n.png'
- cloud.to_file(fn)

Draw it with matplotlib

```
from wordcloud import WordCloud
text = "a apple a day keep docter away"
wordcloud = WordCloud(background_color='white').generate(text)

import matplotlib.pyplot as plt
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
# Shown in Colab
```

Word Cloud with Chinese

```
from wordcloud import WordCloud
text = '羌笛 何須 怨 楊柳 春風 不度 玉門關'
cloud = WordCloud(font_path='gdrive/MyDrive/kaiu.ttf').generate(text)

import matplotlib.pyplot as plt
plt.imshow(cloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

By default, WordCloud does not support Chinese fonts.
You may upload C:\Windows\Fonts\kaiu.ttf to your Google Drive and request WordCloud to use this font.

文字雲

- 將全班名字以「字」為單位，產生一文字雲。
- 注意：提供給 WordCloud 的必須為一字串而非 list。中間以空白隔開。