AI詩人初體驗

暨南大學 資工系
吳坤熹
(solomon@mail.ncnu.edu.tw)

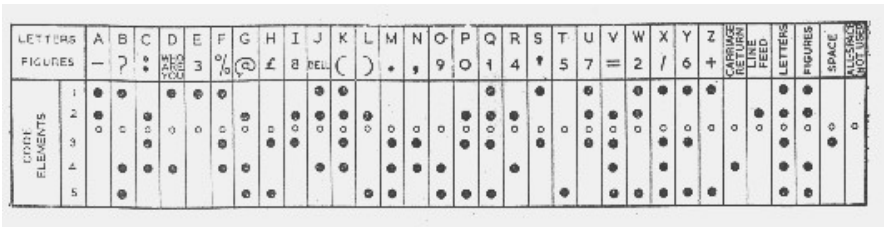
課程大綱

- 要讓電腦有「人工智慧」可以寫詩
  - 訓練
    - ☞Data collection
    - ☞Data cleaning
    - ☞Model building
  - 產出
    - ☞Text Synthesis
- 程式寫作環境：
  - 在 Google Colab 上以 Python 寫作。
- Week 2 - 字串處理
- Week 3 - 字典與檔案
- Week 4 - 網路爬蟲 + 凱比機器人
- Week 5 - 新詩合成
- Week 6 - 作品賞析

Main Power of Computers

Ballistic Missile

Code Breaking



Different Data Type, Different Computing

Numbers

- a = 10
- b = 20
- c = a + b
- print(c)

30

Texts

- a = "10"
- b = "20"
- c = a + b
- print(c)

1020

Operations of Strings: + *

- print("=" * 13)
- print("Good" + " " + "morning!")
- print("=" * 13)

```
=============
Good morning!
=============
```

- name = "Alice"
- msg = "Hello, " + name
- print(msg)

```
Hello, Alice
```

Data Types Conversion

- a = 10
- b = "20"
- print(a, type(a))
- print(b, type(b))

```
10 <class 'int'>
20 <class 'str'>
```

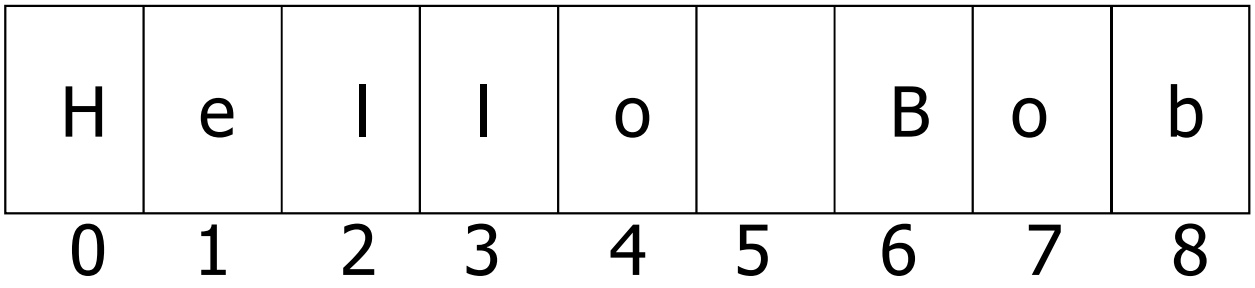
- Type Conversion:
- print(a + int(b))
- print(str(a) + b)

```
30
1020
```

Q: What will happen for

```
print(a+b)
```

Indexing



- The above string has 9 characters. Indexing allows us to retrieve individual characters.
  - s = "Hello Bob"
  - print(s[0], s[8])
  - print(s[-1], s[-3])

```
H b
b B
```

- We can index from the right side using negative indexes.

Slicing

H	e	l	l	o		B	o	b
0	1	2	3	4	5	6	7	8

- We can also obtain a substring (a contiguous sequence of characters) by *slicing*.
  - Note the position end is not included.
- print(s[0:3])
- print(s[5:9])
- print(s[:5])
- print(s[5:])
- print(s[:])

```
Hel
 Bob
Hello
 Bob
Hello Bob
```

len()

- s = "Hello"
- print(len(s))

```
for i in range(5):
    print(i)
```

```
0
1
2
3
4
```

- for i in range(len(s)):
- print(s[i])

```
5
H
e
l
l
o
```

- for c in s:
- print(c)

Python String Operations

Operator	Meaning
+	Concatenation
*	Repetition
<string>[]	Indexing
<string>[:]	Slicing
len(<string>)	Length
for <var> in <string>	Iteration through characters

input() and print()

```
print("Please input a string:", end=' ')
s = input()
print("Its length is", len(s))
```

Without supplying a newline.

Automatically separated by a space.

Example: Stars

- You must have learned to write a program with nested loop to print stars:

```
? 5
*
**
***
****
*****
```

```
N = int(input('? '))
for i in range(1, N+1):
    for j in range(i):
        print('*', end='')
    print()
```

```
N = int(input('? '))
for i in range(1, N+1):
    print('*' * i)
```

Exercise: Stars

- What should be the expression in the 3rd line if you want to print a triangle like the following?

```
N = int(input('? '))
for i in range(1, N+1):
    print(                    )
```

```
? 5
    *
   **
  ***
 ****
*****
```

Courtesy: 高雄女中 方珮雯老師

Parallelogram (平行四邊形)

```
? 5
    #
   ##
  ###
 ####
#####
####
###
##
#
```

- Traditionally, this is done by combining two nested loops, each printing a triangle.

```
N = int(input("? "))
for i in range(1, N+1):
    for j in range(N-i):     # Print N-i spaces
        print(' ', end='')
    for j in range(i):
        print('#', end='')   # Print i '#'
    print()
for i in range(N-1, 0, -1):
    for j in range(i):
        print('#', end='')   # Print i '#'
    print()
```

With skills you learned about strings, only 5 lines

```
N = int(input("? "))
for i in range(1, N+1):
    print(                  )
for i in range(N-1, 0, -1):
    print(     )
```

Challenge:

Can you do it with fewer lines?

The shortest I have seen:
Two lines of Python code!

Exercise: Transpose

■ 輸入一首詩（橫式輸入）後，改成用直式顯示。

松下問童子
言師採藥去
只在此山中
雲深不知處

松 言 只 雲
下 師 在 深
問 採 此 不
童 藥 山 知
子 去 中 處

■ 請試以「五言絕句」、「五言律詩」、「七言絕句」、「七言律詩」各一首來測試你的程式。。

Q: How to end with an empty line?

- sentences = []
- while True:
- line = input("? ")
- if line == '': break
- sentences.append(line)
- print(sentences)

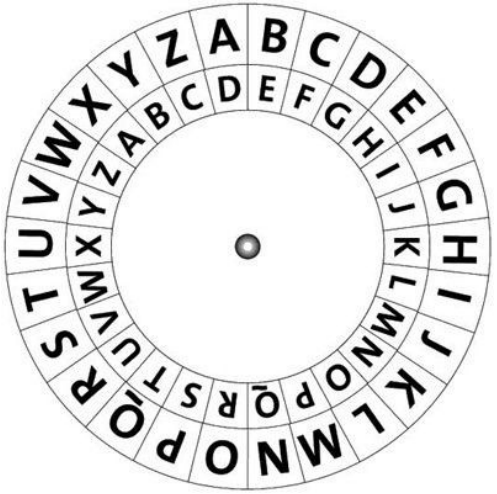
密碼學 (Cryptography)

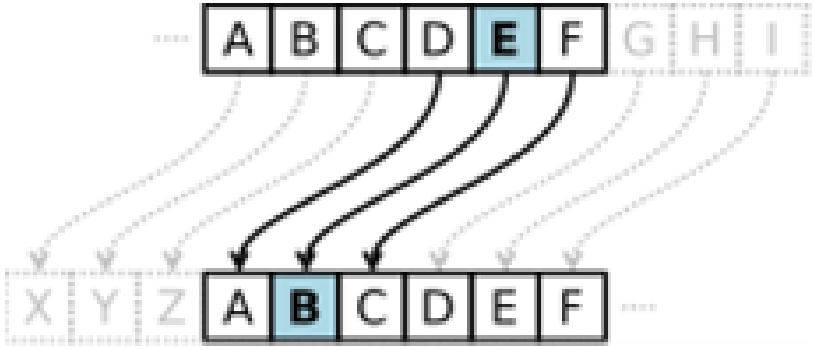
- Some systems encode messages to numbers:
  - HELLO -> 72 69 76 76 79
- Some encodes to a different alphabet:

- A more popular approach is using the same alphabet:
  - HELLO -> KHOOR

Caesar Cipher (凱撒密碼) 49 B.C.

- A -> D
- B -> E
- C -> F
- H -> K
- E -> H
- L -> O
- O -> R





Two Common Questions

- Space and Punctuation?
  - Alittledifficult

- Uppercase vs. Lowercase?
  - ThisisadogThatisacar
  - THISISADOGTHATISACAR

Sample Code

```
# Caesar Cipher
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
msg  = "HELLO"
key = 3
for c in msg:
    i = alphabet.find(c)
    i = (i + key) % 26
    print(alphabet[i], end='')
print()
```

Decryption

- To decrypt the message, you may simple set key = -3 (or 23).
- In fact, because there are only 26 possible key values, you can break this code by brute force -- just try all the 25 key values (key=0 can be skipped).

[Code] brute_force.py

```
alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
msg = "LTTI RTWSNSL"
for key in range(1, 26):
    print(key, '-', end=' ')
    for c in msg:
        i = alphabet.find(c)
        if i >= 0:
          i = (i + 26 - key) % 26
          print(alphabet[i], end='')
        else:
          print(c, end='')
    print()
```

Homework: Variation of Caesar Cipher

- Weakness of the original Caesar Cipher:
  - Only 26 possible key values; only 26 different mappings.
  - For human, it takes a long time to map a message with a key.
  - For a computer, it is extremely fast.

- Let's use an English word as the key.
  - Suppose we choose "UNIVERSITY".

Write down the word and remove duplicate characters.

- ABCDEFGHIJKLMNOPQRSTUVWXYZ

- UNIVERSITY

Copy remaining characters to the 2nd row.

- ABCDEFGHIJKLMNOPQRSTUVWXYZ

- UNIVERSTYAB WXZ

Now you obtain a mapping table.

- ABCDEFGHIJKLMNOPQRSTUVWXYZ

- UNIVERSTYABCDFGHJKLMOPQWXZ

- ATTACK AT DAWN
- UMMUIB UM VUQF

Strength of the Proposed Variation

- This proposed variation:
  - 26! possible mappings. (= 403291461126605635584000000)
  - Is this secure enough?
  - Suppose your 10GHz computer can try 10 billion mappings per second.
  - You have 1,000,000 computers in your lab.
  - There are 365 days in a year; there are 86400 seconds in one day.
  - 403291461126605635584000000 / 10000000000 / 1000000 / 365 / 86400 =
  - 1278 years
    - ☞唐天寶四年(西元745年至今)

Useful String Functions

- lower()
- upper()
- capitalize()
- title()
- strip()
- find()
- count()
- replace()

replace()

- `s = "72 69 76 76 79"`
- `s.replace(' ', ',')`
  - `'72,69,76,76,79'`
- `s.replace(' ', '')`
  - `'7269767679'`

Question: 計算字數

- 琵琶行和長恨歌，哪一首的字數比較多？

- 琵琶行：
  - ...因為長句，歌以贈之。凡六百一十六言，命曰：《琵琶行》。
- 長恨歌
  - long_hate = '漢皇重色思傾國，御宇多年求不得...天長地久有時盡，此恨綿綿無絕期。'
  - punctuation = "，。；？"
  - for c in punctuation:
  - long_hate = long_hate.replace(c, '')
  - print(len(long_hate))

http://poet.ncnu.org/

`isalpha()` **and** `isdigit()`

- 'c'.isalpha()
  - True
- '1'.isalpha()
  - False
- '1'.isdigit()
  - True
- 'c'.isdigit()
  - False

Alphabet:
A~Z, a~z

Digit:
0~9

isdigit() vs. isnumeric()

- '5'.isdigit()
  - True
- '5'.isnumeric()
  - True
- '五'.isdigit()
  - False
- '五'.isnumeric()
  - True

Exercise: 長恨歌密碼

- 在陳郁如的《詩魂》故事中，最後男主角宗元必須解開的密碼，是將《長恨歌》詩中所有一到九的數字逐一按鈕輸入。
- 試撰寫一程式，幫宗元算出他所需要輸入的數字。
- 答案應是：1163319651911177 7
  - 一朝選在君王側　　七月七日長生殿
  - 回眸一笑百媚生
  - 六宮粉黛無顏色

Homework: 長恨歌密碼（三）

- 加強版：「百」、「千」、「萬」也要處理。
- 「千乘萬騎」對應到1000和10000。
- 但「三千寵愛」對應到 3000 而非 31000。

- 答案應是：111006300030001910001000010065191111177
  - 回眸一笑百媚生
  - 九重城闕煙塵生，千乘萬騎西南行
  - 翠華搖搖行復止，西出都門百餘里

Lists

- You can store many data in a list and accessing each item by its index,
  - Just as you access each character in a string.
  - a.k.a. (also known as) “arrays” in other computer languages.

- name = ["Alice", "Bob", "Carol", "Daniel"]
- grade = [75, 28, 34, 70]
- print(name[2], grade[2])

Iterating Through a List

```
name = ["Alice", "Bob", "Carol", "Daniel"]
stu_id   = [11, 12, 13, 14]
grade = [75, 28, 34, 70]
for i in range(4):
    print(name[i], stu_id[i], grade[i], sep='\t')
```

Q: Will it be different if we don't specify the separator?

```
Alice	11	75
Bob	12	28
Carol	13	34
Daniel	14	70
```

split() and join()

- s = "a tale of two cities"
- aList = s.split()
  - ['a', 'tale', 'of', 'two', 'cities']

- aString = ' '.join(aList)
  - 'a tale of two cities‘

```
>>> ':'.join(aList)
'a:tale:of:two:cities'

>>> ','.join(aList)
'a,tale,of,two,cities'

>>> ''.join(aList)
'ataleoftwocities'
```

A List as Counters

```
import random
count = [0] * 13
for i in range(100):
    d1 = random.randint(1, 6)
    d2 = random.randint(1, 6)
    total = d1 + d2
    count[total] = count[total] + 1

for i in range(2, 13):
    print(i, '#' * count[i], count[i])
```

```
2 # 1
3 ## 2
4 ##### 5
5 ############### 15
6 ################## 17
7 ######################## 23
8 ########## 10
9 ########## 10
10 ######## 8
11 ##### 5
12 #### 4
```

Exercise: Occurrence of Characters

- Dickens: A Tale of Two Cities

```
s = "book"
alphabet = 'abcdefghijklmnopqrstuvwxyz'
count = [0] * 26
for c in s.lower():
    if c.isalpha():
        i = alphabet.find(c)
        count[i] = count[i] + 1
for i in range(26):
    if count[i] > 0:
        print(alphabet[i], count[i])
```