Python

可能不是第一堂的 Python 課

ANACONDA

- 免費開源
- 擁有絕大多數需要用到的套件
- 一種肥大的蛇、是全世界最重的蛇
- 有點太胖了，浪費記憶體

MINICONDA

- 佔用空間較小
- 大概是還沒長大的 Anaconda

Some Command

- Conda's version
  - conda -V
  - conda --version
- conda --help
- conda list

Python's Version

- python -V || python --version
- Python 2.7 將於 2020年停止維護
- 目前最新穩定版本：Python 3.7
- Python 3.x 並不向下相容

Jupyter

- Jupyter = Julia + Python + R
- 互動式開發環境
- jupyter notebook
- jupyter notebook list
- 內建 Markdown語法

Python

- 動態型別(dynamic typing)
  - 在 runtime 進行型別檢查
- 強型別(strong typing)
  - 永遠會進行型別檢查

Easter Egg

- import this

Beautiful is better than ugly.
優美勝於醜陋。

Explicit is better than implicit.
明確勝於晦澀。

Simple is better than complex.
簡單勝於複雜。

Complex is better than complicated.
複雜勝於繁複。

Flat is better than nested.
平坦勝於築巢。

Sparse is better than dense.
分散勝於密集。

Readability counts.
可讀性很重要。

Special cases aren't special enough to break the rules.
特例也不該違背這些規則，

Although practicality beats purity.
即使實用性打敗了純粹性。

Python，以編寫優美的程式碼為目標。

而優美的程式碼應該簡單明瞭。

優美的程式碼應該編寫簡單，不該有複雜的關係。

即使需要複雜的關係，也不該有繁複的介面。

優美的程式碼不該有過多的內嵌結構。

優美的程式碼寧願分散程序，也不該擠在一行。

優美的程式碼，一定要易讀，加上註解吧。

這些規則應當遵守，就算傷害了程式碼的實用性。

Errors should never pass silently.
錯誤不該被無聲地忽略，

Unless explicitly silenced.
除非你如此期望。

除非需要，否則要捕捉所有的錯誤。

In the face of ambiguity, refuse the temptation to guess.
面對雙關的語意時，拒絕猜測的誘惑。

There should be one– and preferably only one –obvious way to do it.
用明顯的方法來完成一件事，而且最好只有一種。

不要去猜想完成程序的方式，只需要用一種明顯的解法。

Although that way may not be obvious at first unless you're Dutch.
這並不是件容易的事，誰叫你不是荷蘭人呢？

找出一種明顯的解法，在一開始並不容易，畢竟我們都不是Python之父。

Now is better than never.
把握現在勝於停滯不前，

Although never is often better than *right* now.
即使停滯不前勝於立刻動手。

先考慮過程式是好的，但必須要動手寫。

If the implementation is hard to explain, it's a bad idea.
如果實作難以被說明，那就是個壞主意。

If the implementation is easy to explain, it may be a good idea.
如果實作能輕鬆說明，那可能是個好主意。

能夠被說明的程式才是好程式。

Namespaces are one honking great idea — let's do more of those!
命名空間是個絕妙的點子，我們應當多加利用！

善用python的命名空間。