

# A Simple Example for use Python Compiler

yen3

SoC Lab, CGUCSIE

January 5, 2010

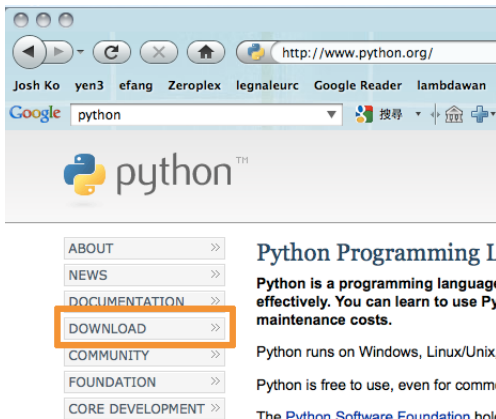
About

Download

Compile and Use

Discusses

- ▶ Go to <http://www.python.org> and press “Download”



► Choose “Python 2.6.4 Windows Installer”

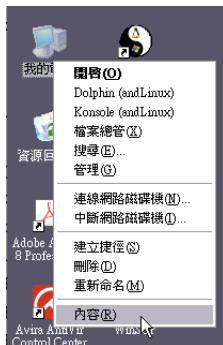
For the MD5 checksums and OpenPGP signatures, look at the [detailed Python 2.6.4](#) page:

- [Python 2.6.4 Windows installer](#) (Windows binary -- does not include source)
- [Python 2.6.4 Windows AMD64 installer](#) (Windows AMD64 binary -- does not include source)
- [Python 2.6.4 Mac Installer Disk Image](#)
- [Python 2.6.4 compressed source tarball](#) (for Linux, Unix or OS X)
- [Python 2.6.4 bziped source tarball](#) (for Linux, Unix or OS X, more compressed)

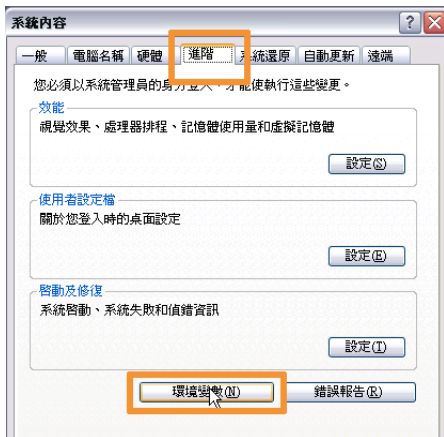
# Install

- ▶ 請自行安裝

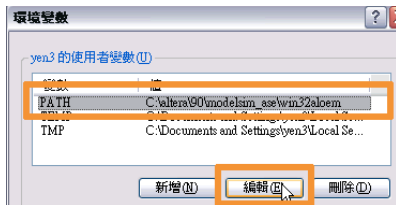
► 我的電腦 → 右鍵 → 內容



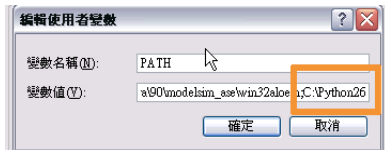
## ► 進階 → 環境變數



► 選擇 PATH → 編輯



► 在變數值後加上 “;C:\Python26”



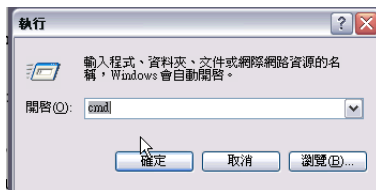


## start to compile

- ▶ 開始 → 執行

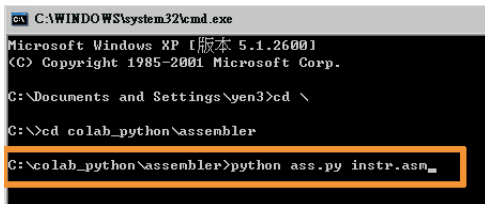


- ▶ 輸入 cmd 之後確定



## Produce Instruction Memory

- ▶ 到存放 `ass.py` 與 `instr.asm` 的資料夾
- ▶ 輸入 “`python ass.py instr.asm`” (`instr.asm` 是你的輸入檔名, 自行改變成自己的檔名)



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [版本 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\yen3>cd \

C:\>cd colab_python\assembler

C:\colab_python\assembler>python ass.py instr.asm
```

## Produce Instruction Memory

- ▶ 會看到編譯後的成果, 並產生相對應的 .mem 檔, 即可使用, 如果有錯誤訊息也會顯示在上面。

```

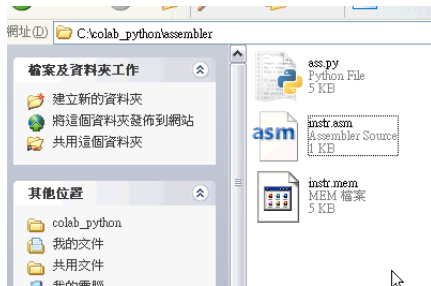
C:\WINDOWS\system32\cmd.exe

Final Result
0: 0000000 100 101 000      MOVA   R4      R5
1: 0000001 101 110 000      INC    R5      R6
2: 0000010 001 010 011      ADD    R1      R2      R3
3: 0000101 011 100 101      SUB    R3      R4      R5
4: 0000110 110 111 000      DEC    R6
5: 0001000 001 001 011      AND    R1      R1      R3
6: 0001001 010 011 100      OR     R2      R3      R4
7: 0001010 010 011 100      XOR    R2      R3      R4
8: 0001011 101 110 000      NOT    R5
9: 0001100 101 000 110      MOVB   R5      R6
10: 1001100 001 000 101      LDI    R1      5
11: 0001101 110 000 111      SHR    R6      R7
12: 0001110 110 000 111      SHL    R6      R7
13: 1000010 010 010 111      ADI    R2      R2      7
14: 0010000 110 111 000      LD     R6      R7
15: 0100000 000 110 011      ST     R6      R3
16: 0000000 000 000 000
17: 0000000 000 000 000
18: 1100000 000 111 101      BRZ    R7      5
19: 1100001 111 101 011      BRN    R5      -5
20: 1110000 000 110 000      JMP    R6

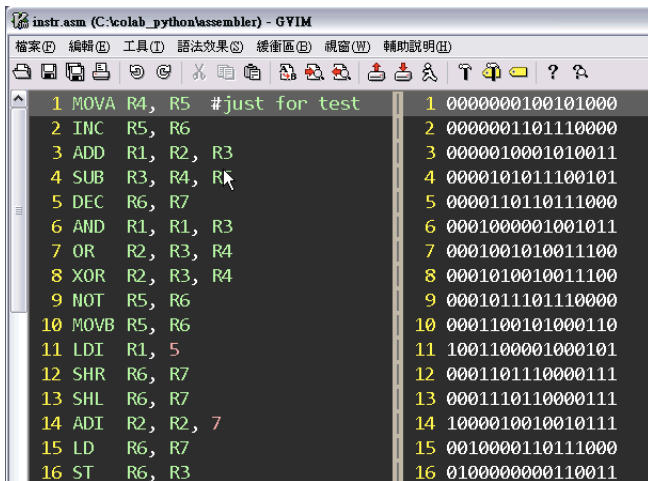
C:\colab_python\assembler>

```

► 打開該資料夾會看到的相對應檔案



- ▶ 該檔案是一對一對應，可自行除錯



The screenshot shows the GYIM assembler interface with the file 'instr.asm' open. The window title is 'instr.asm (C:\colab\_python\assembler) - GYIM'. The menu bar includes '檔案(F)', '編輯(E)', '工具(T)', '語法效果(S)', '緩衝區(B)', '視窗(W)', and '輔助說明(H)'. The toolbar contains various icons for file operations, editing, and viewing. The main window displays assembly code on the left and its binary representation on the right, separated by a vertical line. The code is as follows:

Line	Assembly Code	Binary Representation
1	MOVA R4, R5 #just for test	0000000100101000
2	INC R5, R6	0000001101110000
3	ADD R1, R2, R3	0000010001010011
4	SUB R3, R4, R5	0000101011100101
5	DEC R6, R7	0000110110111000
6	AND R1, R1, R3	0001000001001011
7	OR R2, R3, R4	0001001010011100
8	XOR R2, R3, R4	0001010010011100
9	NOT R5, R6	0001011101110000
10	MOVB R5, R6	0001100101000110
11	LDI R1, 5	1001100001000101
12	SHR R6, R7	0001101110000111
13	SHL R6, R7	0001110110000111
14	ADI R2, R2, 7	1000010010010111
15	LD R6, R7	0010000110111000
16	ST R6, R3	0100000000110011

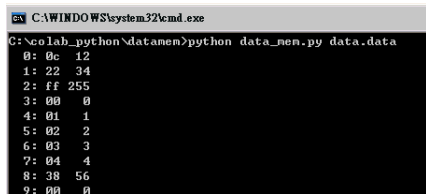
## Produce Data Memory

- ▶ 到存放 data\_mem.py 與 data.data 的資料夾
- ▶ 輸入 “python data\_mem.py data.data” (data.data 是你的輸入檔名, 自行改變成自己的檔名)



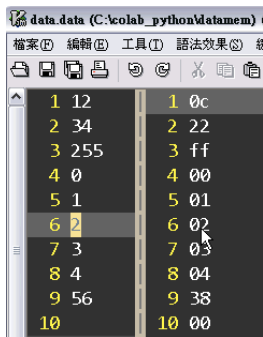
```
C:\>cd colab_python\datanem
C:\colab_python\datanem>python data_mem.py data.data
```

- ▶ 會看到編譯後的成果, 並產生相對應的 .mem 檔, 即可使用, 如果有錯誤訊息也會顯示在上面。



```
CAWINDOWS\system32\cmd.exe
C:\colab_python\datanem>python data_mem.py data.data
0: 0c 12
1: 22 34
2: ff 255
3: 00 0
4: 01 1
5: 02 2
6: 03 3
7: 04 4
8: 38 56
9: 00 0
```

- ▶ 該檔案也是一對一對應，可自行除錯



1	12	1	0c
2	34	2	22
3	255	3	ff
4	0	4	00
5	1	5	01
6	2	6	02
7	3	7	03
8	4	8	04
9	56	9	38
10		10	00

# Do you have any problem ?