

# LẬP TRÌNH PYTHON CƠ BẢN

(Basic Python Programming)

Th.S Nguyễn Hoàng Thành

Email: [thanhnh@ptithcm.edu.vn](mailto:thanhnh@ptithcm.edu.vn)

Tel: 0909 682 711

# TỔNG QUAN VỀ NGÔN NGỮ LẬP TRÌNH PYTHON

# Các câu hỏi thường gặp về Python

- Tại sao người ta sử dụng Python?
- Python có phải là ngôn ngữ hướng kịch bản?
- Ai sử dụng Python hiện nay?
- Bạn có thể làm gì với Python?
- Python được hỗ trợ như thế nào?
- Điểm mạnh của Python?

# 1. LỊCH SỬ NGÔN NGỮ PYTHON

# TẠI SAO LẠI CÓ TÊN PYTHON

Tác giả Python **Guido van Rossum** đã đặt tên nó theo loạt phim hài **Monty Python's Flying Circus** của đài **BBC**.



Guido van Rossum



# Ý TƯỞNG CỦA PYTHON

- Python được hình thành vào **cuối những năm 1980**. Thời gian đó, Guido van Rossum làm việc trong một dự án tại CWI, có tên là Amoeba, một hệ điều hành phân tán.



# Ý TƯỞNG CỦA PYTHON

“ Vào đầu những năm 1980, tôi làm việc với tư cách là người triển khai một nhóm **xây dựng ngôn ngữ có tên là ABC** tại **Centrum voor Wiskunde en Informatica** (CWI).

Tôi không biết mọi người hiểu như thế nào về Ảnh hưởng của ABC đối với Python. Tôi cố gắng đề cập đến ảnh hưởng của ABC vì tôi mang ơn tất cả những gì tôi đã học được trong dự án đó và những người đã làm việc với nó. ”

Guido van Rossum



**NGÔN NGỮ ABC**



# Ý TƯỞNG CỦA PYTHON

“ Tôi nhớ lại tất cả kinh nghiệm và những điều **không hài lòng** của mình với ABC. Tôi quyết định cố gắng thiết kế **một ngôn ngữ kịch bản đơn giản** có một số **đặc tính tốt** hơn của ABC. Vì vậy, tôi bắt đầu nhập.

Tôi đã tạo một **máy ảo đơn giản**, một trình **phân tích cú pháp đơn giản** và một **thời gian chạy đơn giản**. Tôi đã tạo **phiên bản** của riêng mình cho các phần ABC khác nhau mà tôi thích. ”

Guido van Rossum



# Ý TƯỞNG CỦA PYTHON

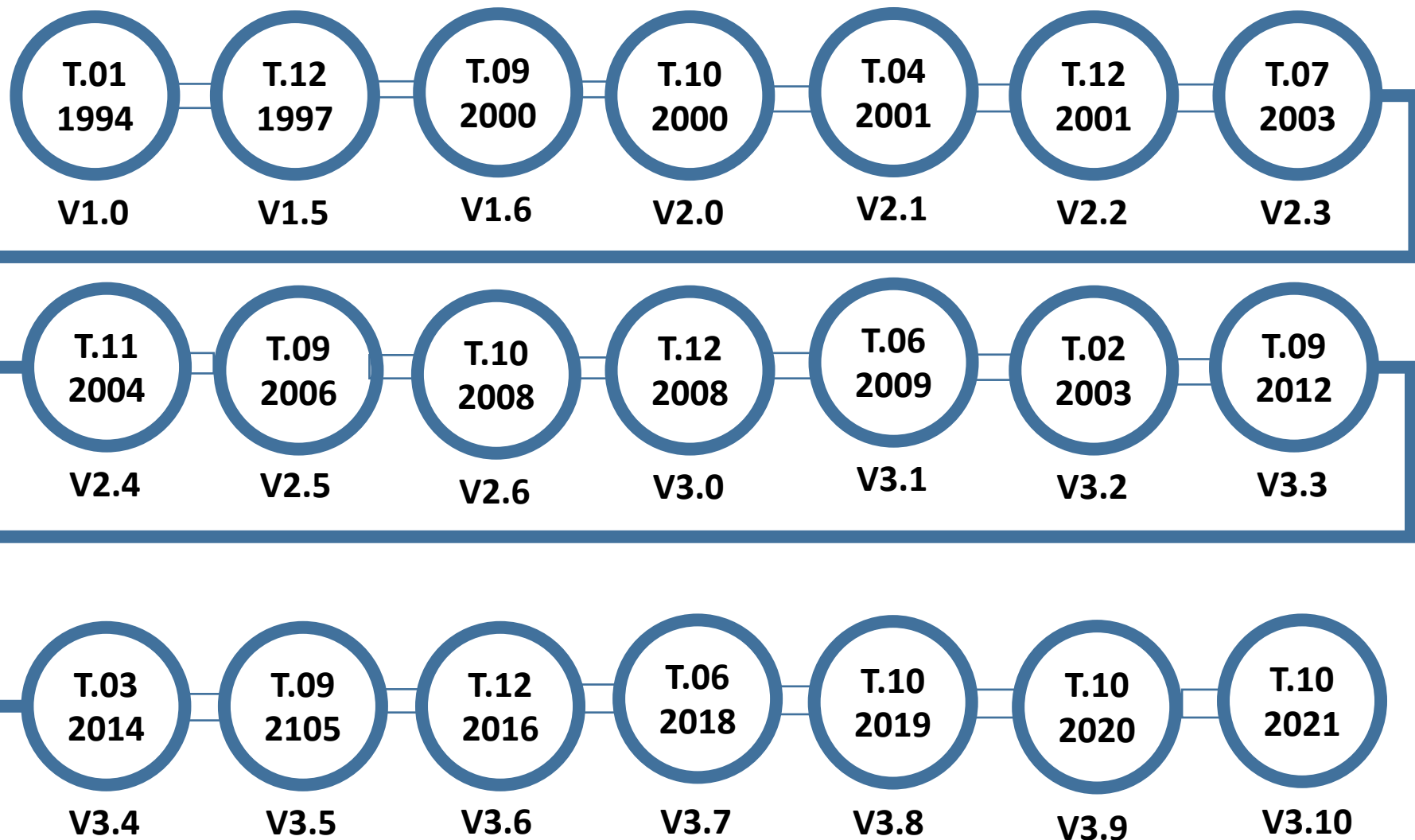
“ Tôi đã tạo một **cú pháp cơ bản**, sử dụng **thụt lề** để **nhóm câu lệnh** thay vì **dấu ngoặc nhọn** hoặc **khởi begin-end** và phát triển một số lượng nhỏ các **kiểu dữ liệu mạnh mẽ**: **bảng băm** (hoặc từ điển, như chúng tôi gọi), **danh sách**, **chuỗi** và **số**. ”

Guido van Rossum

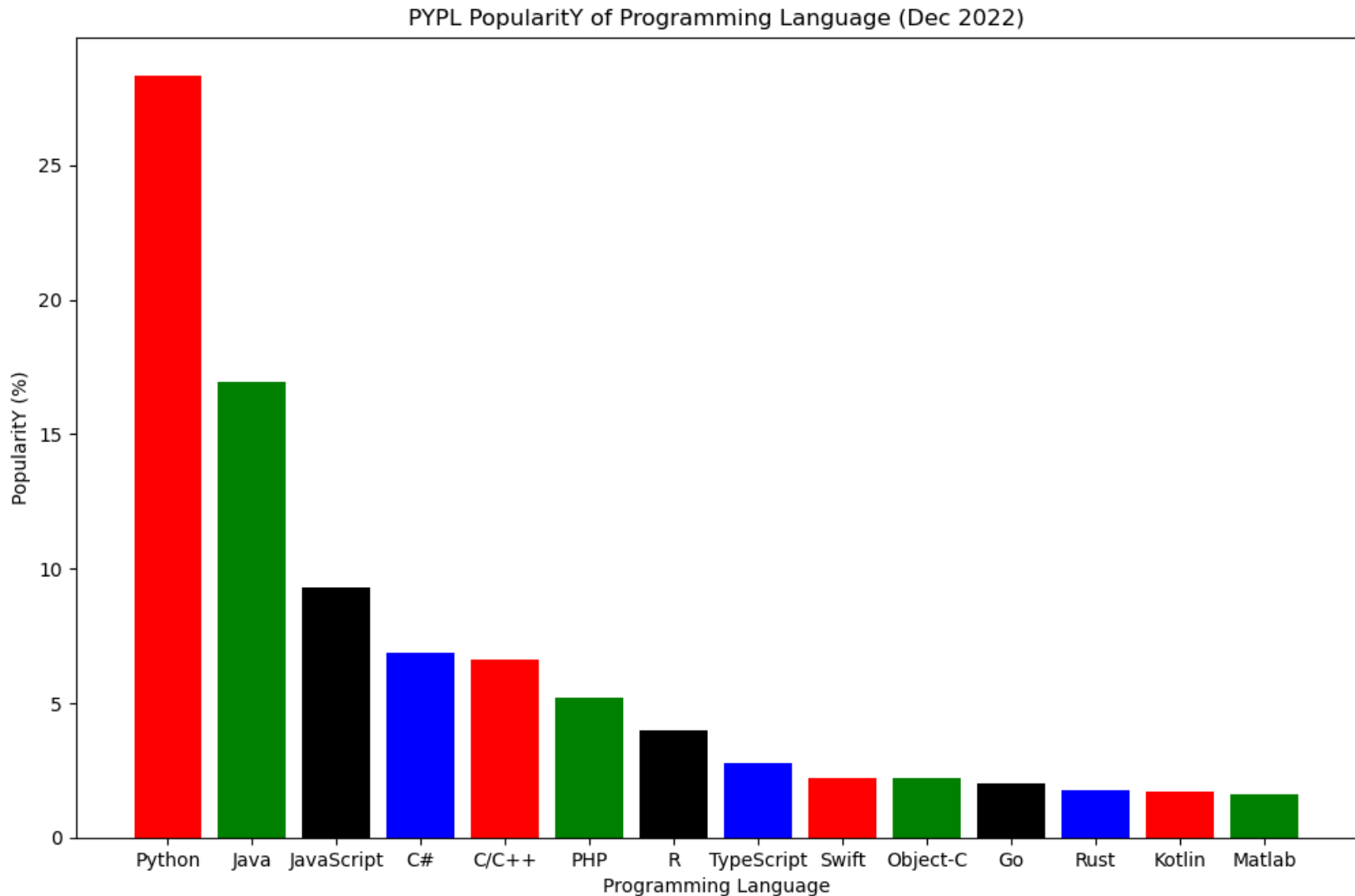
# Mục tiêu Python

- Một ngôn ngữ **đễ dàng và trực quan** cũng **mạnh mẽ** như ngôn ngữ của các đối thủ cạnh tranh lớn;
- **Nguồn mở**, vì vậy bất kỳ ai cũng có thể đóng góp vào sự phát triển của nó;
- Mã dễ hiểu như **tiếng Anh đơn giản** ;
- Thích hợp **cho các công việc hàng ngày**, cho phép thời gian phát triển ngắn.

# Các phiên bản Python



# Tại sao người ta sử dụng Python?



Source: <https://pypl.github.io/PYPL.html>

# Tại sao người ta sử dụng Python? (tt)

- Software quality – Chất lượng phần mềm
- Developer productivity – Năng suất nhà phát triển
- Program portability – Tính di động
- Support libraries – Hỗ trợ nhiều thư viện
- Component integration – Tích hợp nhiều thành phần
- Enjoyment – Yêu thích

# Software quality – Chất lượng phần mềm

- Theo thiết kế, Python thực hiện một cú pháp đơn giản và dễ. Điều này làm cho ngôn ngữ dễ học, dễ hiểu và dễ nhớ hơn.
- Python áp dụng một cách tiếp cận hơi tối giản.
- Python bao gồm các công cụ như mô-đun và OOP thúc đẩy khả năng sử dụng lại mã một cách tự nhiên.

Và bởi vì Python tập trung vào chất lượng, nên các lập trình viên Python cũng vậy.

# Developer productivity – Năng suất nhà phát triển

- Rất khó tìm đủ lập trình viên để thực hiện các dự án phần mềm;
- Các nhà phát triển được yêu cầu triển khai các hệ thống nhanh chóng.

=> Python là một công cụ cho phép các lập trình viên làm được nhiều việc hơn với ít nỗ lực hơn.

# Python có phải là ngôn ngữ hướng kịch bản?

- Là ngôn ngữ kịch bản **hướng đối tượng**
- Là sự kết hợp hỗ trợ cho OOP với định hướng tổng thể đối với các vai trò kịch bản.



# Ai sử dụng Python hiện nay?

- Hiện có khoảng **1 triệu người** dùng Python trên khắp thế giới.
  - Mã nguồn mở
  - Hoàn toàn không có đăng ký giấy phép
  - Tự động bao gồm:
    - Bản phân phối Linux,
    - Máy tính Macintosh, và
    - Một số sản phẩm và phần cứng

# Ai sử dụng Python hiện nay? (tt)

- Python cũng đang được các công ty thực tế áp dụng trong các sản phẩm tạo ra doanh thu:
  - Google: các hệ thống tìm kiếm trên web
  - Dịch vụ chia sẻ video YouTube: phần lớn được viết bằng Python
  - BitTorrent
  - Khung phát triển web App Engine phổ biến của Google
  - EVE Online
  - Maya

# Bạn có thể làm gì với Python?

- Systems Programming
- GUIs
- Internet Scripting
- Component Integration
- Database Programming
- Rapid Prototyping
- Numeric and Scientific Programming
- Gaming, Images, Serial Ports, XML, Robots, and More

# Python được hỗ trợ như thế nào?

- Cộng đồng phát triển lớn và tích cực
- PSF (Python Software Foundation), một nhóm phi lợi nhuận chính thức, tổ chức các hội nghị và giải quyết các vấn đề về sở hữu trí tuệ.
- Nhiều hội nghị về Python được tổ chức trên khắp thế giới; OSCON của O'Reilly và PyCon của PSF là lớn nhất.

# Điểm mạnh của Python?

- Hướng đối tượng
- Miễn phí
- Di động
- Mạnh mẽ
- Nó có thể mix được
- Dễ dàng để sử dụng
- Dễ dàng để học

# CÁCH PYTHON CHẠY CHƯƠNG TRÌNH

# Python Interpreter

## ▪ **Trình thông dịch**

- là một loại chương trình thực thi các chương trình khác.
- đọc chương trình và thực hiện các hướng dẫn trong đó.
- Một gói Python khi được cài đặt, nó **tạo ra một số thành phần**, tối thiểu là **một trình thông dịch** và một thư viện hỗ trợ.
- Mã Python luôn được chạy bởi trình thông dịch này.

# Cài đặt Python

- **Người dùng Windows** tìm nạp và chạy tệp thực thi tự cài đặt để đặt Python trên máy của họ. Chỉ cần nhấp đúp và nói Có hoặc Tiếp theo ở tất cả các lời nhắc.
- **Người dùng Linux và Mac OS X** có thể đã cài đặt sẵn Python có thể sử dụng được trên máy tính của họ - nó là một thành phần tiêu chuẩn trên các nền tảng này ngày nay.
- **Một số người dùng Linux và Mac OS X** (và hầu hết người dùng Unix) biên dịch Python từ gói phân phối mã nguồn đầy đủ của nó.
- Người dùng Linux cũng có thể tìm thấy các **tệp RPM** và người dùng Mac OS X có thể tìm thấy các gói cài đặt dành riêng cho Mac khác nhau.



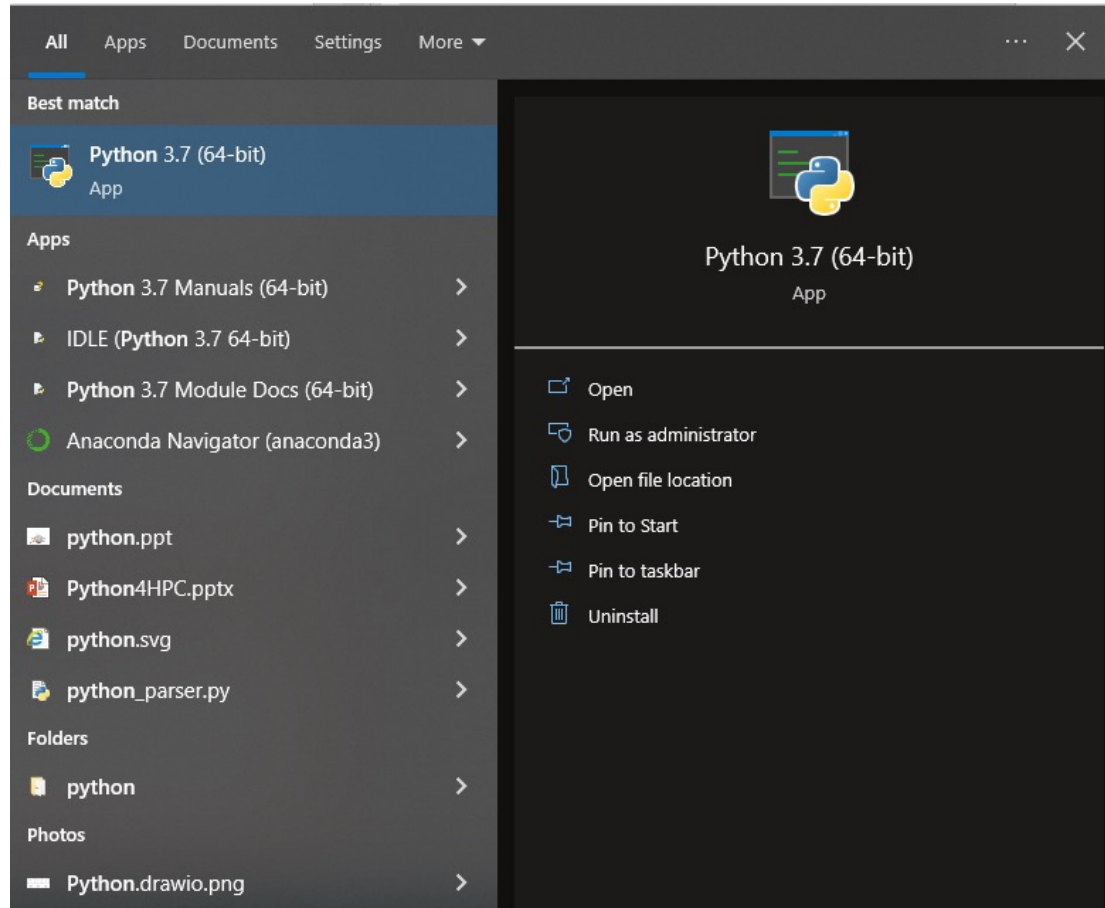
# Thực thi chương trình Python

- Viết và chạy tập lệnh Python phụ thuộc vào việc bạn xem các tác vụ này với tư cách là
  - một lập trình viên hay
  - một trình thông dịch Python.

# Góc nhìn của Lập trình viên

- Chương trình Python chỉ là **một tệp văn bản** chứa các câu lệnh Python
- Có thể tạo một tệp báo cáo như vậy với bất kỳ trình soạn thảo văn bản nào
- Sau khi bạn đã nhập các câu lệnh này vào tệp văn bản, chúng ta cần yêu cầu Python thực thi tệp.

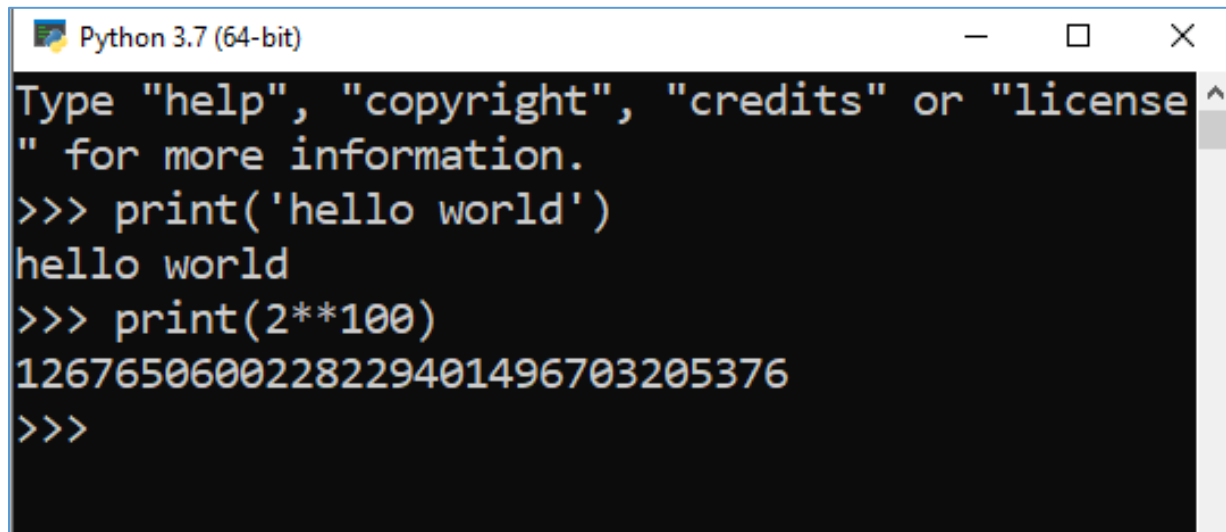
# Góc nhìn của Lập trình viên (tt)



# Góc nhìn của Lập trình viên (tt)

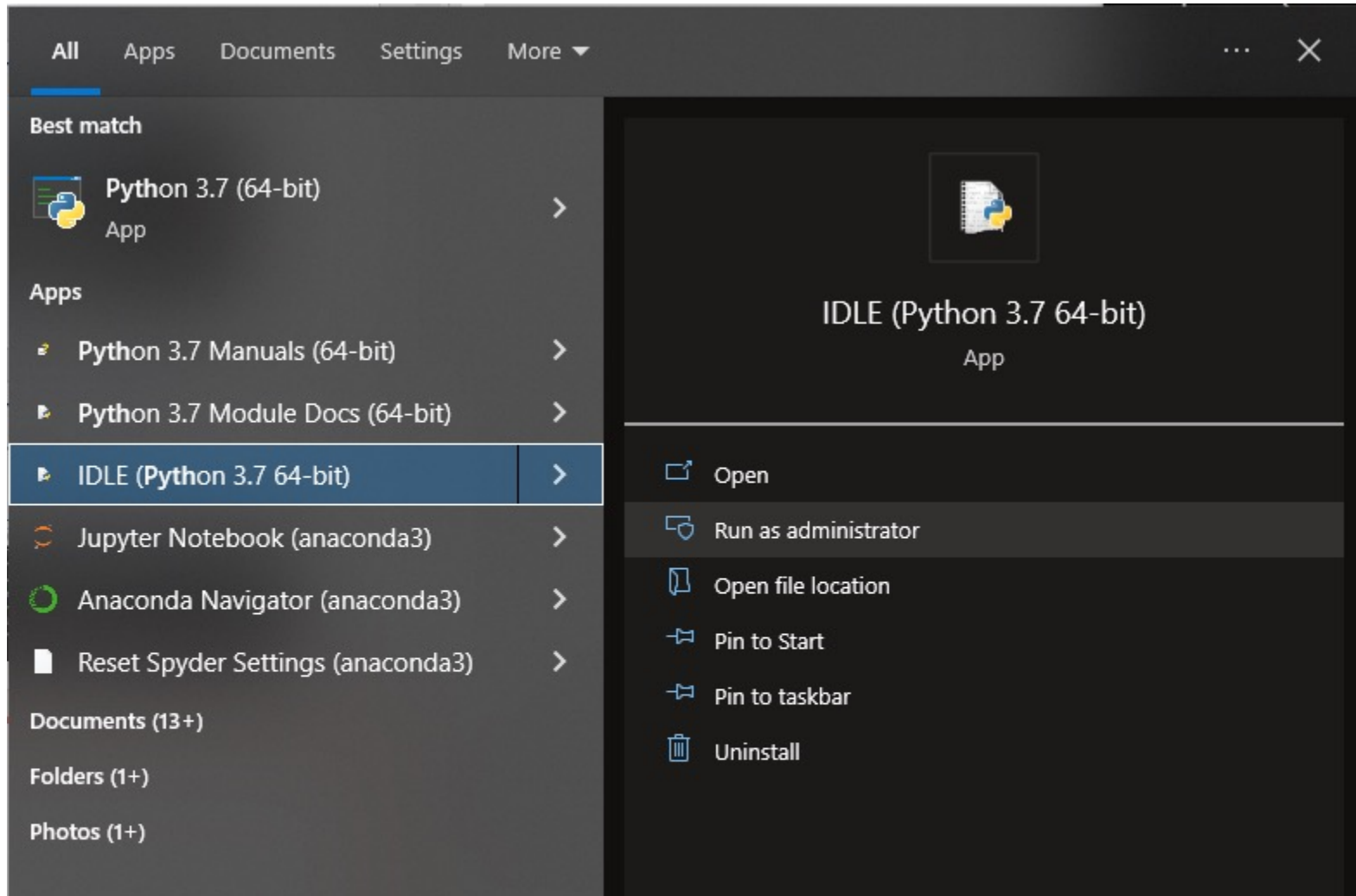
```
print('hello world')  
print(2 ** 100)
```

Chạy tập lệnh này từ một dòng lệnh DOS



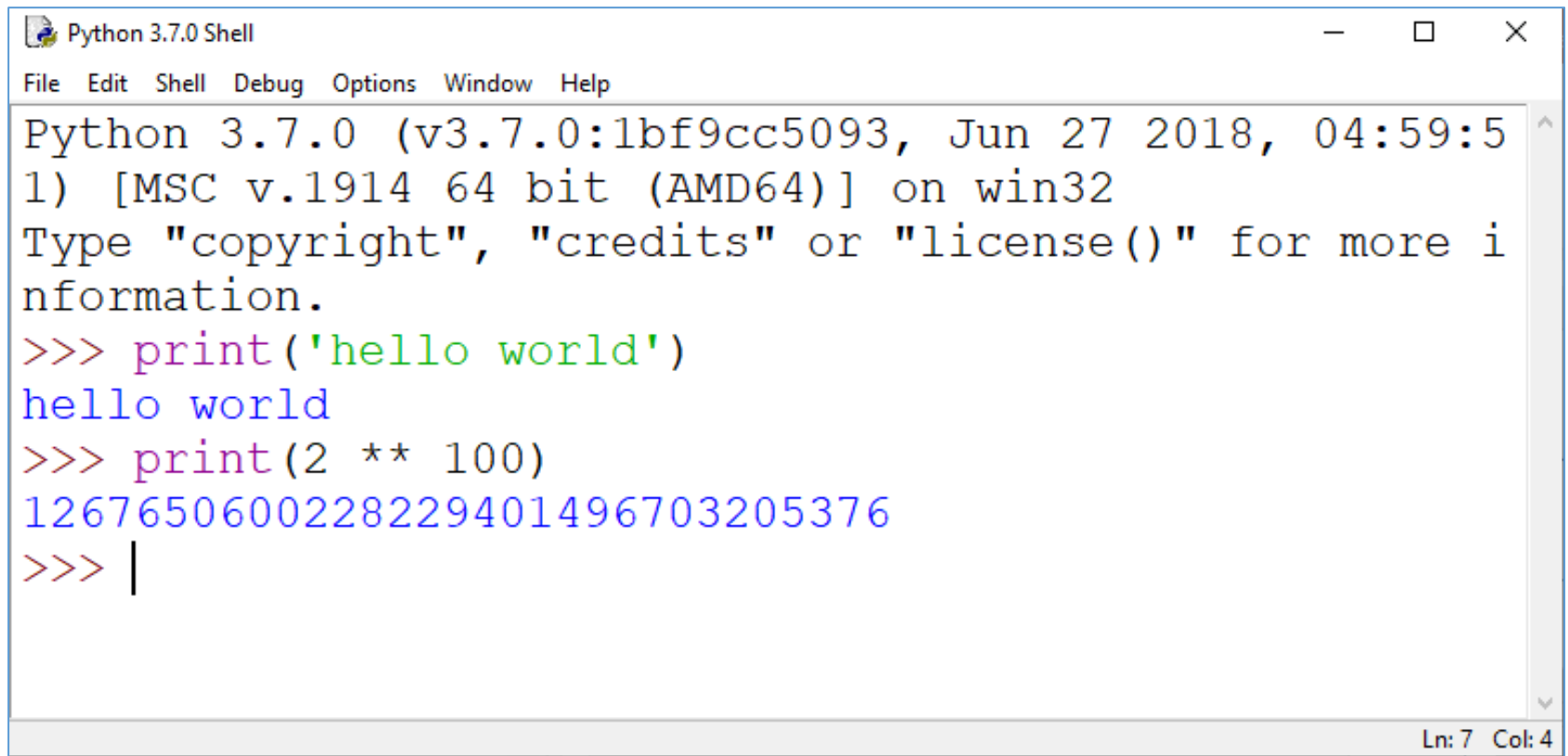
```
Python 3.7 (64-bit)  
Type "help", "copyright", "credits" or "license"  
" for more information.  
>>> print('hello world')  
hello world  
>>> print(2**100)  
1267650600228229401496703205376  
>>>
```

# Góc nhìn của Lập trình viên (tt)



# Góc nhìn của Lập trình viên (tt)

Run this script from a Python Shell

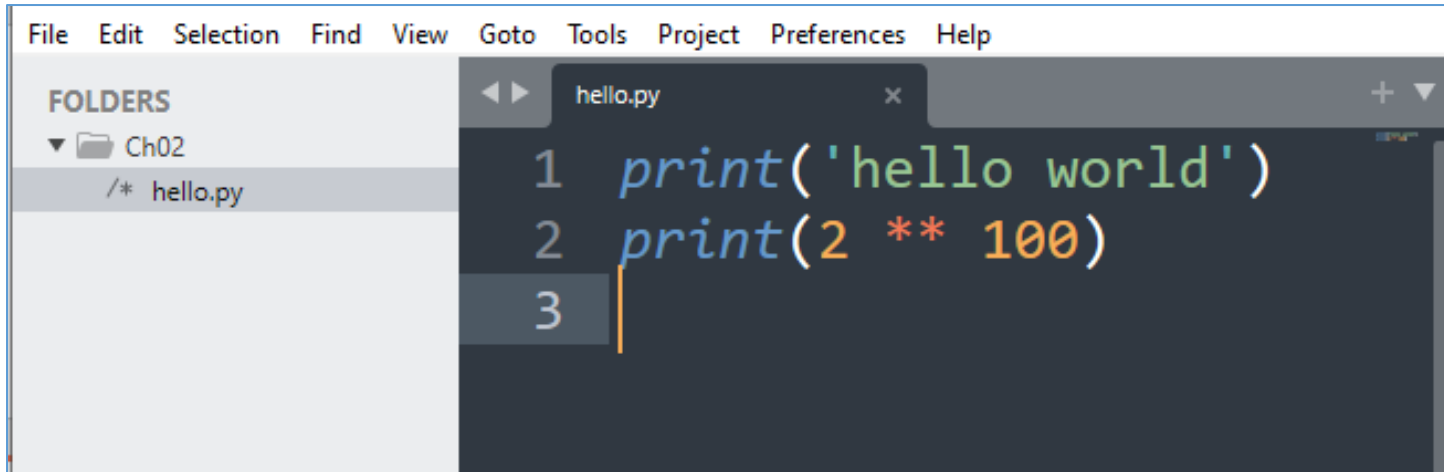
A screenshot of a Python 3.7.0 Shell window. The window has a title bar with the text "Python 3.7.0 Shell" and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with the following items: File, Edit, Shell, Debug, Options, Window, and Help. The main area of the window contains the following text:

```
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more i
nformation.
>>> print('hello world')
hello world
>>> print(2 ** 100)
1267650600228229401496703205376
>>> |
```

The text is color-coded: the prompt ">>>" is red, the code is black, the output is blue, and the cursor "|" is black. The status bar at the bottom right of the window shows "Ln: 7 Col: 4".

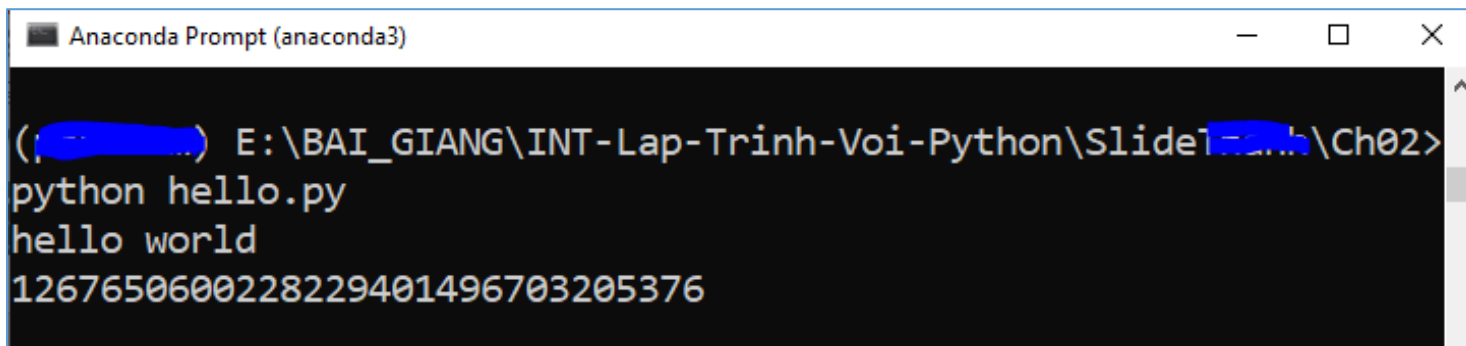
# Góc nhìn của Lập trình viên (tt)

- Chạy tập lệnh này từ một dòng lệnh DOS



The screenshot shows an IDE window with a menu bar (File, Edit, Selection, Find, View, Goto, Tools, Project, Preferences, Help) and a sidebar labeled 'FOLDERS' containing 'Ch02' and '/\* hello.py'. The main editor area shows the file 'hello.py' with the following code:

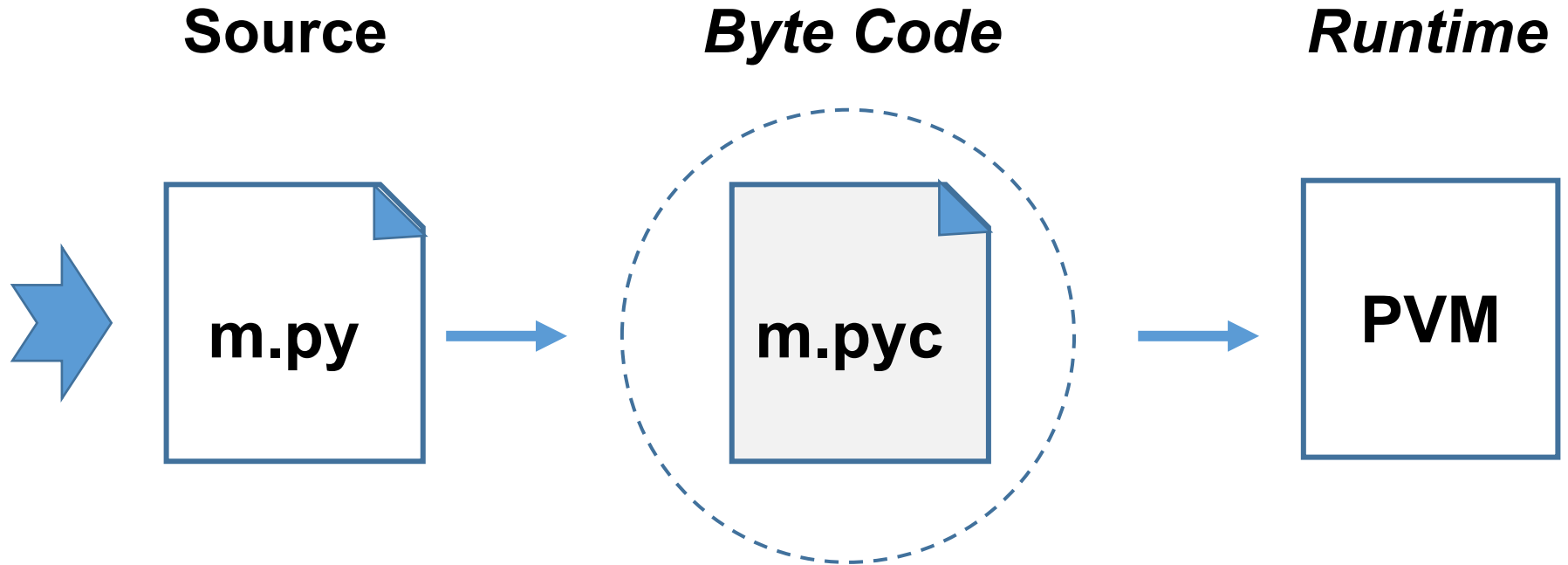
```
1 print('hello world')
2 print(2 ** 100)
3
```



The screenshot shows an 'Anaconda Prompt (anaconda3)' window. The command prompt shows the user running 'python hello.py' in the directory 'E:\BAI\_GIANG\INT-Lap-Trinh-Voi-Python\Slide\Ch02'. The output of the script is displayed on the next two lines:

```
([redacted]) E:\BAI_GIANG\INT-Lap-Trinh-Voi-Python\Slide\Ch02>
python hello.py
hello world
1267650600228229401496703205376
```

# Góc nhìn của Python





# Góc nhìn của Python (tt)

- Biên dịch mã byte
- Máy ảo Python (PVM)
- Ý nghĩa hiệu suất
- Ý nghĩa phát triển

# Biên dịch mã byte

- Python sẽ biên dịch mã nguồn thành một định dạng được gọi là mã byte.
- Biên dịch chỉ đơn giản là một bước dịch và mã byte là cấp thấp hơn.
- Python lưu mã byte (.pyc) để tối ưu hóa tốc độ khởi động.
- Nếu Python không thể ghi tệp mã byte vào máy, chương trình vẫn hoạt động

# Máy ảo Python (PVM)

- Khi chương trình của bạn đã được biên dịch thành mã byte
- Nó được vận chuyển để thực thi một thứ thường được gọi là Máy ảo Python
- PVM chỉ là một vòng lặp lớn lặp lại từng hướng dẫn mã byte của bạn để thực hiện các hoạt động của chúng
- PVM là *runtime engine* của Python;

# Ý nghĩa hiệu suất

- Mã byte Python không phải là mã máy nhị phân
- Mã byte là một đại diện dành riêng cho Python
- Mã Python có thể không chạy nhanh như mã C hoặc C++
- Mã Python chạy ở tốc độ nằm giữa tốc độ của ngôn ngữ biên dịch truyền thống và ngôn ngữ thông dịch truyền thống.

# Ý nghĩa phát triển

- Không cần biên dịch trước và liên kết trước khi bắt đầu thực thi.
- Đơn giản chỉ cần gõ và chạy mã
- Cấu trúc này cũng là lý do tại sao Python tự cho phép tùy chỉnh sản phẩm. Vì vậy, mã Python có thể được thay đổi nhanh chóng

# Các biến thể của mô hình thực thi

- Các lựa chọn thay thế triển khai Python
- Công cụ tối ưu hóa thực thi
- Frozen Binaries
- Các tùy chọn thực thi khác

# Các lựa chọn thay thế triển khai Python

- Ba triển khai chính của ngôn ngữ Python
  - Cpython
  - Jython
  - IronPython
- CPython là triển khai tiêu chuẩn

# CPython

- Việc triển khai Python ban đầu và tiêu chuẩn thường được gọi là Cpython.
- Đây là Python được cài đặt từ [http://www .python.org](http://www.python.org), nhận bản phân phối ActivePython và có tự động trên hầu hết các máy Linux và Mac OS X



# Jython

- Được nhắm mục tiêu để tích hợp với ngôn ngữ lập trình Java
- Jython bao gồm các lớp Java biên dịch mã nguồn Python thành **mã byte Java** và sau đó định tuyến mã byte kết quả tới Máy ảo Java

# IronPython

- IronPython được thiết kế để cho phép các chương trình Python tích hợp với các ứng dụng được mã hóa để hoạt động với .NET Framework cho Windows của Microsoft.
- IronPython cho phép các chương trình Python hoạt động như cả thành phần máy khách và máy chủ, có thể truy cập được từ các ngôn ngữ .NET khác.

# Công cụ tối ưu hóa thực thi

- CPython, Jython và IronPython đều triển khai ngôn ngữ Python theo những cách tương tự:
  - bằng cách biên dịch mã nguồn thành mã byte và
  - thực thi mã byte trên một máy ảo thích hợp.
- Còn các hệ thống khác, bao gồm trình biên dịch just-in-time của Psyco và trình dịch Shedskin C++, thay vào đó cố gắng tối ưu hóa mô hình thực thi cơ bản.

# Trình biên dịch tức thời của Psycho

- Mở rộng mô hình thực thi mã byte để làm cho các chương trình chạy nhanh hơn.
- Psycho là một cải tiến cho PVM.
- Sau đó, mã máy sẽ thay thế phần tương ứng của mã byte gốc để tăng tốc độ thực thi tổng thể của chương trình.
- Mã Python có thể trở nên nhanh như mã C được biên dịch trong Psycho.
- Psycho thường được biết đến như một trình biên dịch đúng lúc (JIT).



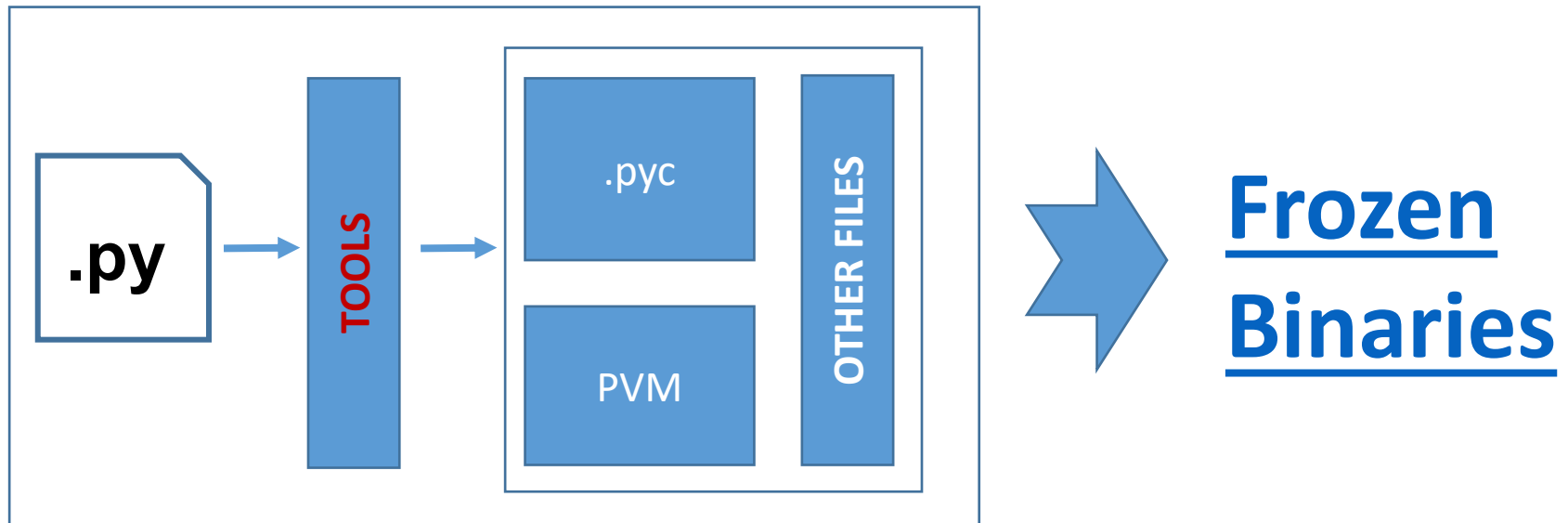
<https://psyco.sourceforge.net/>

# Trình dịch Shedskin C++

- Dịch mã nguồn Python sang mã C++
- Trình biên dịch C++ sau đó biên dịch thành mã máy.
- Shedskin vẫn còn đang thử nghiệm

# Frozen Binaries

- Đưa các chương trình Python thành các tệp thực thi
- Được gọi là Frozen binaries
- Kết hợp mã byte của các tệp chương trình, cùng với PVM (trình thông dịch) và bất kỳ tệp hỗ trợ Python nào mà chương trình cần, thành một gói duy nhất.



# Frozen Binaries (tt)

- Ba hệ thống chính có khả năng tạo nhị phân cố định:
  - **py2exe**: dành cho Windows
  - **PyInstaller**: tương tự như py2exe nhưng cũng hoạt động trên Linux và Unix và có khả năng tạo các tệp nhị phân tự cài đặt.
  - **freeze**: bản gốc

# Những tùy chọn thực thi khác

- **Hệ thống Python không ngăn xếp** là một biến thể triển khai CPython tiêu chuẩn không lưu trạng thái trên ngăn xếp cuộc gọi ngôn ngữ C.
  - Dễ dàng hơn để chuyển sang kiến trúc ngăn xếp nhỏ
  - Tùy chọn đa xử lý hiệu quả
- Hệ thống Cython là một ngôn ngữ lai kết hợp mã Python với khả năng gọi hàm C và sử dụng khai báo kiểu C cho các biến, tham số và thuộc tính lớp.



# CÁCH PYTHON CHẠY CHƯƠNG TRÌNH

# Cách Python chạy một chương trình

1. The Interactive Prompt
2. System Command Lines and Files
3. Clicking File Icons
4. Module Imports and Reloads
5. Using exec to Run Module Files
6. The IDLE User Interface
7. Other IDEs
8. Other Launch Options

# The Interactive Prompt

- Nhập chương trình tại dòng lệnh tương tác của Python, đôi khi được gọi là dấu nhắc tương tác.
- Truy cập nó tùy theo hệ điều hành:
  - **Trên Windows:** python trong bảng điều khiển **DOS**
  - **Trên Unix, Linux và Mac OS X:** gõ lệnh này trong cửa sổ **shell** hoặc **terminal**.
  - Các hệ thống khác có thể sử dụng các thiết bị tương tự hoặc dành riêng cho nền tảng.

# Tại sao dùng Interactive Prompt?

- Mã được thực thi ngay lập tức, dấu nhắc tương tác là một nơi hoàn hảo để thử nghiệm với ngôn ngữ.
- Trình thông dịch tương tác là nơi lý tưởng để kiểm tra mã bạn đã viết trong tệp.

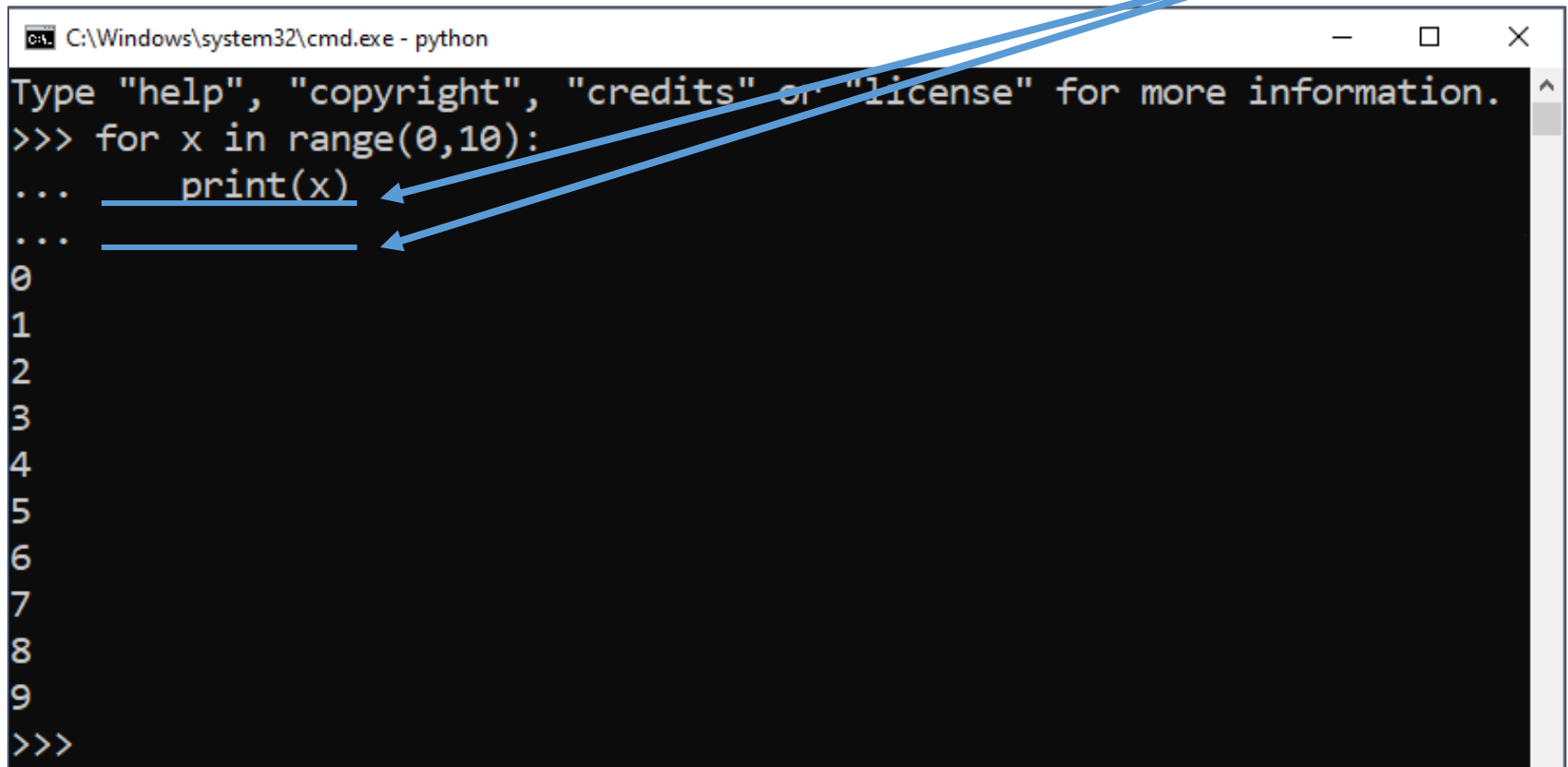
# Sử dụng Interactive Prompt

- Có một số lưu ý cần ghi nhớ
  - Chỉ nhập lệnh Python.
  - Câu lệnh print chỉ được yêu cầu trong các tệp
  - Không thụt lề tại dấu nhắc tương tác
  - Chú ý với các câu lệnh nhiều dòng
  - Kết thúc các câu lệnh nhiều dòng tại dấu nhắc tương tác bằng một dòng trống.
  - Dấu nhắc tương tác chạy một câu lệnh tại một thời điểm.

# Nhập câu lệnh nhiều dòng

- Nhấn phím **Enter hai lần** để kết thúc toàn bộ **câu lệnh nhiều dòng**.

Double Enter

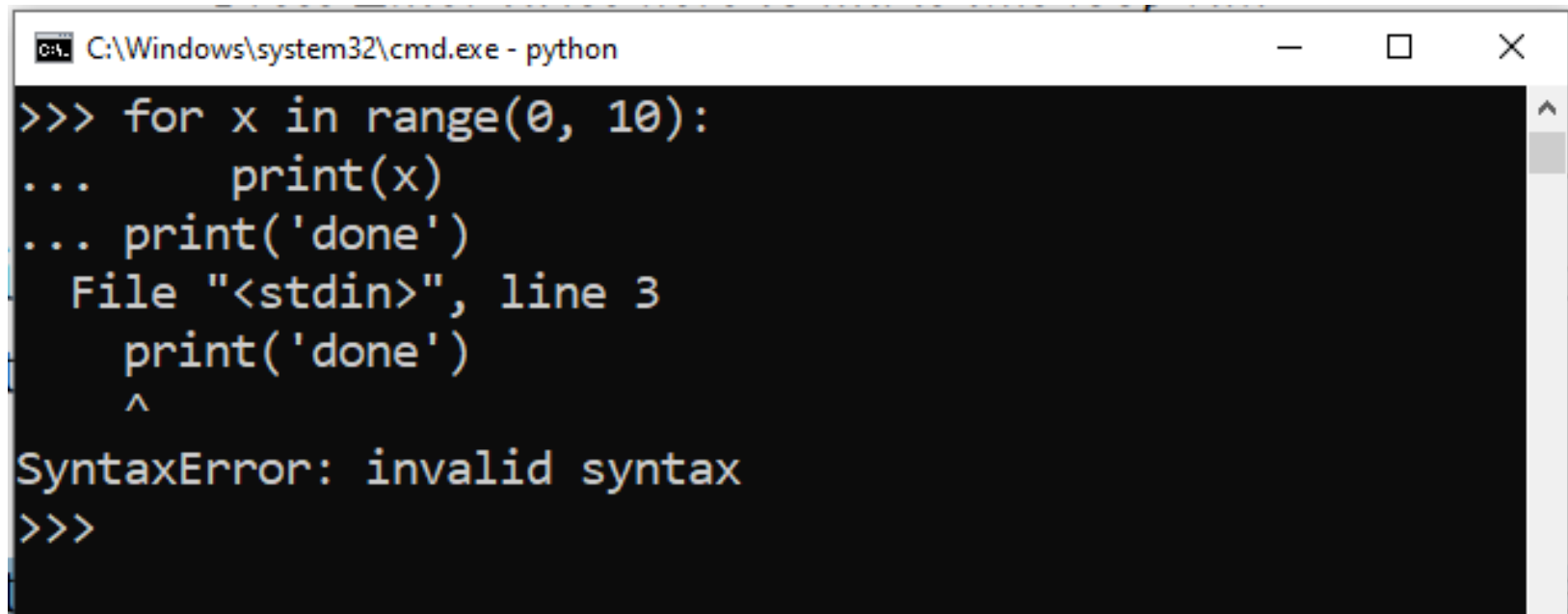


The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - python". The prompt displays the text "Type 'help', 'copyright', 'credits' or 'license' for more information." followed by a Python prompt ">>>". The user has entered a multi-line command: ">>> for x in range(0,10):" on the first line, "... print(x)" on the second line, and "... " on the third line. Two blue arrows point from the text "Double Enter" to the end of the second and third lines, indicating where to press the Enter key twice to complete the command. The output of the command is a list of numbers from 0 to 9, each on a new line. The prompt ">>>" is visible at the bottom of the window.

```
C:\Windows\system32\cmd.exe - python
Type "help", "copyright", "credits" or "license" for more information.
>>> for x in range(0,10):
...   print(x)
...
0
1
2
3
4
5
6
7
8
9
>>>
```

# Nhập câu lệnh nhiều dòng (tt)

- **Enter hai lần** để chạy vòng lặp hoặc câu lệnh nhiều dòng khác trước khi có thể nhập câu lệnh tiếp theo

A screenshot of a Windows command prompt window titled "C:\Windows\system32\cmd.exe - python". The window has a black background with white text. The user has entered a Python loop: >>> for x in range(0, 10):, followed by an indented print(x) on the next line, and another indented print('done') on the third line. The cursor is at the end of the third line. The prompt then shows the error message: File "<stdin>", line 3, print('done') with a caret under the opening quote, followed by "SyntaxError: invalid syntax" and a new prompt >>>.

```
C:\Windows\system32\cmd.exe - python
>>> for x in range(0, 10):
...     print(x)
...     print('done')
File "<stdin>", line 3
    print('done')
    ^
SyntaxError: invalid syntax
>>>
```

# System Command Lines and Files

- Một **nhược điểm** lớn của **interactive prompt**
  - các chương trình sẽ biến mất ngay sau khi trình thông dịch Python thực thi chúng.
  - mã **interactive prompt** không bao giờ được lưu trữ trong tệp
- Để lưu vĩnh viễn các chương trình, cần viết mã của mình vào các tệp, thường được gọi là mô-đun.
- Các tệp mô-đun được chạy trực tiếp.



# A First Script

script1.py

*# A first Python script*

Imports a  
Python module

`import sys`

Uses a variable  
named **x**

`print(sys.platform)`

`print(2 ** 100)`

`x = 'Spam!'`

`print(x * 8)`

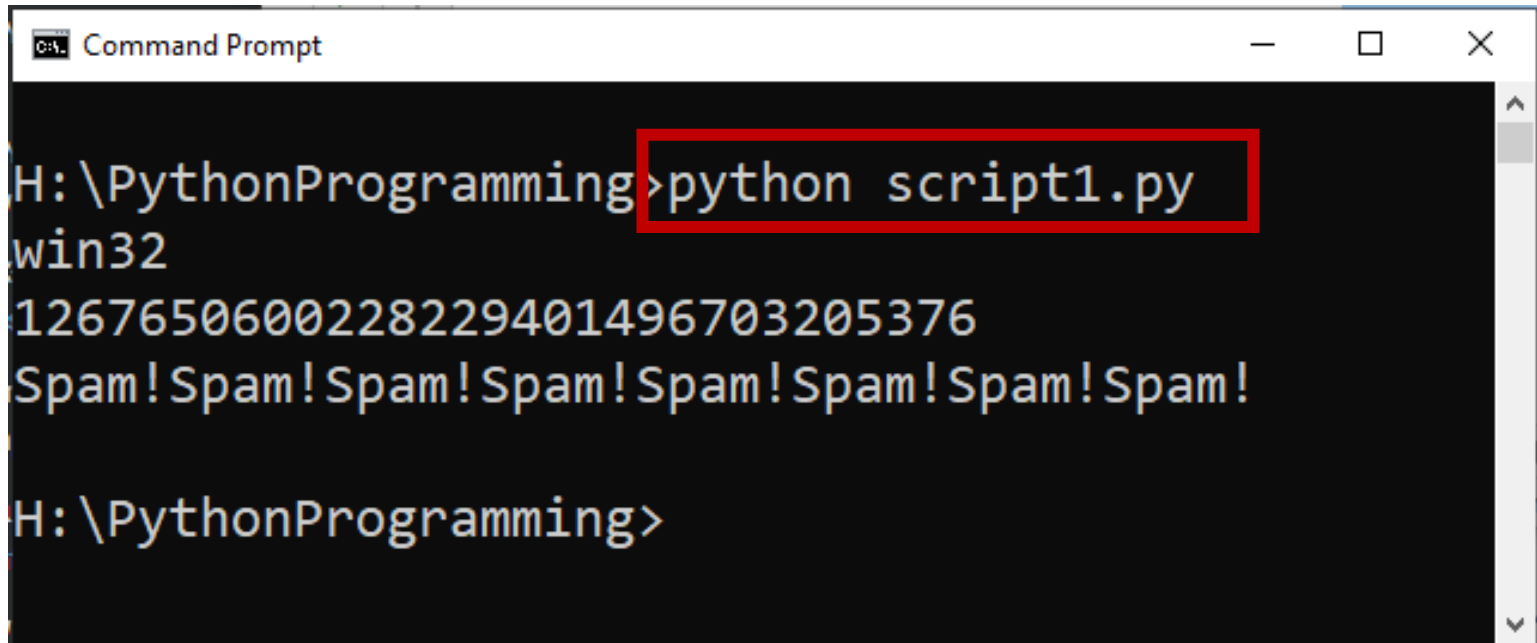
Runs three  
print function calls



# Chạy tệp bằng dòng lệnh

- Chạy script1.py bằng cách liệt kê tên tệp đầy đủ của nó làm đối số đầu tiên cho lệnh python, được nhập tại system shell prompt:

**% python full\_file\_name.py**

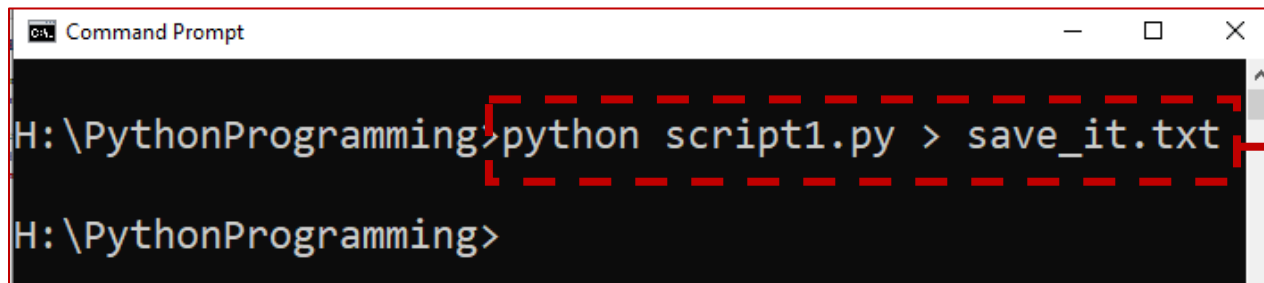


```
Command Prompt
H:\PythonProgramming>python script1.py
win32
1267650600228229401496703205376
Spam! Spam! Spam! Spam! Spam! Spam! Spam! Spam!
H:\PythonProgramming>
```

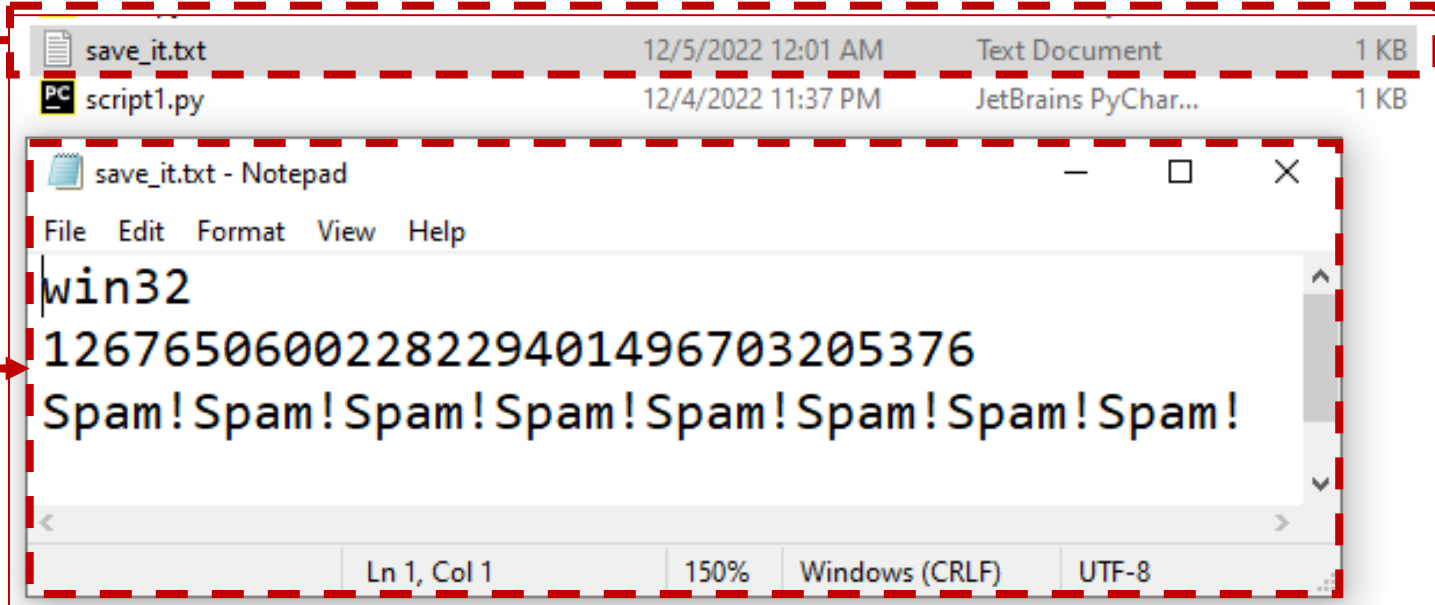
# Chạy tệp bằng dòng lệnh (tt)

- Route the output of a Python script to a file to save it for later use or inspection by using special shell syntax

**% python script1.py > saveit.txt**



```
Command Prompt
H:\PythonProgramming>python script1.py > save_it.txt
H:\PythonProgramming>
```

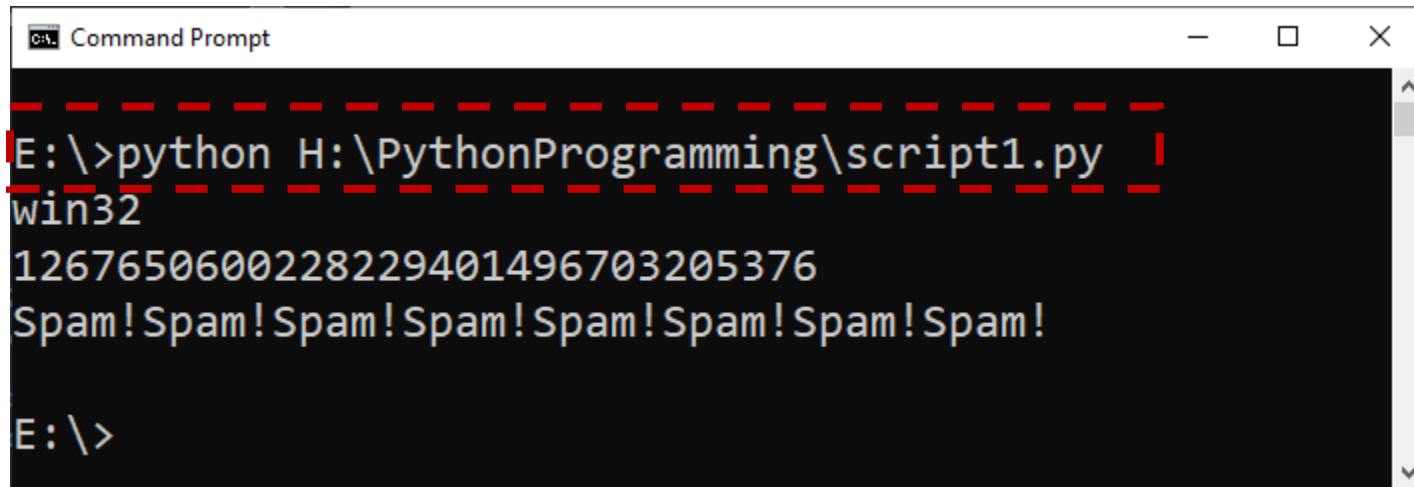


File Name	Modified	Type	Size
save_it.txt	12/5/2022 12:01 AM	Text Document	1 KB
script1.py	12/4/2022 11:37 PM	JetBrains PyChar...	1 KB

```
save_it.txt - Notepad
File Edit Format View Help
win32
1267650600228229401496703205376
Spam! Spam! Spam! Spam! Spam! Spam! Spam! Spam!
```

# Chạy tệp bằng dòng lệnh (tt)

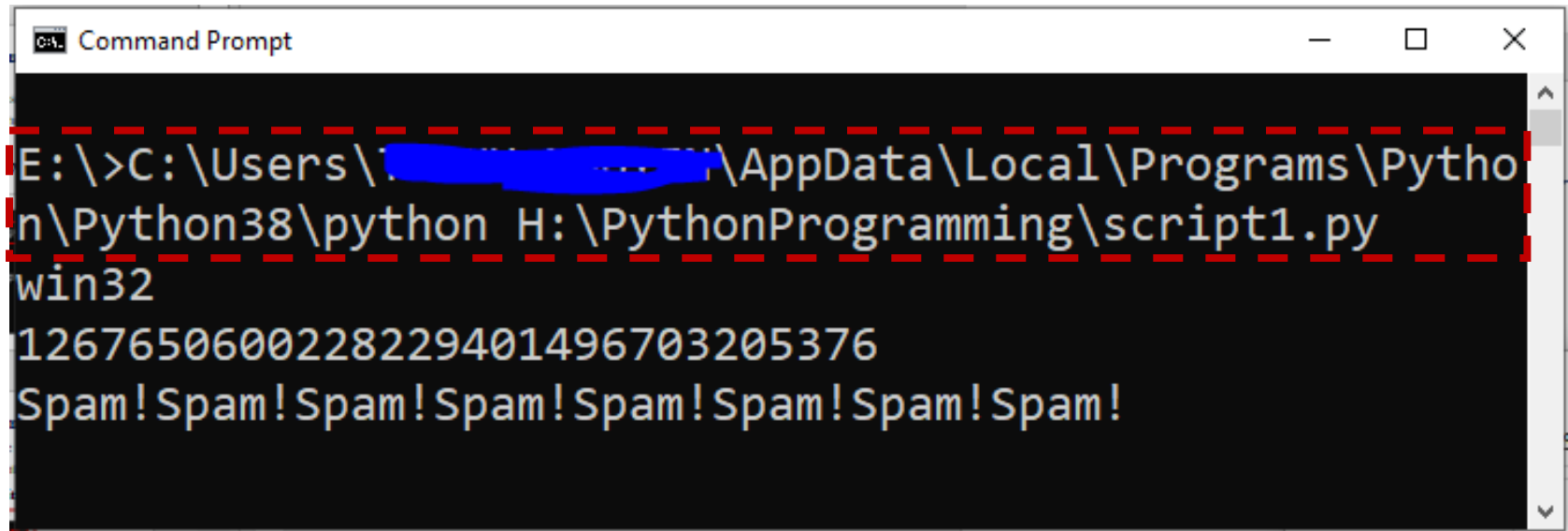
Cung cấp **đường dẫn đầy đủ đến tệp script** nếu nó nằm trong một thư mục khác với thư mục mà bạn đang làm việc



```
Command Prompt
E:\>python H:\PythonProgramming\script1.py
win32
1267650600228229401496703205376
Spam!Spam!Spam!Spam!Spam!Spam!Spam!Spam!
E:\>
```

# Chạy tệp bằng dòng lệnh (tt)

Nhập đường dẫn đầy đủ tới Python nếu chưa đặt biến môi trường PATH



```
Command Prompt
E:\>C:\Users\ [redacted] \AppData\Local\Programs\Python\Python38\python H:\PythonProgramming\script1.py
win32
1267650600228229401496703205376
Spam!Spam!Spam!Spam!Spam!Spam!Spam!Spam!
```

# Sử dụng dòng lệnh và tập tin

Vài gợi ý về những cái bẫy phổ biến dành cho người mới bắt đầu:

- Cảnh giác với tiện ích mở rộng tự động trên Windows
- Sử dụng phần mở rộng tệp và đường dẫn thư mục tại interactive prompt, nhưng không phải để nhập.
- Sử dụng câu lệnh in trong tệp.

# Tập lệnh thực thi Unix (#!)

- Các tập lệnh thực thi kiểu Unix chỉ là các tệp văn bản bình thường chứa các câu lệnh Python, nhưng có hai thuộc tính đặc biệt:
  - Dòng đầu tiên của họ là đặc biệt: `#!` (hash bang)
  - Chúng thường có đặc quyền thực thi.

```
#!/usr/local/bin/python  
print('The Bright Side ' + 'of Life...')
```

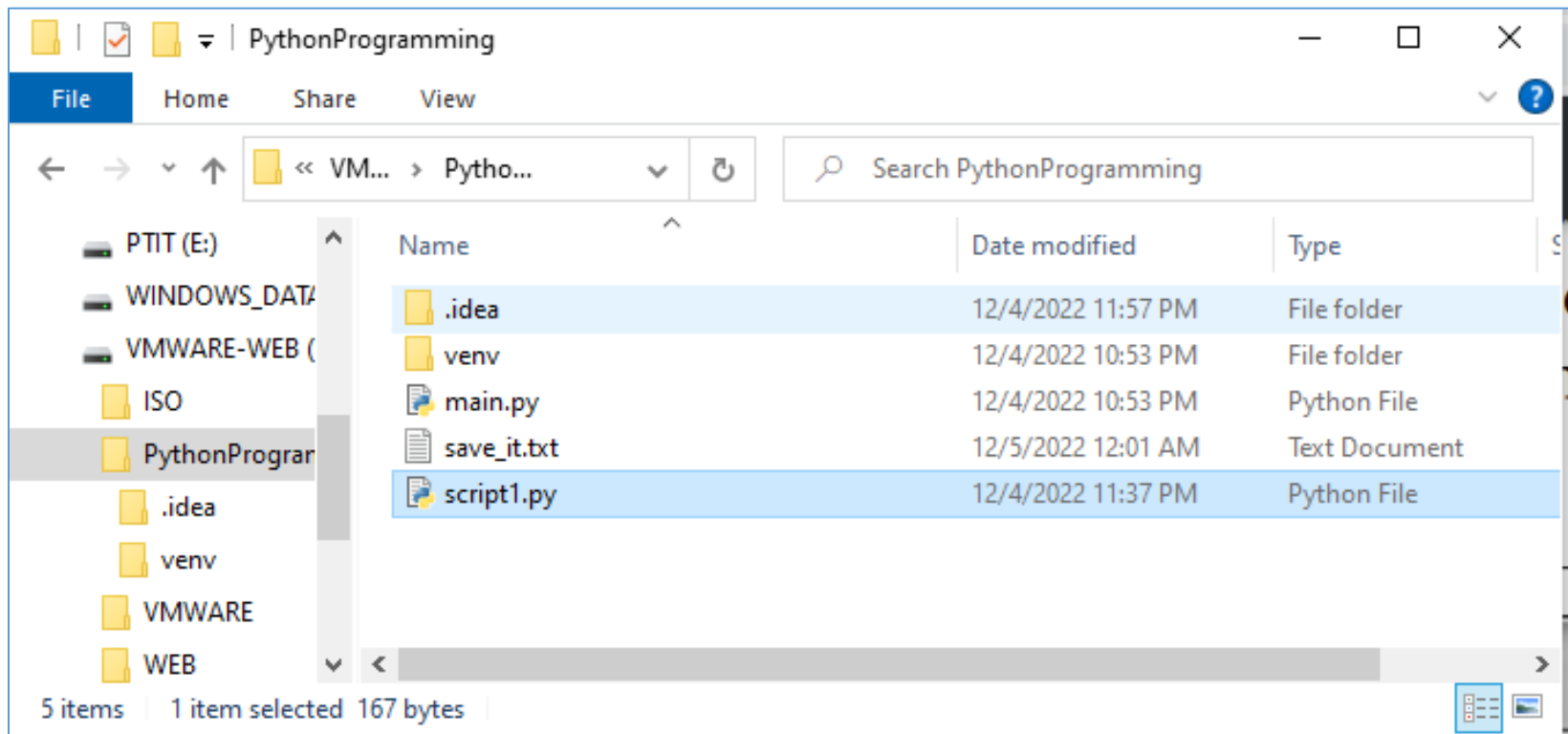


# Nhấp vào biểu tượng tệp tin

- Trên Windows, Registry giúp việc mở tệp bằng cách nhấp vào biểu tượng trở nên dễ dàng.
- Trên các hệ thống không phải Windows, có thể thực hiện thủ thuật tương tự.

# Nhấp vào biểu tượng tệp tin (tt)

- Để khởi chạy tệp ở đây, chỉ cần nhấp vào biểu tượng cho script1.py.



# Thủ thuật với input

- ❑ Nếu một tập lệnh chỉ in và thoát
- ❑ Khó thấy đầu ra của mình.
- ❑ Thực hiện một cuộc lệnh input ở cuối tập tin để giữ cửa sổ dòng lệnh.

# Thủ thuật với input

*# A first Python script*

`import sys`      *# Load a library module*

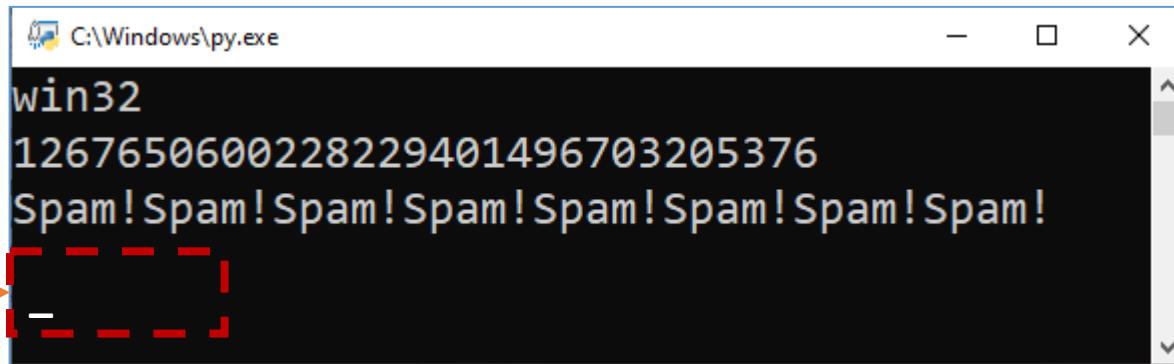
`print(sys.platform)`

`print(2 ** 100)`    *# Raise 2 to a power*

`x = 'Spam!'`

`print(x * 8)`      *# String repetition*

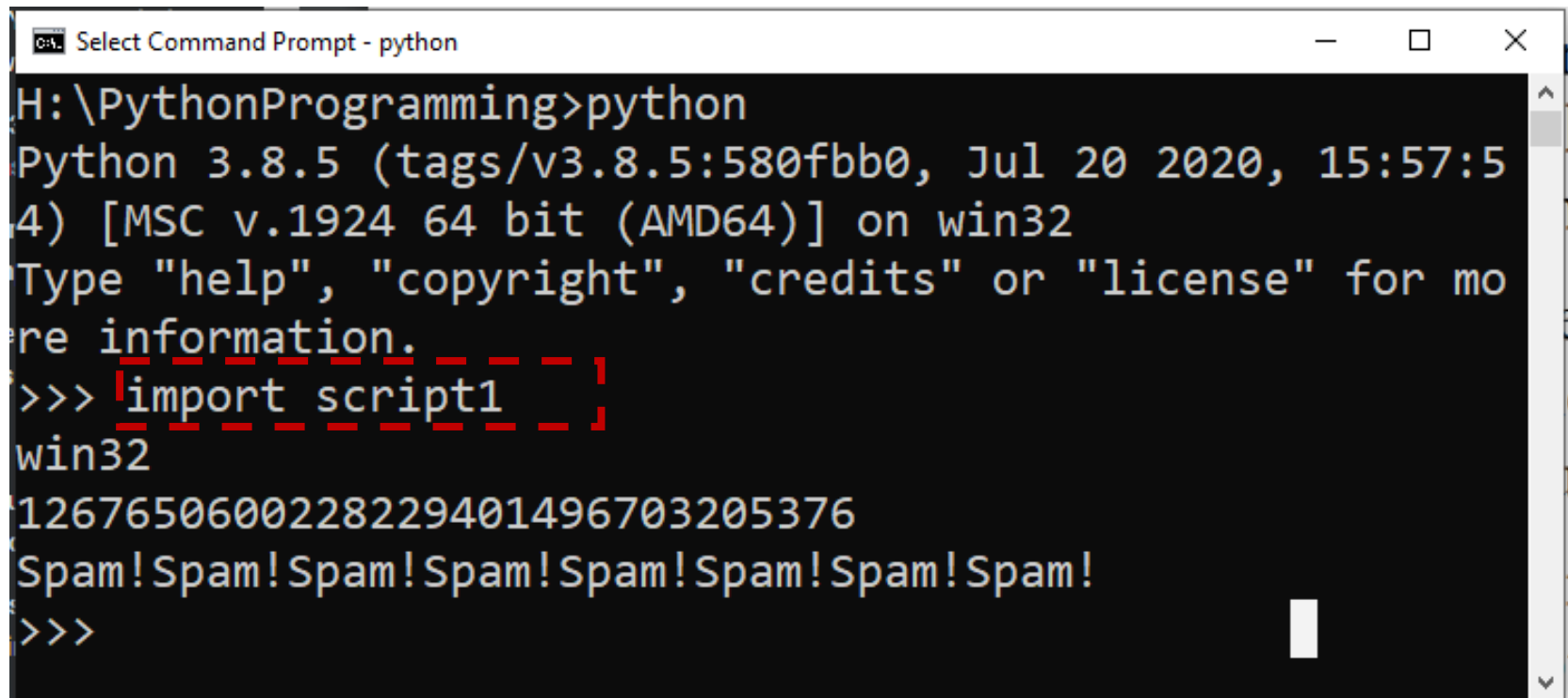
`input()`      *# <== ADDED*



```
C:\Windows\py.exe
win32
1267650600228229401496703205376
Spam!Spam!Spam!Spam!Spam!Spam!Spam!Spam!
_
```

# Module Imports và Reloads

- Chạy tệp script1.py bạn đã tạo trước đó bằng lệnh **import**



```
Select Command Prompt - python
H:\PythonProgramming>python
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import script1
win32
1267650600228229401496703205376
Spam! Spam! Spam! Spam! Spam! Spam! Spam! Spam!
>>>
```

# Module Imports và Reloads (tt)

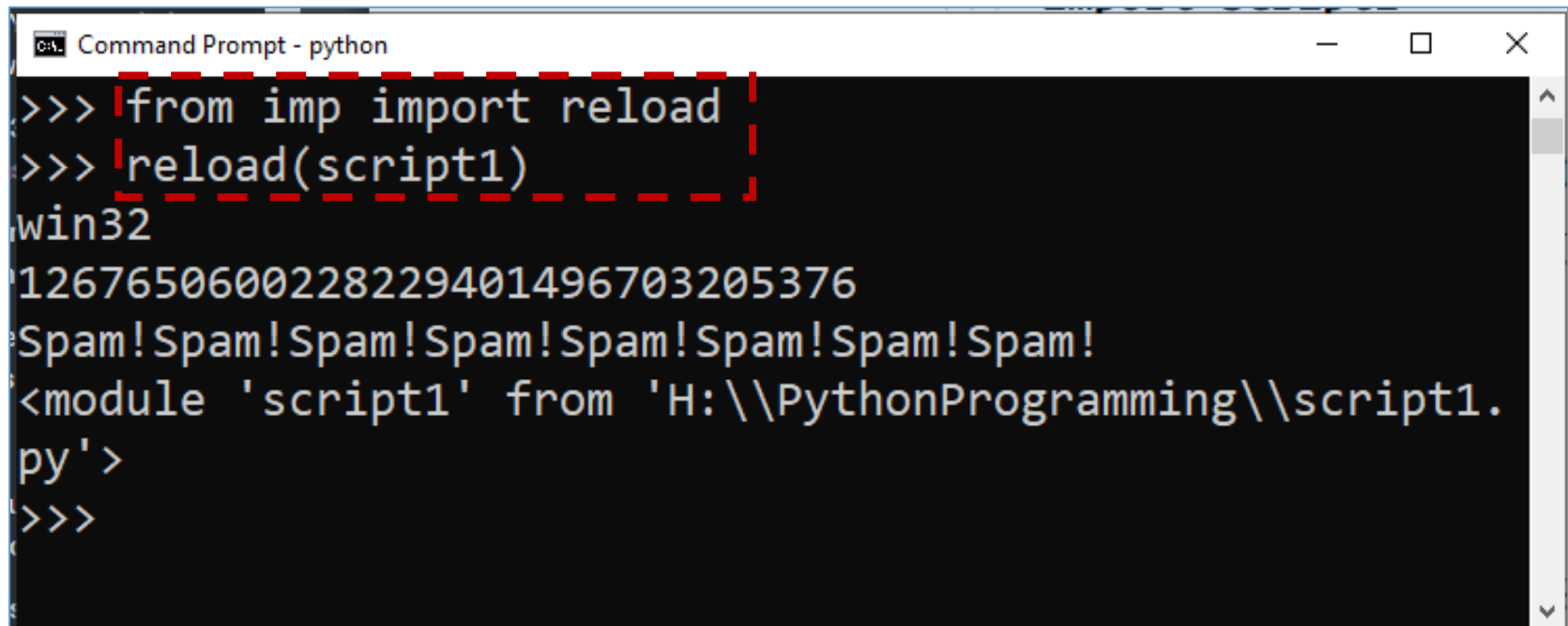
- Sau lần nhập đầu tiên, các lần nhập sau không thấy thực thi điều gì



```
Command Prompt - python
4) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import script1
win32
1267650600228229401496703205376
Spam!Spam!Spam!Spam!Spam!Spam!Spam!Spam!
>>> import script1
>>> import script1
>>>
```

# Module Imports và Reloads (tt)

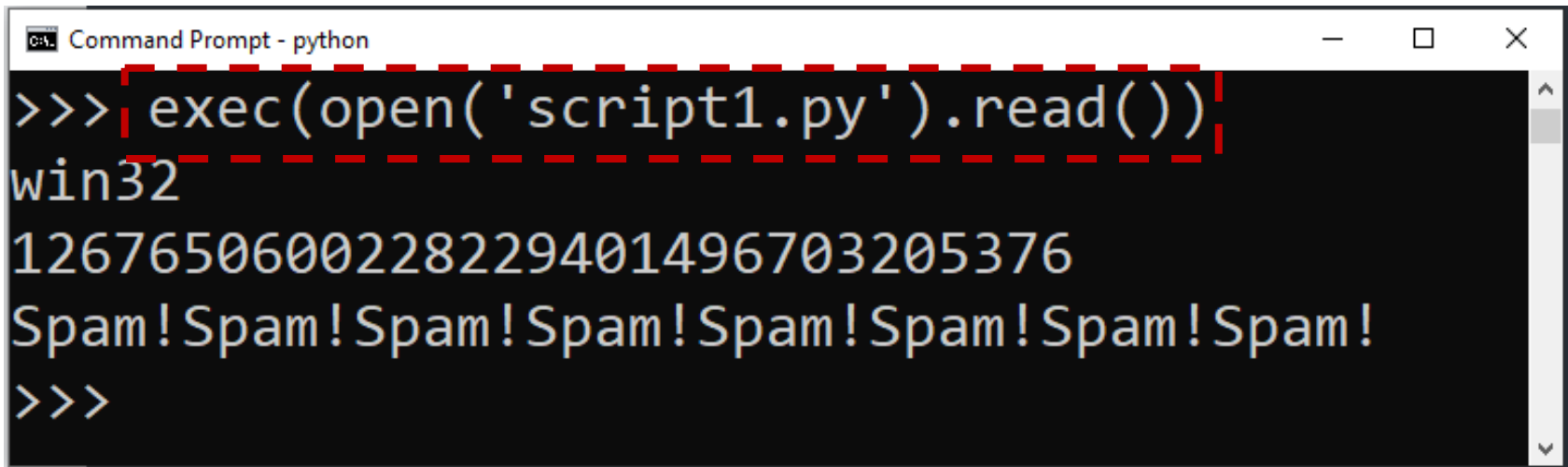
Hàm **reload** mong đợi tên của một đối tượng mô-đun đã được tải



```
Command Prompt - python
>>> from imp import reload
>>> reload(script1)
win32
1267650600228229401496703205376
Spam!Spam!Spam!Spam!Spam!Spam!Spam!Spam!
<module 'script1' from 'H:\\PythonProgramming\\script1.
py'>
>>>
```

# Sử dụng exec để chạy tệp mô-đun

- Lệnh gọi hàm tích hợp **exec(open('module.py').read())** là một cách khác để khởi chạy tệp từ lời nhắc tương tác mà không cần phải nhập và tải lại sau đó.

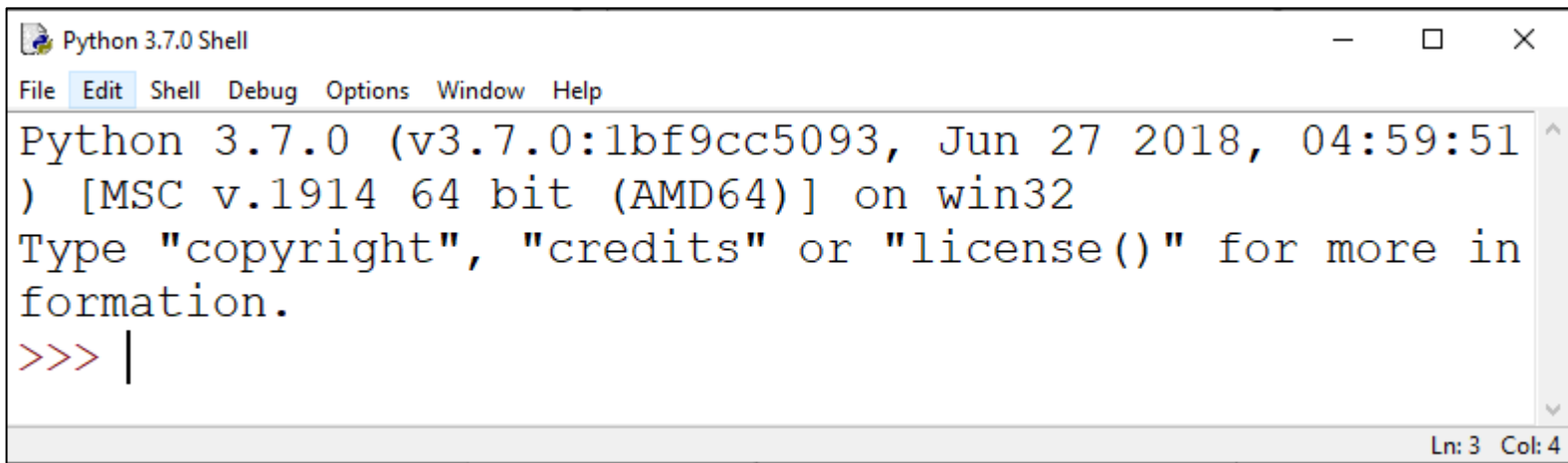


```
C:\> Command Prompt - python
>>> exec(open('script1.py').read())
win32
1267650600228229401496703205376
Spam!Spam!Spam!Spam!Spam!Spam!Spam!Spam!
>>>
```



# Giao diện người dùng IDLE

- IDLE là một GUI cho phép bạn:
  - chỉnh sửa,
  - chạy,
  - duyệt và
  - gỡ lỗi chương trình Python,
- tất cả từ một giao diện duy nhất.



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51)
[MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more in
formation.
>>> |
```

Ln: 3 Col: 4

# Sử dụng IDLE

- Danh sách các vấn đề mà người mới bắt đầu IDLE nên lưu ý:
  - Bạn phải thêm “.py” một cách rõ ràng khi lưu tệp của mình.
  - Chạy tập lệnh bằng cách chọn Run → Run Module trong cửa sổ chỉnh sửa văn bản, không phải bằng cách nhập và tải lại tương tác.
  - Chỉ cần reload lại các mô-đun đang được thử nghiệm tương tác.
  - Có thể tùy chỉnh IDLE.
  - Hiện tại không có tùy chọn xóa màn hình trong IDLE.
  - GUI tkinter và các chương trình về thread có thể không hoạt động tốt với IDLE.

# Các IDE khác

- **Eclipse and PyDev:**

- Eclipse is an advanced open source IDE GUI.
- supports Python development when install the PyDev

- **Komodo:**

- A full-featured development environment GUI for Python
- including project files, source-control integration, regular-expression debugging, and a drag-and-drop **GUI builder**

- **NetBeans IDE for Python**

- **PyCharm**