# Chương 0. Giới thiệu môn học Lập trình Python

NGUYỄN HOÀNG ANH

CNTT1. PTIT

# Thông tin môn học

Thông tin môn:

- Tên: Lập trình Python (Python Programming)
- Số tín chỉ: 3

# Yêu cầu với môn học

- Mỗi chương đều có bài tập và yêu cầu sinh viên hoàn thành trước khi sang chương mới
- Thiếu một điểm thành phần (bài tập, bài kiểm tra giữa kỳ), hoặc nghỉ quá 20% tổng số giờ của môn học, không được thi hết môn.

# Phương pháp đánh giá sinh viên

| Hình thức kiểm tra | Tỷ lệ đánh giá | Đặc điểm đánh giá |
|---|---|---|
| - Đi học đầy đủ (trong lớp gây ảnh hưởng đến người khác, mỗi lần nhắc nhở trừ một điểm, mỗi buổi nghỉ học trừ một điểm)<br>- Tích cực thảo luận (không phát biểu buổi nào sẽ được 0 điểm, phát biểu 1 buổi sẽ được 4 điểm, sau đó số buổi học có phát biểu tăng lên 1 thì điểm tăng lên 1) | 10 % | Cá nhân |
| - Trung bình kiểm tra | 10% | Cá nhân |
| - Trung bình các điểm bài tập lớn | 20% | Cá nhân/Nhóm |
| - Kiểm tra cuối kỳ | 60% | Cá nhân |

# Tại sao lại học Python?

# Tại sao lại học Python?

- Python là một mã nguồn mở di động miễn phí mạnh mẽ và là ngôn ngữ lập trình dựa trên kịch bản dễ học dễ nhớ.
- Python được sử dụng để phát triển các ứng dụng phía máy chủ cũng như để phát triển những ứng dụng độc lập.

# Tại sao lại học Python? (1)

- Ngôn ngữ lập trình Python tập trung vào khả năng đọc. Khi có thể đọc được, mã nguồn được viết bằng Python có thể sử dụng lại, có thể bảo trì và có chất lượng cao hơn.
- Nhiều cơ chế tích hợp cho phép chúng ta phát triển mã có thể dễ dàng giao tiếp với các phần khác của ứng dụng, ngay cả khi được viết bằng ngôn ngữ lập trình khác, chẳng hạn như C, C ++, Java và C #.
- Python được biết đến với hiệu suất của nó. Mã được viết bằng Python được rút gọn hơn so với mã tương đương được viết bằng Java hoặc C++. Ngoài ra, chu trình phát triển Python đơn giản hơn. Không cần bất kỳ giai đoạn biên dịch và liên kết dài dòng nào.

# Tại sao lại học Python? (2)

- Python có một bộ sưu tập lớn các chức năng sẵn sàng để sử dụng, được gọi là thư viện chuẩn. Thư viện đó có thể được mở rộng bằng cách thêm nhiều thư viện hơn, cũng như các thư viện được phát triển bởi các nhà phát triển bên thứ ba.
- Nhiều công cụ phát triển của bên thứ ba cho ngôn ngữ lập trình Python cho phép chúng ta sử dụng ngôn ngữ đó cho các tác vụ khác nhau, chẳng hạn như phát triển trang web, phát triển trò chơi, lập trình tương đương Matlab và các công việc khác.
- Mã được viết bằng Python thường không yêu cầu bất kỳ thay đổi nào để thực thi trên một nền tảng máy tính khác. Chuyển từ nền tảng này sang nền tảng khác là chuyển tiếp thẳng.

# Tại sao lại học Python? (3)

- Python hỗ trợ rất tốt cho hầu hết các cơ chế lập trình hướng đối tượng phổ biến.
- Python là một ngôn ngữ lập trình phổ biến và thú vị, nó đã được sử dụng bởi hơn 1 triệu nhà phát triển từ khắp nơi trên thế giới (2018). Python được sử dụng bởi một số lượng lớn ngày càng tăng của các công ty lớn, chẳng hạn như Google và những người khác.
- Python miễn phí để sử dụng và phân phối.
- Quản lý bộ nhớ trong Python là tự động. Bộ gom rác theo dõi tất cả các việc cấp phát bộ nhớ.

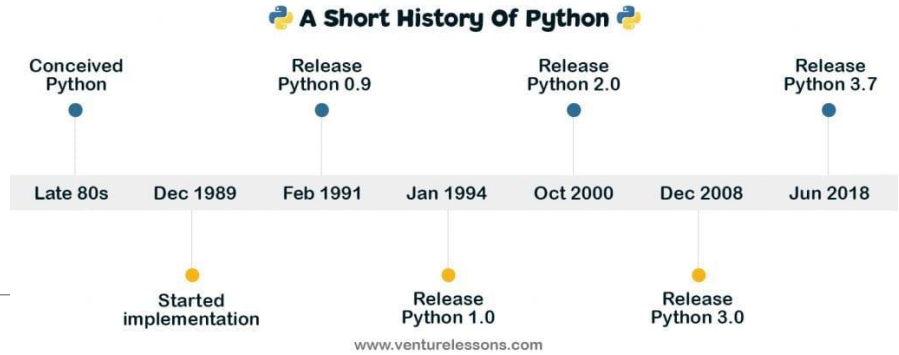
# Tại sao lại học Python? (3)

- Python là một ngôn ngữ lập trình kiểu động. Nó tiếp tục theo dõi tất cả các đối tượng mà chương trình sử dụng khi nó thực thi. Không cần khai báo các biến có kiểu cụ thể và kích thước cụ thể.

# Lịch sử Python

Nguồn: venturelessons.com

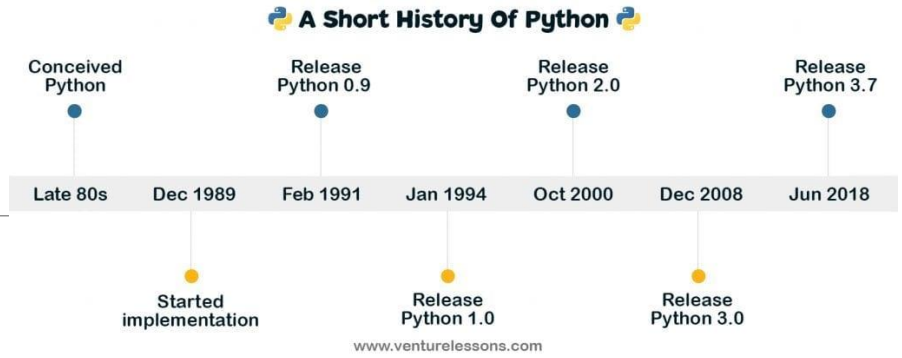
# Lịch sử Python




- Python lần đầu tiên được giới thiệu vào năm 1989 bởi Guido van Rossum tại CWI như một sự kế thừa cho ngôn ngữ lập trình ABC.
- Guido Van Rossum đã xuất bản phiên bản đầu tiên của mã Python (phiên bản 0.9.0) tại alt.sources vào tháng 2 năm 1991. Bản phát hành này đã bao gồm xử lý ngoại lệ, các hàm và các kiểu dữ liệu cốt lõi của list, dict, str và các loại khác. Nó cũng hướng đối tượng và có một hệ thống mô-đun.
- Phiên bản Python 1.0 được phát hành vào tháng 1 năm 1994. Các tính năng mới chính bao gồm trong phiên bản này là các công cụ lập trình chức năng lambda, bản đồ, bộ lọc và giảm thiểu.

# Lịch sử Python

Nguồn: venturelessons.com




- Python 2.0 được phát hành vào tháng 10 năm 2000. Bản phát hành này đã giới thiệu một bộ thu gom rác và được xây dựng để hỗ trợ bảng mã Unicode.
- Python 3.0 được phát hành vào tháng 12 năm 2008. Những thay đổi trong phiên bản này lớn đến nỗi nó bao gồm một công cụ giúp chuyển đổi mã Python được viết trong phiên bản trước thành một phiên bản tương thích với Python 3.0.

# Các định luật của Python

1. Beautiful is better than ugly.
2. Explicit is better than implicit.
3. Simple is better than complex.
4. Complex is better than complicated.
5. Flat is better than nested.
6. Sparse is better than dense.
7. Readability counts.
8. Special cases aren't special enough to break the rules.
9. Although practicality beats purity.

# Các định luật của Python (2)

10. Errors should never pass silently.
11. Unless explicitly silenced.
12. In the face of ambiguity, refuse the temptation to guess.
13. There should be one -- and preferably only one -- obvious way to do it.
14. Although that way may not be obvious at first unless you're Dutch.
15. Now is better than never.
16. Although never is often better than *right* now.
17. If the implementation is hard to explain, it's a bad idea.
18. If the implementation is easy to explain, it may be a good idea.
19. Namespaces are one honking great idea -- let's do more of those!

# Tài liệu tham khảo

***Học liệu bắt buộc***

- Eric Matthes. Python crash course, 2nd Edition: A hands-on, project-based introduction to programming, No Starch Press, 2019.

***Học liệu tham khảo***

- Allen B. Downey, Think Python: How to Think Like a Computer Scientist, O'Reilly, 2015
- Zed A. Shaw, Learn Python 3 the Hard Way, Addison-Wesley, 2016

# Trao đổi

Hình thức liên lạc

Chia nhóm

# Hỏi đáp !!!