**TÌM HIỂU VỀ PYTHON**

1. Tìm hiểu lý thuyết
2. Định ngĩa:

* Python là 1 ngôn ngữ lập trình bậc cao cho các mục đích lập trình đa năng.
* Được tạo bởi Guido và Rossum và lần đầu ra mắt năm 1991
* Dễ đọc, dễ học và dễ nhớ; hình thức sáng sủa, cấu trúc rõ ràng
* Cấu trúc ngắn gọn -> viết mã lệnh với số lầ gõ phím tối thiểu

1. Tính năng chính của Python

* Ngôn ngữ lập trình đơn giản, dễ học
* Miễn phí, mã nguồn mở
* Khả năng di chuyển
* Khả năng mở rộng và có thể nhúng
* Ngôn ngữ thông dịch cấp cao
* Thư viện tiêu chuẩ lớn để giải quyết những tác vụ phổ biến
* Hướng đối tượng
* Có thể mở rộng và cải tiến

1. Ứng dụng:

* Phân tích dữ liệu và học máy
* Phát triển web
* Tự động hóa và phát triển phần mềm
* Tạo nguyên mẫu phần mềm
* Ứng dụng trong khoa học và tính toán
* Viết tool để tự động hóa công việc
* Làm dự án Blockchain
* Làm game
* Machine Learning

1. Cú pháp cơ bản
2. Định danh và từ khóa trong Python

* Định danh chỉ chứa ký tự số, ký tự chữ cái và dấu gạch chân.
* Có thể đặt định danh bằng tên tiếng Việt có dấu
* Định danh không được bắt đầu bằng chữ số
* Định danh không trùng với các từ khóa
* Định danh phân biệt chữ hoa/ thường

1. Dòng lệnh

* Mặc định mỗi lệnh được viết trên 1 dòng -> Các lệnh không bắt buộc phải dùng dấu báo kết thúc (;).
* Ký tự \ dùng để nối dòng
* Các lệnh chứa các cặp dấu [], {}, () có thể viết trên nhiều dòng mà không cần ký tự nối dòng

1. Ghi chú: Đánh dấu bắt đầu bằng ký tự #
2. Thụt đầu dòng:

* Dấu cách ở đầu dòng lệnh có vai trò quan trọng giúp tạo ra cấu trúc code (thành các khối code)
* Không được tùy tiện thụt đầu dòng vì có thể sai logic hoặc sai cú pháp
* Chỉ thụt đầu dòng bằng dấu tab khi cần viết khối code
* Dấu cách ở các vị trí khác không có ý nghĩa gì trong Python

1. Khối code trong Python:

* 1 khối code thì các lệnh sẽ cùng số thụt đầu dòng.

1. Xuất nhập dữ liệu với console

* print() – xuất dữ liệu ra console
* input() – nhập dữ liệu từ console

1. Biến, phép gán và các kiểu dữ liệu cơ sở trong Python
2. Biến

* Chỉ là 1 tên gọi được sử dụng để truy xuất giá trị
* Bắt buộc phải tạo ra cùng phép gán
* Giá trị của biến trong bộ nhớ được tự động “thu dọn” khi không sử dụng đến nhờ tiến trình Garbage Collection
* Kiểu đặt tên biến:

+ Ký tự đầu viết thường; các chữ cái trong từ viết thường; nếu tên có nhiều từ thì viết hoa chữ cái đầu mỗi từ tiếp theo.

+ Tất cả ký tự đều viết thường, nếu tên có nhiều từ thì phân tách bằng dấu gạch chân \_.

1. Phép gán

* Khai báo biến luôn phải đi kèm gán giá trị
* Không cần chỉ định kiểu dữ liệu
* Không cần từ khóa nào khi khai báo biến

1. Các kiểu dữ liệu cơ bản trong Python: số, chuỗi, danh sách, tuple, từ điển, boolean.
   1. Kiểu số
2. Kiểu số nguyên int

* Khi biểu diễn số ở cơ số 8 dùng tiếp tố 0o hoặc 0O
* Khi biểu diễn số ở hệ 16 dùng tiếp tố 0x hoặc 0X
* Cho phép dùng ký tự gạch chân để nhóm các chữ số
* Kiểu boolean là 1 kiểu con của kiểu số nguyên: 0 tương ứng với False, mọi giá trị nguyên khác tương ứng với True
* int() - biến đổi từ các kiểu dữ liệu khác về số nguyên

1. Kiểu số thực float

* Cần đặt 1 dấu chấm thập phân
* Cho phép dùng ký tự gạch chân để nhóm các chữ số
* float() - biến đổi từ các kiểu dữ liệu khác về kiểu float

1. Kiểu số phức complex

* Chứa các ký tự j (hoặc J) để biểu diễn phần ảo và phải đi kèm số
* real và imag – trích giá trị phần nguyên và phần ảo
  1. Kiểu chuỗi

1. Kiểu string
2. Escape character

* \r – đưa con trỏ văn bản console về đầu dòng
* \t – chèn dấu tab
* \n – bắt đầu 1 dòng mới

1. Các phép toán trên kiểu string

|  |  |
| --- | --- |
| **Phép toán** | **Ý nghĩa** |
| + | Ghép xâu |
| \* | lặp xâu |
| [i] | Phép cắt, lấy ký tự ở vị trí i |
| [i1:i2] | phép cắt đoạn từ vị trí i1 đến i2; i1 hoặc i2 âm thì tính từ cuối chuỗi |
| in | kiểm tra thành viên |
| not in | kiểm tra thành viên (phủ định) |
| r/R | chuỗi thô |
| % | Định dạng chuỗi |
| == | So sánh xâu (bằng) |
| != | So sánh xâu (khác) |

1. Chuỗi định dạng
2. Các phương thức xử lý chuỗi
   1. Kiểu dữ liệu bool

* Cho phép chuyển đổi giữa các kiểu dữ liệu khác về bool qua hàm bool() theo quy tắc:

+ giá trị kiểu số thành giá trị True nếu số đó khác 0, False nếu số đó bằng 0.

+ giá trị kiểu chuỗi thành giá trị False nếu là chuỗi rỗng, True nếu chuỗi có từ 1 ký tự trở lên.

* Các phép toán trên kiểu bool: and, or, not, == và !=
  1. Kiểu danh sách (list)

1. Kiểu list
   * Có khả năng chứa 1 danh sách các phần tử (có thể khác kiểu)
   * Phải viết trong cặp dấu ngoặc vuông, các phần từ tách nhau bởi dấu phẩy.
   * Các phần từ được đánh chỉ số (index). Chỉ số bắt đầu từ 0.
   * Danh sách lồng nhau: Danh sách chứa danh sách khác.
2. Truy xuất phần tử: thông qua phép toán cắt hoặc cắt đoạn
3. Thêm mới, cập nhật, xóa
   * Cập nhật: gán giá trị mới cho phần tử được chọn.
   * Xóa: dùng lệnh del trước phần tử cần xóa
   * Thêm:

+ Sử dụng phép cộng gán +=

+ append() – thêm 1 phần tử

+ extend() – thêm nhiều phần tử

* + Một số thao tác khác:

+ Duyệt danh sách: dùng vòng for

+ Phép toán: ghép danh sách (+), lặp danh sách (\*), kiểm tra thành viên (in)

+ len(): trả về số phần tử của danh sách.

+ max(): trả về phần tử lớn nhất trong danh sách.

+ min(): trả về phần tử nhỏ nhất trong danh sách.

* + Các phương thức:

+ count(…): đếm xem giá trị xuất hiện bao nhiêu lần

+ index(…): xác định chỉ số của phần tử

+ pop(): lấy ra phần tử cuối và loại bỏ nó khỏi danh sách

+ insert(x, …): chèn phần tử vào vị trí chỉ định

+ remove(…): loại bỏ phần tử khỏi danh sách

+ reverse(): đảo ngược vị trí các phần tử

+ sort(): sắp xếp danh sách

* 1. Kiểu dữ liệu tuple

1. Kiểu dữ liệu tuple
   * Có thể chứa nhiều giá trị viết trong cặp dấu ngoặc tròn (). Mỗi giá trị viết tách nhau bởi dấu phẩy.
   * Tuple lồng nhau: phần tử của 1 tuple có thể là 1 tuple khác
   * Tính bất biến -> không thể cập nhật, không thể xóa một phần tuple mà chỉ có thể xóa toàn bộ tuple.
   * Có thể truy xuất các phần tử của tuple qua phép toán slice và range slice giống như đối với list.
2. Các phép toán và hàm: giống list
   * tuple(): chuyển đổi list về tuple
3. Một số cấu trúc trong Python
4. Cấu trúc điều kiện if

* Mệnh đề elif và else
* Từ khóa pass

1. Các cấu trúc lặp: for và while

* Hàm range() – trả về biến chứa 1 dãy các giá trị nằm trong 1 khoảng xác định theo quy luật.

+ range(stop): trả về dãy số [0, stop-1]

+ range(start, stop): trả về dãy số [start, stop-1]

+ range(start, stop, step): bước nhảy là step

* Có thể sử dụng vòng lặp for với các kiểu danh sách khác
* break – phá vỡ vòng lặp, kết thúc sớm
* continue – bỏ qua chu kỳ hiện tại, bắt đầu chu kỳ lặp mới

1. Function (hàm) trong Python
2. Xây dựng và sử dụng hàm

* Hàm được định nghĩa bằng từ khóa def, theo sau là tên hàm và cặp dấu () chứa các tham số.

1. Biến cục bộ và phạm vi của biến

* Biến toàn cục được khai báo bên ngoài hàm: sử dụng ở bất cứ chỗ nào sau vị trí nó được khai báo.
* Biến cục bộ được khai báo bên trong hàm: chỉ sử dụng bên trong hàm đó, hàm khác không nhìn thấy nó.

1. Hàm đệ quy
2. Module và package trong Python

* Module cho phép lưu trữ hàm (và code khác) trên các file riêng rẽ
* Package cho phép nhóm các module lại với nhau
* Sử dụng module với import… - tải toàn bộ module
* Sử dụng module với from … import … (chỉ sử dụng được các hàm khai báo sau từ khóa “import”)

1. Lập trình Python hướng đối tượng
2. Class, constructor
3. Khai báo class

Class tên\_class:

‘’’docstring’’’

1. Sử dụng class: tương tự sử dụng hàm
2. Các thành phần chính trong class:

* Constructor (hàm tạo): tạo các đặc tính thành viên
* Các attribute (biến)
* Các method (phương thức)
* Các property (thuộc tính)

1. Constructor trong Python:

\_\_\_init\_\_\_() là một hàm đặc biệt trong Python: hàm tạo

* Phải có ít nhất 1 tham số, self bắt buộc phải là tham số đầu tiên

1. Dữ liệu của class: instance và class attribute
2. Instance attribute

* Biến thành viên của class là instance attribute
* Được khai báo trong hàm tạo \_\_\_init\_\_\_()

1. Class attribute

* Là 1 biến gắn liền với chính class, có thể được truy xuất từ các object và có giá trị chung cho tất cả các object

1. Các loại phương thức trong Python class

* Method trong class Python không bắt buộc phải khai báo trong thân class. Method độc lập với class và object.

1. Instance method

* Là những phương thức có khả năng truy xuất trạng thái của object

1. Class method

* Là những phương thức đặc trưng cho class và gắn liền với class, thay vì gắn với object.
* Có nhiệm vụ xử lý dữ liệu lưu trong class attribute
* Để tạo ra class method từ một method thông thường, sử dụng decorator @classmethod
* Bắt buộc phải có 1 biến đặc biệt trong danh sách tham số: biến cls

1. Static method

* Là loại phương thức hoàn toàn tự do, không có gì ràng buộc về dữ liệu với class hay object.
* Để tạo ra static method sử dụng decorator @staticmethod

1. Thuộc tính trong Python
2. Mô hình getter/setter trong Python class

* Sử dụng 2 dấu gạch chân ở đầu cho biến private
* Ứng với mỗi biến private cần xây dựng 2 phương thức get/set để xuất/ nhập dữ liệu

1. Property trong Python, hàm property()

* 2 cách để tạo property: sử dụng hàm property() và sử dụng decorator @property

1. Kế thừa trong Python

* Cú pháp: class <lớp con>(<lớp cha>):

Vd: class Student(Person):

* Hỗ trợ đa kế thừa
* Cho phép class con được ghi đè phương thức từ class cha
* Ghi đè hàm tạo, bắt buộc phải gọi hàm tạo của lớp cha trong hàm tạo của lớp con qua phương thức super()
* Class con ko kế thừa được các thành viên private (sử dụng name mangling)