**Ngày 01 - Giới thiệu về Python**

Với phần giới thiệu về python, sách đã đưa ra lịch sử hình thành, ưu nhược điểm và ứng dụng của python.

Cách cài đặt môi trường, cách comment 1 (#) hay nhiều dòng (“”” “””) trong Python, câu lệnh install (pip install “thư viện cần cài đặt”).

Giáo trình đưa ra 1 số đoạn code thể hiện lệnh in và sử dụng thư viện turtle để vẽ.

**Ngày 02 - Yếu tố ngôn ngữ**

Giáo trình đưa ra hệ thống phần cứng máy tính bao gồm 5 phần: đơn vị số học, bộ điều khiển, bộ nhớ, thiết bị đầu vào và thiết bị đầu ra. Máy tính Von Neumann, các ý nghĩa của hệ nhị phân, thập phân, bát phân và thập lục phân.

Biến và kiểu: Biến là vật chứa để lưu trữ dữ liệu, giá trị của biến có thể được đọc và sửa đổi là cơ sở để tính toán. Các kiểu dữ liệu hay dùng: kiểu số nguyên (int), kiểu dấu phẩy động (kiểu số thực – float), kiểu chuỗi (“string” or ‘string’), kiểu Boolean, kiểu số phức (complex).

Đặt tên biến:

* Quy tắc bắt buộc:
* Tên biến bao gồm các chữ cái (ký tự Unicode, không bao gồm các ký tự đặc biệt), số và dấu gạch dưới. Số không thể bắt đầu.
* Phân biệt chữ hoa chữ thường (chữ hoa A và chữ thường a là hai biến khác nhau).
* Không xung đột với các từ khóa và các từ dành riêng cho hệ thống (như tên hàm, mô-đun, v.v.).
* PEP 8 yêu cầu:
* Đánh vần nó bằng chữ thường và kết nối nhiều từ bằng dấu gạch dưới.
* Các thuộc tính phiên bản được bảo vệ bắt đầu bằng một dấu gạch dưới duy nhất.
* Các thuộc tính cá thể riêng tư bắt đầu bằng hai dấu gạch dưới.

Sử dụng biến: Có thể sử dụng type các hàm để kiểm tra loại biến.

Các loại biến có thể được chuyển đổi bằng cách sử dụng các hàm dựng sẵn trong Python.

* int(): Chuyển đổi một giá trị số hoặc chuỗi thành số nguyên, có thể chỉ định cơ số.
* float(): Chuyển đổi một chuỗi thành số dấu phẩy động.
* str(): Chuyển đổi đối tượng đã chỉ định thành dạng chuỗi và có thể chỉ định mã hóa.
* chr(): Chuyển đổi số nguyên thành chuỗi (một ký tự) tương ứng với bảng mã.
* ord(): Chuyển đổi một chuỗi (ký tự) thành bảng mã tương ứng (số nguyên).

Toán tử: Python hỗ trợ nhiều loại toán tử. Thứ tự ưu tiên các toán tử được sắp xếp từ cao đến thấp theo bảng sau:

|  |  |
| --- | --- |
| Toán tử | Mô tả |
| [] [:] | chỉ số dưới, lát cắt |
| \*\* | mục lục |
| ~ + - | Phủ định theo bit, dấu |
| \* / % // | Nhân, chia, modulo, chia |
| + - | Cộng, trừ |
| >> << | di chuyển sang phải, di chuyển sang trái |
| & | Theo bit VÀ |
| ^ | | XOR theo bit, theo bit HOẶC |
| <= < > >= | Nhỏ hơn hoặc bằng, nhỏ hơn, lớn hơn, lớn hơn hoặc bằng |
| == != | bằng, không bằng |
| is is not | toán tử nhận dạng |
| in not in | nhà điều hành thành viên |
| not or and | Toán tử logic |
| = += -= \*= /= %= //= \*\*= &=` | =  ^=  >>=  <<=` |

Toán tử gán: Toán tử gán phải là toán tử phổ biến nhất, chức năng của nó là gán giá trị bên phải cho biến ở bên trái.

Toán tử so sánh: Toán tử so sánh ở một số nơi còn được gọi là toán tử quan hệ, bao gồm ==, !=, <, >, <=, >=. Các toán tử so sánh tạo ra các giá trị Boolean, is True hoặc is False.

Toán tử logic: Gồm 3 loại là and, or và not.

**Ngày 03 – Cấu trúc rẽ nhánh**

Ứng dụng của cấu trúc nhánh: Nhằm mục đích xét tối đa các trường hợp có thể xảy ra, tránh trường hợp bỏ xót trường hợp dẫn đến sai bài toán.

- Có 3 từ khóa có thể sử dụng trong cấu trúc rẽ nhánh: if, elif, else.

VD: a = float(input(‘Nhập số a’: ))

If a/3==0:

Print(a chia hết cho 3)

Else:

Print(a không chia hết cho 3)

- Không giống như các ngôn ngữ khác, Python không sử dụng dấu <> để xây dựng các khối mà sử dụng thụt lề để biểu thị cấu trúc phân cấp của mã if. (Nếu các câu lệnh sử dụng cùng 1 mức thụt lề thì chúng thuộc về cùng một khối mã). Không nên sử dụng phím tab hoặc đặt công cụ chỉnh sửa mã của mình để tự động thay đổi phím tab thành 4 khoảng trắng.

-Khi xây dựng nhiều nhánh, ta có thể sử dụng cấu trúc if….elif….else… hoặc cấu trúc if…else… lồng nhau.

**Ngày 04 - Cấu trúc vòng lặp**

Ứng dụng của cấu trúc vòng lặp: Khi gặp phải tình huống cần thực hiện lặp lại 1 hay nhiều lần thì ta sử dụng cấu trúc vòng lặp.

Có 2 cách để xây dựng cấu trúc vòng lặp trong python là: for-in và while.

1. Vòng lặp for-in

Nếu biết rõ số lần thực thi vòng lặp hoặc muốn lặp lại trong1 vùng chứa thì nên sử dụng vòng lặp for-in.

VD: Tính tổng từ 1 đến 100

Sum = 0

For x in range(101):

Sum += x

Print(sum)

Cách sử dụng range:

Range(101): Tạo ra các số nguyên từ 0 đến 100.

Range(1,101): Tạo ra các số nguyên từ 1 đến 100 (đánh dấu từ điểm bắt đầu và điểm kết thúc)

Range(1,101,2): Tạo ra các số lẻ đếm từ 1 đến 100 (số thứ 3 trong range dùng để chỉ bước nhảy trong dãy)

Range(100,0,-2): Tạo ra các số chẵn đếm từ 100 đến 1 (số thứ 3 trong range dùng để chỉ bước nhảy trong dãy, mỗi lần giảm đi 2 đơn vị)

Ta có thể lồng cấu trúc nhánh vào trong vòng lặp.

1. Vòng lặp while

Vòng lặp while được sử dụng khi ta không thể xác định được số vòng lặp 1 cách chính xác. Vòng lặp while thông qua 1 biểu thức có thể tạo ra hoặc chuyển đổi 1 giá trị.

Chỉ vòng lặp chứa break mới có thể kết thúc vòng lặp (cần lưu ý khi sử dụng vòng lặp lồng nhau). Ngoài break ra, ta còn có thể sử dụng từ khóa continue có thể dùng để loại bỏ code tiếp theo của vòng lặp này và trực tiếp cho vòng lặp vào vòng tiếp theo.

Giống như cấu trúc rẽ nhánh ta có thể xây dựng vòng lặp bên trong vòng lặp.

**Ngày 05 - Xây dựng logic chương trình**

Ở bài học này, giáo trình đưa ra các bài tập để ôn luyện các bài trước đã học. Đầu tiên là các bài tập và lời giải, cuối cùng là câu hỏi thực hành như: chuỗi Fibonacci, Số hoàn hảo và số nguyên tố.

**Ngày 06 -Sử dụng chức năng và module**

Hàm có vai trò giúp cho đoạn code không bị lặp lại 1 hay nhiều lần. Chỉ cần gọi hàm ra để thực hiện chức năng mà không cần bị lặp code. (đóng gói các mô-đun chức năng bằng các hàm).

Xác định hàm: trong python người ta sử dụng từ khóa def để định nghĩa hàm. Sau khi hàm được thực hiện chúng ta có thể trả về một giá trị thông qua từ return (tương đương với biến phụ của 1 hàm trong toán học)

Gọi hàm: Chúng ta có thể thực hiện add thêm tham số cho hàm, chúng ta cũng có thể thêm tham số là biến khi không biết chắc chắn về số lượng tham số. Khi sử dụng hàm hay biến ta cũng cần chú ý về vấn đề đặt tên, để tránh những tình huống như xung đột tên. Vì python không có khái niệm nạp chồng hàm nên định nghĩa sau sẽ thay thế cho định nghĩa trước (nghĩa là thực tế hai hàm cùng tên chỉ có 1 chức năng tồn tại). Chúng ta cũng có thể khắc phục việc đặt trùng tên bằng cách sử dụng các module khác nhau, khi muốn gọi chúng ra ta có thể thực hiện cấu trúc câu lệnh: ‘from (tên module) import (tên hàm)’. Ngay cả khi tách từng module mà chúng ta thực hiện hàm cùng 1 đoạn mã thì vẫn bị tình trạng ghi đè hàm.

Phạm vi biến đổi: khi Python tìm kiếm một biến, nó sẽ tìm kiếm theo thứ tự "phạm vi cục bộ", "phạm vi lồng nhau", "phạm vi toàn cầu" và "phạm vi tích hợp". "phạm vi tích hợp" là các mã định danh được tích hợp sẵn trong Python

Chúng ta nên cố gắng hạn chế tối đa việc sử dụng biến toàn cục, vì phạm vi và mức độ ảnh hưởng của biến toàn cục quá rộng, có thể xảy ra những sửa đổi và cách sử dụng không mong muốn.

Tóm lại, từ giờ trở đi, chúng ta có thể viết mã Python theo định dạng sau:

def main():

# Todo: Add your code here

pass

if \_\_name\_\_ == '\_\_main\_\_':

main()

**Ngày 07 - Chuỗi và cấu trúc dữ liệu phổ biến**

Sử dụng chuỗi: Chuỗi là một chuỗi giới hạn bao gồm 0 hoặc nhiều ký tự. Trong python, nếu chúng ta bao quanh 1 hoặc nhiều ký tự bằng dấu ngoặc đơn hay dấu ngoặc kép thì chúng ta đã biểu thị một chuỗi.

Có thể sử dụng dấu ‘\’ (dấu gạch chéo ngược) trong một chuỗi để biểu thị dấu thoát. Nó cũng có thể được theo sau \ bởi một số bát phân hoặc số thập lục phân để biểu thị ký tự. Ví dụ: \141cả \x61hai đều biểu thị các chữ cái viết thường a.

Chúng ta có thể ghép chuỗi hay lặp chuỗi bằng các bộ toán tử cộng hoặc nhân. Chúng ta có thể sử dụng in và not in để xác định xem một chuỗi có chứa ký tự khác hay không), chúng ta cũng có thể sử dụng toán tử [] AND [:] để loại bỏ một ký tự nhất định hoặc các ký tự nhất định khỏi chuỗi (thao tác cắt lát).

Chúng ta có thể thêm chữ cái f vào trước chuỗi để viết định dạng ngắn hơn. Ngoài chuỗi thì python có nhiều loại cấu trúc viết sẵn và phổ biến nhất đó là danh sách, bộ và từ điển.

Cách sử dụng cơ bản của danh sách: Danh sách được viết dưới dạng ngôn ngữ máy trong python là list, nó là một loại cấu trúc, không vô hướng và là một chuỗi các ký tự có giá trị. Sử dụng [] để phân tách nhiều phần tử cũng có thể sử dụng vòng lặp for để duyệt qua các phần tử trong danh sách hoặc có thể sử dụng toán tử [] hoặc [:] để xóa một hoặc nhiều phần tử khỏi danh sách.

Tạo danh sách: Chúng ta có thể tạo danh sách bằng cách sử dụng các câu lệnh được đặt trong []. Ví dụ như:

f = [x for x in range(1, 10)]

print(f)

f = [x + y for x in 'ABCDE' for y in '1234567']

print(f)

Còn có một cách khác để định nghĩa trình tạo trong Python, đó là yield chuyển đổi một hàm thông thường thành hàm tạo thông qua các từ khóa.

Sử dụng bộ dữ liệu: Bộ dữ liệu trong Python tương tự như danh sách và cũng là kiểu dữ liệu chứa. Một biến (đối tượng) có thể được sử dụng để lưu trữ nhiều dữ liệu. Điểm khác biệt là các phần tử của bộ dữ liệu không thể sửa đổi được. Chúng ta đã sử dụng nó nhiều lần trong mã trước đó Tuple.

Kiểu dữ liệu tuple có thể coi là kiểu dữ liệu bất biến, còn kiểu dữ liệu list là kiểu dữ liệu có thể thay đổi được. Tuy nhiên, nếu một phương thức trả về nhiều giá trị thì sử dụng bộ dữ liệu cũng là một lựa chọn tốt.

Cách sử dụng cơ bản của tập hợp: Các tập hợp trong Python nhất quán với các tập hợp toán học. Không được phép có các phần tử trùng lặp và có thể thực hiện các phép toán như giao, hợp và hiệu.

Chúng ta có thể tạo, thêm, xóa các phần tử trong bộ dữ liệu thông qua các câu lệnh như: không có câu lệnh cho tạo, add(), update([]), discard(), remove() (remove phần tử trong câu điều kiện hoặc vòng lặp). Các phép toán như tập hợp, giao hợp, … cũng được sử dụng trong kiểu Tuple.

Cách sử dụng cơ bản của từ điển: Từ điển là một mô hình chứa có thể thay đổi.

Từ điển có thể lưu trữ bất kì đối tượng nào, mỗi phần tử trong từ điển bao gồm một cặp khóa ‘’ và giá trị và được cách nhau bởi dấu :

Ta cũng có thể thêm, xóa các giá trị trong từ điển.

**Ngày 08 -Cơ bản về lập trình hướng đối tượng**

Lớp và Đối tượng:

* Lớp là bản thiết kế và mẫu cho đối tượng.
* Đối tượng là cách thể hiện của lớp.

Xác định lớp: Có thể sử dụng từ khóa class để định nghĩa một lớp, sau đó định nghĩa các phương thức trong lớp thông qua các hàm

Sử dụng đối tượng: Sau khi định nghĩa một lớp, chúng ta có thể tạo các đối tượng và gửi tin nhắn đến các đối tượng theo những cách sau.

def main():

#Tạo một đối tượng sinh viên và chỉ định tên và tuổi

stu1 = Student('Văn A', 38)

#Gửi tin nhắn nghiên cứu tới đối tượng

stu1.study('Lập trình Python ')

#Gửi tin nhắn watch\_av tới đối tượng

stu1.watch\_movie()

stu2 = Student('Văn B', 15)

stu2.study('Lập trình Python')

stu2.watch\_movie()

if \_\_name\_\_ == '\_\_main\_\_':

main()

Nếu muốn thuộc tính ở chế độ riêng tư, ta có thể bắt đầu thuộc tính bằng hai dấu gạch dưới khi đặt tên.

Bốn trụ cột của hướng đối tượng là trừu tượng, đóng gói, kế thừa và đa hình.

**Ngày 09 - Hướng đối tượng nâng cao**

Thuộc tính: muốn truy cập thuộc tính, ta có thể sử dụng phương thức getter (accessor) và setter (modifier) ​​của thuộc tính. Để thực hiện thao tác tương ứng. Nếu muốn thực hiện việc này, ta có thể cân nhắc sử dụng trình bao bọc @property để bao bọc các phương thức getter và setter, giúp việc truy cập vào các thuộc tính trở nên an toàn và thuận tiện.

Nếu ta cần giới hạn đối tượng loại tùy chỉnh chỉ bị ràng buộc với một số thuộc tính nhất định, ta có thể giới hạn nó bằng cách xác định biến \_\_slots\_\_ trong lớp. Cần lưu ý rằng giới hạn của \_\_slots\_\_ chỉ có tác dụng đối với các đối tượng của lớp hiện tại và không có bất kỳ tác dụng nào đối với các lớp con.

Các phương thức trong lớp đều là các phương thức đối tượng, có nghĩa là các phương thức này là các thông điệp được gửi đến đối tượng.

Có 3 phương thức được sử dụng phương thức trong lớp là: phương thức cá thể, phương thức lớp và phương thức tĩnh

Quá tải toán tử - \_\_add\_\_/ \_\_sub\_\_/ \_\_or\_\_/ \_\_getitem\_\_/ \_\_setitem\_\_/ \_\_len\_\_/ \_\_repr\_\_/ \_\_gt\_\_/ \_\_lt\_\_/ \_\_le\_\_/ \_\_ge\_\_/ \_\_eq\_\_/ / \_\_ne\_\_/\_\_contains\_\_

Mối quan hệ giữa các lớp (đối tượng): có ba loại mối quan hệ giữa các lớp: mối quan hệ is-a, has-a và use-a.

* Mối quan hệ is-a còn được gọi là mối quan hệ kế thừa hay khái quát hóa, ví dụ như mối quan hệ giữa điện thoại di động và sản phẩm điện tử đều là mối quan hệ kế thừa.
* Mối quan hệ has-a thường được gọi là mối liên kết, ví dụ như mối quan hệ giữa các phòng ban với nhân viên, giữa ô tô và động cơ đều là mối quan hệ liên kết, nếu mối quan hệ liên kết là liên kết giữa cái toàn thể và bộ phận thì ta gọi là nó là một mối quan hệ tổng hợp; nếu toàn bộ chịu trách nhiệm hơn nữa Vòng đời của bộ phận (toàn bộ và bộ phận là không thể phân chia, cùng tồn tại và chết đi), thì đây là mối tương quan mạnh nhất, mà chúng ta gọi là mối quan hệ tổng hợp.
* Mối quan hệ sử use- a thường được gọi là phụ thuộc, ví dụ người lái xe có hành vi (phương pháp) lái xe trong đó (các tham số) sử dụng ô tô thì mối quan hệ giữa người lái xe và ô tô là mối quan hệ phụ thuộc.

Ta có thể sử dụng một thứ gọi là UML (Ngôn ngữ mô hình hóa thống nhất) để lập mô hình hướng đối tượng.

Bằng cách sử dụng mỗi quan hệ này giữa các lớp, ta có thể hoàn thành một số thao tác nhất định dựa trên các lớp hiện có hoặc tạo các lớp mới dựa trên các lớp hiện có.

Kế thừa và đa hình: Ta có thể tạo các lớp mới dựa trên các lớp hiện có. Một cách để làm điều này là để một lớp kế thừa các thuộc tính và phương thức trực tiếp từ một lớp khác, do đó giảm việc viết mã lặp lại. Những người cung cấp thông tin kế thừa được gọi là lớp cha, còn được gọi là siêu lớp hoặc lớp cơ sở; những người nhận được thông tin kế thừa được gọi là lớp con, còn được gọi là lớp dẫn xuất hoặc lớp dẫn xuất. Ngoài việc kế thừa các thuộc tính và phương thức do lớp cha cung cấp, các lớp con còn có thể định nghĩa các thuộc tính và phương thức riêng của mình nên lớp con có nhiều khả năng hơn lớp cha, trong quá trình phát triển thực tế, ta thường thay thế chúng bằng các đối tượng của lớp con. Đây là một hành vi phổ biến trong lập trình hướng đối tượng và nguyên tắc tương ứng được gọi là nguyên tắc thay thế Richter.

Sau khi kế thừa các phương thức của lớp cha, lớp con có thể cung cấp một phiên bản triển khai mới của các phương thức hiện có của lớp cha, hành động này được gọi là ghi đè phương thức. Thông qua việc viết lại phương thức, ta có thể làm cho cùng một hành vi của lớp cha có các phiên bản triển khai khác nhau trong lớp con. Khi ta gọi phương thức này đã được lớp con viết lại, các đối tượng của lớp con khác nhau sẽ hiển thị các hành vi khác nhau. Điều này được gọi là Đa hình.

**Ngày 10 - Giao diện người dùng đồ họa và phát triển trò chơi**

Mô-đun phát triển GUI mặc định của Python là tkinter

Về cơ bản việc sử dụng tkinter để phát triển ứng dụng GUI yêu cầu 5 bước sau:

1. Nhập những gì chúng ta cần trong mô-đun tkinter.
2. Tạo một đối tượng cửa sổ cấp cao nhất và sử dụng nó để lưu trữ toàn bộ ứng dụng GUI.
3. Thêm các thành phần GUI trên đối tượng cửa sổ cấp cao nhất.
4. Chức năng của các thành phần GUI này được tổ chức thông qua mã.
5. Nhập vòng lặp sự kiện chính (vòng lặp chính).

Tk cung cấp ba trình quản lý bố cục để sắp xếp các điều khiển. Các điều khiển có thể được định vị thông qua trình quản lý bố cục. Ba trình quản lý bố cục này là: Placer (nhà phát triển cung cấp kích thước và vị trí của điều khiển), Packer (tự động điền các điều khiển vào các vị trí thích hợp) và Lưới (đặt các điều khiển dựa trên tọa độ lưới), sẽ không được mô tả ở đây.

Pygame được xây dựng dựa trên SDL là một tập hợp các thư viện phát triển đa phương tiện nền tảng được triển khai bằng ngôn ngữ C và được sử dụng rộng rãi trong việc phát triển trò chơi.

Điều quan trọng nhất ở bài này là chúng ta phải hiểu cách sử dụng các bước trước đó trong quy trình. Giải thích lập trình hướng đối tượng và học cách sử dụng ý tưởng lập trình này để giải quyết các vấn đề thực tế.

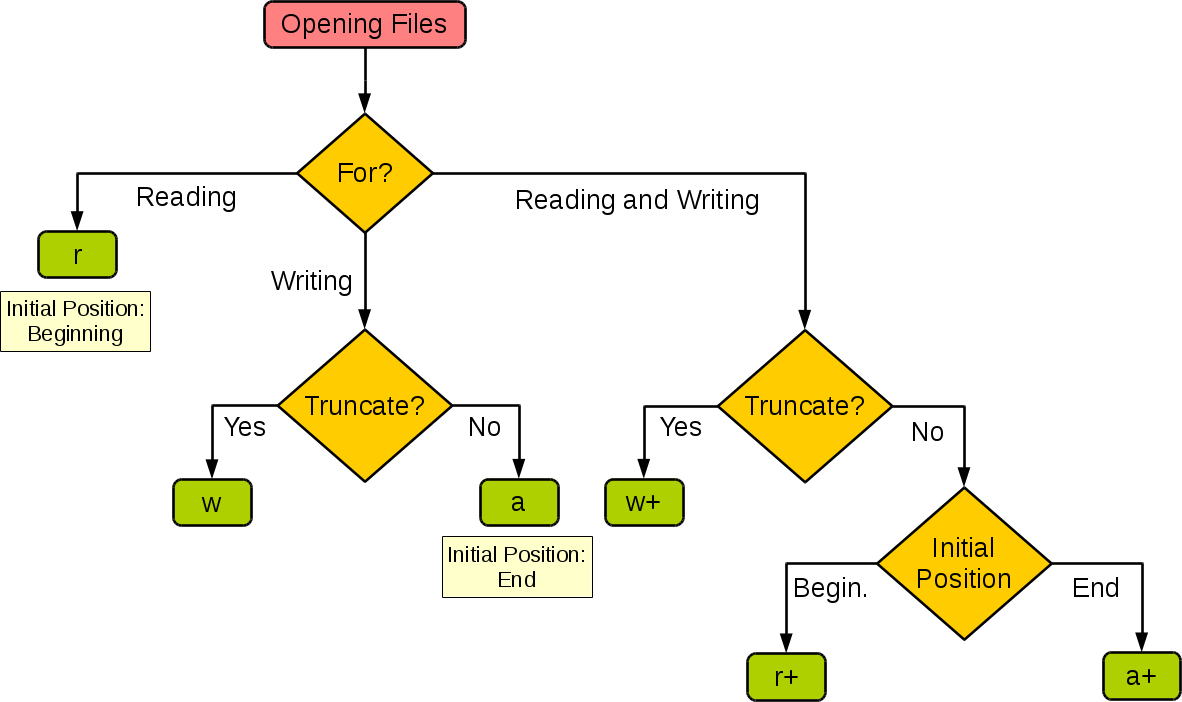
Để tạo ra một chương trình game ta cần các bước cơ bản sau:

1. Tạo cửa sổ trò chơi
2. Vẽ trong cửa sổ
3. Tải hình ảnh
4. Tạo hiệu ứng hoạt hình
5. Kiểm tra tác động
6. Xử lý sự kiện

**Ngày 11 - Tệp và Ngoại lệ**

Đọc file: Python đọc file thông qua các hàm open có sẵn của Python, ta có thể chỉ định tên file, chế độ hoạt động, thông tin mã hóa, v.v. để lấy đối tượng thao tác với file, sau đó chúng ta có thể đọc và viết tập tin. Chế độ hoạt động được đề cập ở đây đề cập đến loại tệp nào sẽ mở (tệp ký tự hoặc tệp nhị phân) và loại hoạt động nào cần thực hiện (đọc, ghi hoặc nối thêm), như được hiển thị trong bảng sau.

|  |  |
| --- | --- |
| Chế độ hoạt động | Ý nghĩa cụ thể |
| ‘r’ | Đọc (mặc định) |
| ‘w’ | Viết (nội dung trước sẽ được cắt ngắn trước) |
| ‘x’ | Viết, một ngoại lệ sẽ được tạo nếu tệp đã tồn tại |
| ‘a’ | Nối, ghi nội dung vào cuối file hiện có |
| ‘b’ | Chế độ nhị phân |
| ‘t’ | Chế độ văn bản (mặc định) |
| ‘+’ | Cập nhật (có thể vừa đọc vừa viết) |



Khi đọc file văn bản, ta cần open chỉ định tên file kèm theo đường dẫn khi sử dụng hàm (có thể sử dụng đường dẫn tương đối hoặc đường dẫn tuyệt đối) và đặt chế độ file thành 'r'(nếu không chỉ định thì giá trị mặc định cũng là 'r'), sau đó encodingchỉ định mã hóa thông qua các tham số (nếu không chỉ định, giá trị mặc định là Không, thì mã hóa mặc định của hệ điều hành sẽ được sử dụng khi đọc tệp) Nếu không thể đảm bảo rằng phương thức mã hóa được sử dụng khi lưu tệp phù hợp với phương pháp mã hóa được chỉ định bởi tham số mã hóa thì có thể không thực hiện được do không thể giải mã ký tự gây ra lỗi đọc.

Ngoài việc sử dụng read phương thức của đối tượng tệp để đọc tệp, ta cũng có thể sử dụng for-in vòng lặp để đọc từng dòng hoặc sử dụng readlinesphương thức để đọc tệp theo từng dòng vào một thùng chứa danh sách.

Ghi tập tin: Việc ghi thông tin văn bản vào file cũng rất đơn giản, chỉ cần open chỉ định tên file và đặt chế độ file khi sử dụng chức năng 'w'. Lưu ý rằng nếu ta cần nối thêm nội dung tệp, chế độ nên được đặt thành 'a'. Nếu tệp cần ghi không tồn tại, tệp sẽ được tạo tự động thay vì đưa ra ngoại lệ.

Xử lý ngoại lệ: Trong Python, chúng ta có thể đặt mã có thể gây ra sự cố trong thời gian chạy trong try một khối mã và try một hoặc nhiều khối mã có thể được theo sau bởi một hoặc nhiều khối mã except để nắm bắt các ngoại lệ có thể xảy ra.

Chúng ta try thực hiện theo ba bước except để xử lý chúng một cách riêng biệt. Ba tình trạng bất thường khác nhau. Cuối cùng, chúng ta sử dụng finally khối mã để đóng tệp đang mở và giải phóng các tài nguyên bên ngoài thu được trong chương trình

Muốn lưu dữ liệu trong danh sách hoặc từ điển vào file thì phải lưu dữ liệu ở định dạng JSON. JSON là tên viết tắt của "JavaScript Object Notation"

JSON trên thực tế giống với từ điển trong Python, ta có thể dễ dàng tìm thấy mối quan hệ tương ứng giữa các kiểu dữ liệu của JSON và các kiểu dữ liệu của Python, như thể hiện trong hai bảng sau.

|  |  |
| --- | --- |
| JSON | Python |
| object | dict |
| array | list |
| string | str |
| number (int / real) | int / float |
| true / false | True / False |
| null | None |

|  |  |
| --- | --- |
| Python | JSON |
| dict | object |
| list, tuple | array |
| str | string |
| int, float, int- & float-derived Enums | number (int / real) |
| True / False | true / false |
| None | null |

Mô-đun json chủ yếu có bốn chức năng quan trọng, đó là:

* dump- Tuần tự hóa các đối tượng Python thành các tệp ở định dạng JSON
* dumps- Xử lý các đối tượng Python thành các chuỗi có định dạng JSON
* load- Deserialize dữ liệu JSON trong file thành các đối tượng
* loads- Giải tuần tự hóa nội dung của chuỗi thành đối tượng Python

**Ngày 12 - Chuỗi và biểu thức chính quy**

Các thao tác chuỗi nâng cao: Khi viết chương trình hoặc trang web xử lý chuỗi ta thường tìm các chuỗi tuân theo một số quy tắc phức tạp được gọi là biểu thức chính quy, biểu thức chính quy rất quan trọng. Hầu như tất cả các ngôn ngữ lập trình ngày nay đều hỗ trợ các thao tác biểu thức chính quy và Python hỗ trợ các thao tác biểu thức chính quy thông qua mô-đun re trong thư viện chuẩn.

Dưới đây là các kí hiệu cơ bản trong biểu thức chính quy.

|  |  |  |  |
| --- | --- | --- | --- |
| Kí hiệu | Giải thích | Ví dụ | Minh họa |
| . | Phù hợp với bất kỳ ký tự nào | bt | Có thể khớp với bat/but/b#t/b1t, v.v. |
| \w | Nối các chữ cái/ số/ dấu gạch dưới | b\wt | Có thể khớp với bat /b1t /b\_t, v.v.  nhưng không khớp với b#t |
| \s | Khớp các ký tự khoảng trắng (bao gồm \r, \n, \t, v.v.) | yêu bạn | Có thể sánh được yêu em |
| \d | Số trận đấu | \d\d | Có thể khớp với 23/01/99, v.v. |
| \b | Nối ranh giới từ | \bCái\b |  |
| ^ | Khớp với phần đầu của một chuỗi | ^Cái | Có thể khớp các chuỗi bắt đầu bằng The |
| $ | So khớp phần cuối của chuỗi | .exe$ | Có thể khớp các chuỗi kết thúc bằng .exe |
| \W | Khớp các chữ cái/số/dấu gạch dưới | b\Wt | Có thể khớp với b#t/b@t, v.v. nhưng không thể khớp với but/b1t/b\_t, v.v. |
| \S | Khớp các ký tự không phải khoảng trắng | yêu bạn | Có thể sánh được với yêu#bạn, v.v. nhưng không thể sánh được với yêu bạn |
| \D | Khớp các chữ số không | \d\D | Có thể khớp 9a/3#/0F, v.v. |
| \B | Khớp các ranh giới không phải từ | \Bio\B |  |
| [] | Khớp với bất kỳ ký tự đơn nào từ bộ ký tự | [tín dụng] | Có thể khớp với bất kỳ ký tự nguyên âm nào |
| [^] | Khớp với bất kỳ ký tự đơn nào không có trong bộ ký tự | [^aiyou] | Có thể khớp với bất kỳ ký tự không phải nguyên âm nào |
| \* | Trận đấu 0 hoặc nhiều lần | \w\* |  |
| + | Trận đấu 1 hoặc nhiều lần | \w+ |  |
| ? | Trận đấu 0 hoặc 1 lần | \w? |  |
| {N} | Trận đấu N lần | \w{3} |  |
| {M,} | Trùng khớp ít nhất M lần | \w{3,} |  |
| {M,N} | Trùng khớp ít nhất M lần và nhiều nhất N lần | \w{3,6} |  |
| | | chi nhánh | foo|bar | Có thể khớp foo hoặc bar |
| (?#) | Bình luận |  |  |
| (điểm kinh nghiệm) | Ghép điểm kinh nghiệm và chụp vào các nhóm được đặt tên tự động |  |  |
| (?<tên>điểm kinh nghiệm) | Ghép điểm kinh nghiệm và chụp vào một nhóm có tên |  |  |
| (?:exp) | Khớp với exp nhưng không thu được văn bản khớp |  |  |
| (?=exp) | Khớp vị trí trước exp | \b\w+(?=ing) | Có thể phù hợp với điệu nhảy trong Tôi đang nhảy |
| (?<=exp) | Khớp vị trí sau điểm kinh nghiệm | (?<=\bdanc)\w+\b | Có thể khớp với chữ đầu tiên trong Tôi thích nhảy múa và đọc sách |
| (?! điểm kinh nghiệm) | Khớp vị trí không có exp theo sau |  |  |
| (?<!exp) | Khớp với vị trí không có exp trước |  |  |
| \*? | Lặp lại bao nhiêu lần tùy thích, nhưng càng ít lần càng tốt | a.\*b a.\*?b | Áp dụng biểu thức chính quy cho aabab. Biểu thức chính quy sẽ khớp với toàn bộ chuỗi aabab và biểu thức chính quy sau sẽ khớp với hai chuỗi aab và ab. |
| +? | Lặp lại 1 lần hoặc nhiều hơn nhưng càng ít lần càng tốt |  |  |
| ?? | Lặp lại 0 hoặc 1 lần, nhưng càng ít lần càng tốt |  |  |
| {M,N}? | Lặp lại M đến N lần, nhưng càng ít lần càng tốt |  |  |
| {M,}? | Lặp lại M lần hoặc nhiều hơn, nhưng lặp lại ít nhất có thể |  |  |

Python cung cấp mô-đun re để hỗ trợ các hoạt động liên quan đến biểu thức chính quy. Dưới đây là các hàm cốt lõi trong mô-đun re.

|  |  |
| --- | --- |
| Chức năng | Minh họa |
| biên dịch (mẫu, cờ = 0) | Biên dịch một biểu thức chính quy trả về một đối tượng biểu thức chính quy |
| match (mẫu, chuỗi, cờ=0) | Khớp một chuỗi với một biểu thức chính quy và trả về đối tượng khớp nếu thành công, nếu không thì trả về Không |
| tìm kiếm(mẫu, chuỗi, cờ=0) | Lần xuất hiện đầu tiên của mẫu biểu thức chính quy trong chuỗi tìm kiếm sẽ trả về thành công một đối tượng phù hợp, nếu không thì trả về Không. |
| chia (mẫu, chuỗi, maxsplit=0, flags=0) | Tách chuỗi bằng dấu phân cách mẫu được chỉ định bởi danh sách trả về biểu thức chính quy |
| sub (mẫu, thay thế, chuỗi, count=0, flags=0) | Sử dụng chuỗi đã chỉ định để thay thế mẫu khớp với biểu thức chính quy trong chuỗi gốc. Bạn có thể sử dụng số đếm để chỉ định số lần thay thế. |
| fullmatch (mẫu, chuỗi, cờ=0) | Phiên bản khớp chính xác (từ đầu đến cuối chuỗi) của hàm khớp |
| findall (mẫu, chuỗi, cờ=0) | Tìm tất cả các mẫu trong một chuỗi khớp với biểu thức chính quy và trả về danh sách các chuỗi |
| finditer (mẫu, chuỗi, cờ=0) | Tìm tất cả các mẫu trong một chuỗi khớp với biểu thức chính quy và trả về một trình vòng lặp |
| thanh lọc() | Xóa bộ nhớ đệm của các biểu thức chính quy được biên dịch ngầm |
| re.I/re.IGNORECASE | Bỏ qua các thẻ khớp chữ hoa chữ thường |
| re.M / re.MULTILINE | thẻ đối sánh nhiều dòng |

**Ngày 13 - Quy trình và chủ đề**

Khái niệm về tiến trình và luồng

* Tiến trình là là một chương trình (program) đang được thực thi. Một tiến trình có thể tạo một tiến trình mới thông qua fork hoặc spawn để thực hiện các tác vụ khác.
* Luồng (thread) là một khối các câu lệnh (instructions) độc lập trong một tiến trình và có thể được lập lịch bởi hệ điều hành

Python hỗ trợ cả đa tiến trình và đa luồng, vì vậy có ba cách chính để sử dụng Python thực hiện lập trình đồng thời: đa tiến trình, đa luồng và đa tiến trình + đa luồng.

Nếu mã trong chương trình chỉ có thể được thực thi từng chút một theo thứ tự, thì ngay cả khi hai tác vụ tải xuống không liên quan được thực thi, bạn cần đợi một tệp được tải xuống trước khi bắt đầu lần tải xuống tiếp theo. Nhiệm vụ này rõ ràng là không hợp lý và không hiệu quả.

Chúng ta cũng có thể sử dụng các lớp và hàm trong module quy trình con để tạo và khởi động các quy trình con, sau đó giao tiếp với quy trình con thông qua các pipe

Chúng ta có thể trực tiếp sử dụng Thread các lớp của mô-đun luồng để tạo các luồng, nhưng trước đây chúng ta đã nói về một khái niệm rất quan trọng được gọi là "kế thừa". Chúng ta có thể tạo các lớp mới từ các lớp hiện có, vì vậy chúng ta cũng có thể tạo các lớp tùy chỉnh bằng cách kế thừa các lớp Thread. class, sau đó tạo đối tượng luồng và bắt đầu luồng.

Cải tiến quan trọng nhất trong hoạt động I/O trong các hệ điều hành hiện đại là hỗ trợ I/O không đồng bộ

Trong ngôn ngữ Python, mô hình lập trình I/O đơn luồng + không đồng bộ được gọi là coroutine

Nếu muốn tận dụng tối đa các tính năng đa lõi của CPU, phương pháp đơn giản nhất là multi-process + coroutine, phương pháp này không chỉ tận dụng tối đa đa lõi mà còn phát huy hết hiệu quả cao của coroutine, và có thể đạt được hiệu suất cực cao.

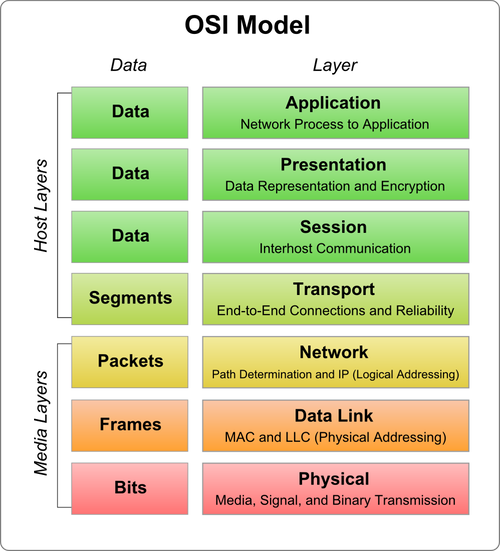
**Ngày 14 - Nhập môn lập trình mạng và phát triển ứng dụng mạng**

Khái niệm cơ bản về mạng máy tính : Mạng máy tính là thuật ngữ chung để chỉ một hệ thống gồm các máy tính độc lập được kết nối với nhau. Mục đích chính của việc thiết lập mạng máy tính là đạt được sự liên lạc và chia sẻ tài nguyên giữa nhiều máy tính.

* Lịch sử phát triển mạng máy tính:

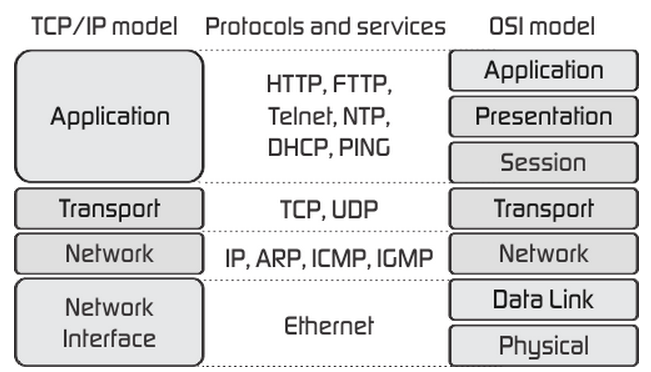
+ Những năm 1960 - Dự án ARPANET của Bộ Quốc phòng Hoa Kỳ được triển khai, đặt nền móng cho các mạng chuyển mạch gói.

+ Những năm 1980 - Tổ chức Tiêu chuẩn hóa Quốc tế (ISO) phát hành OSI/RM, đặt nền tảng cho việc tiêu chuẩn hóa công nghệ mạng.



+ Những năm 1990 - Tim Berners- Lee người Anh phát minh ra trình duyệt đồ họa. Sự đơn giản và dễ sử dụng của trình duyệt khiến mạng máy tính nhanh chóng trở nên phổ biến.

* Mô hình TCP/IP: bộ giao thức là một loạt các giao thức và mô hình truyền thông mà chúng cấu thành và các tập hợp này thường được gọi là mô hình TCP/IP.



Sở dĩ TCP được gọi là giao thức truyền tải đáng tin cậy là vì TCP thực hiện thao tác bắt tay 3 bước:

Không có mất mát hoặc lỗi trong quá trình truyền dữ liệu (điều này có thể đạt được bằng cách sử dụng các cơ chế bắt tay, xác minh và truyền lại).

Kiểm soát luồng (khớp tốc độ truyền giữa người gửi và người nhận dữ liệu qua cửa sổ trượt).

Kiểm soát tắc nghẽn (giảm thiểu tắc nghẽn mạng thông qua thời gian RTT và kiểm soát cửa sổ trượt).

Truy cập tài nguyên mạng dựa trên giao thức HTTP: HTTP là tên viết tắt của Giao thức truyền siêu văn bản. Thông qua HTTP, chúng ta có thể lấy tài nguyên (dựa trên ký tự) trên mạng và API mạng (còn gọi là giao diện dữ liệu mạng) dựa trên HTTP để thực hiện việc truyền dữ liệu.

JSON ( Java Script Object Notation ) là một ngôn ngữ trao đổi dữ liệu nhẹ, dựa trên văn bản dễ đọc (văn bản thuần túy) và được sử dụng để truyền các giá trị thuộc tính hoặc ...nối tiếpcác giá trị. So với XML, JSON có tính di động và thanh lịch hơn.

request là một thư viện của bên thứ ba sử dụng mạng dựa trên giao thức HTTP. Sử dụng các yêu cầu Thư viện làm cho việc sử dụng HTTP trở nên rất thuận tiện, tránh các lỗi bảo mật, mã dư thừa và "phát minh lại bánh xe"

Socket là một tập hợp các thư viện phát triển ứng dụng được viết bằng ngôn ngữ C, chủ yếu được sử dụng để thực hiện giao tiếp giữa các tiến trình và lập trình mạng được sử dụng rộng rãi trong phát triển ứng dụng mạng.

Các shocket được sử dụng trong quá trình phát triển thực tế có thể được chia thành 2 loại chính là Stream Socket (dựa trên giao thức TCP) và Datagram Socket (dựa trên giao thức UDP).

Chúng ta cũng có thể tạo mã trình duyệt để gửi email cũng như tin nhắn sms.

**Ngày 15 – Xử lý hình ảnh và tài liệu**

Ta có thể sử dụng mô-đun của bên thứ 3 để xử lý hình ảnh, tài liệu văn phòng thường xuất hiện trong quá trình phát triển thực tế. Ta có thể cắt ảnh, thu nhỏ hình ảnh, phóng và dán hình ảnh, xoay và lật hình ảnh, xử lí điểm ảnh, hiệu ứng lọc.

Đọc và viết tài liệu Word: Sử dụng mô-đun python-docx, Python có thể tạo và sửa đổi tài liệu Word

from docx import Document

from docx.shared import Inches

document = Document()

document.add\_heading('Document Title', 0)

p = document.add\_paragraph('A plain paragraph having some ')

p.add\_run('bold').bold = True

p.add\_run(' and some ')

p.add\_run('italic.').italic = True

document.add\_heading('Heading, level 1', level=1)

document.add\_paragraph('Intense quote', style='Intense Quote')

document.add\_paragraph(

'first item in unordered list', style='List Bullet'

)

document.add\_paragraph(

'first item in ordered list', style='List Number'

)

document.add\_picture('monty-truth.png', width=Inches(1.25))

records = (

(3, '101', 'Spam'),

(7, '422', 'Eggs'),

(4, '631', 'Spam, spam, eggs, and spam')

)

table = document.add\_table(rows=1, cols=3)

hdr\_cells = table.rows[0].cells

hdr\_cells[0].text = 'Qty'

hdr\_cells[1].text = 'Id'

hdr\_cells[2].text = 'Desc'

for qty, id, desc in records:

row\_cells = table.add\_row().cells

row\_cells[0].text = str(qty)

row\_cells[1].text = id

row\_cells[2].text = desc

document.add\_page\_break()

document.save('demo.docx')

Đọc và ghi file Excel: Mô-đun openpyxl của Python cho phép chúng ta đọc và sửa đổi bảng tính Excel trong các chương trình Python.

import datetime

from openpyxl import Workbook

wb = Workbook()

ws = wb.active

ws['A1'] = 42

ws.append([1, 2, 3])

ws['A2'] = datetime.datetime.now()

wb.save("sample.xlsx")

**Ngày 16~Ngày 20 - Ngôn ngữ Python nâng cao**

Kiến thức quan trọng:

* Sử dụng biểu thức tổng quát
* Cạm bẫy của danh sách lồng nhau
* Mô-đun heapq
* Mô-đun itertools
* Mô-đun collections

+ namedtuple: là một nhà máy lớp chấp nhận tên của một loại và danh sách các thuộc tính để tạo một lớp.

+ deque: một cách triển khai danh sách thay thế. Lớp dưới cùng của danh sách trong Python được triển khai dựa trên mảng, trong khi lớp dưới cùng của deque là danh sách liên kết đôi do đó, khi ta cần thêm và xóa các phần tử ở đầu và đuôi, deque sẽ hiển thị hiệu suất tốt hơn, với tiệm cận độ phức tạp về thời gian của.

+ Counter: value là số đếm của phần tử, most\_common() phương thức của nó có thể giúp chúng ta lấy được phần tử có tần suất cao nhất. Theo nguyên tắc CARP, việc thiết kế mối quan hệ giữa Counter và dict như một mối quan hệ liên kết sẽ hợp lý hơn.

+ OrderedDict: dict Một lớp con ghi lại thứ tự các cặp khóa-giá trị được chèn vào, dường như nó có hoạt động của cả từ điển và danh sách liên kết.

+ defaultdict: Tương tự như loại từ điển, nhưng giá trị mặc định tương ứng với khóa có thể được lấy thông qua hàm mặc định của nhà máy. So với phương thức trong từ điển setdefault(), cách tiếp cận này hiệu quả hơn.

Cấu trúc dữ liệu chung:

* Thuật toán: Phương pháp và các bước giải quyết vấn đề
* Đánh giá chất lượng của thuật toán: độ phức tạp thời gian tiệm cận và độ phức tạp không gian tiệm cận.
* Ký hiệu big O cho độ phức tạp thời gian tiệm cận:

+ https://camo.githubusercontent.com/44ba5f6ba25b0fae0eaec4f1aeccac8ce1124791ef98ded00466e336dff8cf4d/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f286329- Độ phức tạp về thời gian không đổi - Bộ lọc Bloom/lưu trữ băm

+ https://camo.githubusercontent.com/c84069cd18fc61eac8f4f1b899ba6ec878a4c21af64c04d8853fbb150d01a788/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f286c6f675f326e29- Độ phức tạp thời gian logarit - tìm kiếm nhị phân (tìm kiếm nhị phân)

+ https://camo.githubusercontent.com/9ad5f8dcd5bdae8b66205c7472fed2e90d2f569698b6ae5034993c434a4bd11a/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f286e29- Độ phức tạp thời gian tuyến tính - sắp xếp tìm kiếm/đếm tuần tự

+ https://camo.githubusercontent.com/bf71fc4b5a6a518fad4d54001f7c7bee82dc1d97d7abbcc4c1d2277b93d0f67f/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f286e2a6c6f675f326e29- Độ phức tạp thời gian log-tuyến tính - thuật toán sắp xếp nâng cao (sắp xếp hợp nhất, sắp xếp nhanh)

+ https://camo.githubusercontent.com/f3038618c31ec54aa23dcf811d5ed03dcd29eccb0fc4770bd396ae8ed52be161/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f286e5e3229- Độ phức tạp về thời gian bình phương - Các thuật toán sắp xếp đơn giản (sắp xếp lựa chọn, sắp xếp chèn, sắp xếp bong bóng)

+ https://camo.githubusercontent.com/baf1669c4a134e97391d6fc955a13f4ee6939fda151dc759d06d97e0f65b0415/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f286e5e3329- Độ phức tạp thời gian khối

+ https://camo.githubusercontent.com/22b2ff251b344e42a2cdfad014b1b5788dd9b1533e85d99033a6c9616bc7c27b/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f28325e6e29- Độ phức tạp của chuỗi hình học

+ https://camo.githubusercontent.com/213fa8ee7fceaccee605072b5db903789bdbc4bd46b0b5a5ede19a50e2c2e270/687474703a2f2f6c617465782e636f6465636f67732e636f6d2f6769662e6c617465783f4f286e2129- Độ phức tạp về thời gian giai thừa

* Các thuật toán sắp xếp (chọn, tạo bong bóng và hợp nhất) và các thuật toán tìm kiếm (tuần tự và giảm một nửa)
* Các thuật toán thường được sử dụng:

+ (5-6 thuật toán cần tìm hiểu trên mạng)

Cách sử dụng hàm nâng cao:

* Hàm có chức năng như “công dân hạng nhất”

+ Các hàm có thể được gán cho các biến

+ Hàm có thể được sử dụng làm đối số cho hàm

+ Hàm có thể được sử dụng làm giá trị trả về của hàm

* Cách sử dụng các hàm bậc cao hơn (filter, map và các hàm thay thế của chúng)
* Tham số vị trí, tham số biến, tham số từ khóa, tham số từ khóa được đặt tên
* Siêu thông tin về các tham số (vấn đề về khả năng đọc mã)
* Các sử dụng hàm ẩn danh và hàm nội tuyến (hàm lambda)
* Các vấn đề về đóng cửa và phạm vi

+ Thứ tự LEGB của các biến tìm kiếm Python (Local >>> Embedded>>>Global>>>Built-in)

+ Global và nonlocal vai trò của 2 từ khóa này là:

Global: Khai báo hoặc xác định một biến toàn cục (sử dụng trực tiếp một biến phạm vi toàn cục hiện có hoặc xác định một biến và đặt nó trong phạm vi toàn cục).

Nonlocal: Khai báo một biến sử dụng phạm vi lồng nhau (biến phải tồn tại trong phạm vi lồng nhau, nếu không sẽ báo lỗi).

+ Hàm trang trí (sử dụng trang trí và không trang trí): Nếu trình trang trí không muốn được print ghép nối với hàm, bạn có thể viết một trình trang trí có thể được tham số hóa.

Kiến thức nâng cao về hướng đối tượng

* Ba trụ cột: Đóng gói, kế thừa và đa hình
* Mối quan hệ giữa các lớp:

+ is-a: thừa kế

+ has-a: liên kết/tập hợp/tổng hợp

+use-a: phụ thuộc

* Bản sao của các đối tượng (….)
* Thu gom rác, tham chiếu vòng tròn và tham chiếu yếu: Python sử dụng quản lý bộ nhớ tự động. Cơ chế quản lý này dựa trên việc đếm tham chiếu và cũng giới thiệu các chiến lược được bổ sung bởi các cơ chế thu thập thế hệ và xóa dấu.

Các tình huống dẫn đến số lượng tham chiếu +1:

+ Đối tượng được tạo, ví dụ: a = 23

+ Các đối tượng được tham chiếu, ví dụ:b = a

+ Các đối tượng được truyền dưới dạng tham số cho một hàm, ví dụ f(a)

+ Đối tượng được lưu trữ trong vùng chứa dưới dạng một phần tử, ví dụ:list1 = [a, a]

Các tình huống dẫn đến số lượng tham chiếu là -1:

+ Bí danh đối tượng bị hủy rõ ràng, ví dụ:del a

+ Bí danh của đối tượng được đặt cho đối tượng mới, ví dụ:a = 24

+ Khi một đối tượng rời khỏi phạm vi của nó, ví dụ: khi hàm f hoàn thành việc thực thi, các biến cục bộ trong hàm f (các biến toàn cục sẽ không)

+ Vùng chứa đối tượng bị phá hủy hoặc đối tượng bị xóa khỏi vùng chứa

Việc đếm tham chiếu có thể gây ra sự cố tham chiếu vòng và tham chiếu vòng có thể gây rò rỉ bộ nhớ, như minh họa trong đoạn mã sau. Để giải quyết vấn đề này, "quét đánh dấu" và "bộ sưu tập thế hệ" đã được giới thiệu trong Python. Khi một đối tượng được tạo, đối tượng đó được đặt ở thế hệ đầu tiên. Nếu đối tượng sống sót qua việc kiểm tra rác ở thế hệ đầu tiên, đối tượng đó sẽ được đặt vào thế hệ thứ hai. Tương tự, trong thế hệ rác thứ hai Nếu đối tượng sống sót qua kiểm tra , nó sẽ được đặt ở thế hệ thứ ba.

Vấn đề về tham chiếu vòng cũng có thể được weakrefgiải quyết bằng cách xây dựng tham chiếu yếu thông qua các mô-đun.

Nguyên tắc thiết kế hướng đối tượng

* Nguyên tắc trách nhiệm duy nhất ( SRP ) - một lớp chỉ làm những gì nó nên làm (lớp phải được thiết kế để có tính gắn kết cao)
* Nguyên tắc đóng mở ( OCP ) - Các thực thể phần mềm phải đóng cửa để phát triển và sửa đổi phần mở rộng
* Nguyên tắc đảo ngược phụ thuộc (DIP) - Lập trình hướng trừu tượng (đã bị suy yếu ở các ngôn ngữ được gõ yếu)
* Nguyên tắc thay thế Liskov ( L SP) - Bạn có thể thay thế đối tượng lớp cha bằng đối tượng lớp con bất kỳ lúc nào
* Nguyên tắc phân tách giao diện ( ISP ) - Giao diện phải nhỏ và chuyên dụng thay vì lớn và toàn diện (không có khái niệm về giao diện trong Python)
* Nguyên tắc tái sử dụng tổng hợp thành phần (CARP) - Ưu tiên sử dụng các mối quan hệ mạnh mẽ hơn là các mối quan hệ kế thừa để sử dụng lại mã
* Nguyên tắc Ít hiểu biết nhất (Luật Demeter, Lo D ) - Không gửi tin nhắn đến những đối tượng mà bạn không có mối liên hệ cần thiết.

Mẫu thiết kế GoF

* Các mô hình sáng tạo: singleton, nhà máy, người xây dựng, nguyên mẫu
* Các mẫu cấu trúc: bộ chuyển đổi, mặt tiền (ngoại hình), proxy
* Các mẫu hành vi: vòng lặp, người quan sát, trạng thái, chiến lược

Trình vòng lặp:

Iterator là các đối tượng thực hiện giao thức iterator.

* + Protocol Không có từ khóa nào giống như hoặc interfaceđịnh nghĩa các giao thức trong Python .
  + Các phương thức ma thuật được sử dụng để biểu diễn các giao thức trong Python.
  + \_\_iter\_\_Và \_\_next\_\_phương pháp kỳ diệu là giao thức lặp.

Trình tạo là phiên bản đơn giản về mặt cú pháp của trình vòng lặp.

Máy phát điện phát triển thành coroutines.

Các đối tượng của trình tạo có thể sử dụng các phương thức để gửi dữ liệu, dữ liệu này sẽ trở thành giá trị thu được send() thông qua một biểu thức trong hàm của trình tạo yield. Bằng cách này, Generator có thể được sử dụng như một coroutine. Các coroutine đơn giản là các chương trình con có thể hợp tác với nhau.

Lập trình đồng thời: Có ba cách để triển khai lập trình đồng thời trong Python: đa luồng, đa tiến trình và I/O không đồng bộ. Ưu điểm của lập trình đồng thời là có thể cải thiện hiệu quả thực thi của chương trình và cải thiện trải nghiệm người dùng; nhược điểm là các chương trình đồng thời không dễ phát triển và gỡ lỗi và không thân thiện với các chương trình khác.

* Đa luồng: Python cung cấp các Thread lớp được bổ sung bởi Lock, Condition, Có một GIL trong Python để ngăn nhiều luồng thực thi mã byte cục bộ cùng một lúc. Khóa này là cần thiết cho CPython vì việc quản lý bộ nhớ của CPython không an toàn cho luồng. Do sự tồn tại của GIL nên đa luồng không thể tận dụng lợi thế của tính năng đa lõi của CPU Event, Semaphore, Barrier
* Đa tiến trình: Đa tiến trình có thể giải quyết vấn đề GIL một cách hiệu quả. Lớp chính để triển khai đa quy trình là Process. Các lớp phụ trợ khác threading tương tự như các lớp trong mô-đun. Ta có thể sử dụng các đường ống, ổ cắm, v.v. để chia sẻ dữ liệu giữa các quy trình multiprocessing. Có một Queue lớp trong mô-đun. , cung cấp một hàng đợi được chia sẻ bởi nhiều quy trình dựa trên cơ chế đường ống và khóa. Sau đây là ví dụ từ tài liệu chính thức về đa quy trình và nhóm quy trình.
* Xử lý không đồng bộ: Python 3 hỗ trợ xử lý không đồng bộ thông qua asyncio mô-đun await và async từ khóa. Có một thư viện của bên thứ ba tên là Python in Python aiohttp, cung cấp máy khách và máy chủ HTTP không đồng bộ. Thư viện bên thứ ba này có thể asynciohoạt động với các mô-đun và cung cấp Futurehỗ trợ cho các đối tượng. Python 3.6 đã giới thiệu asyncvà awaitxác định các hàm để thực thi không đồng bộ và tạo ngữ cảnh không đồng bộ và chúng chính thức trở thành từ khóa trong Python 3.7

**Ngày 21~30 - Giới thiệu về Web front-end**

HTML là ngôn ngữ dùng để mô tả các trang web, tên đầy đủ của nó là Hyper-Text Markup Language, là ngôn ngữ đánh dấu siêu văn bản. Văn bản, nút, hình ảnh, video và các thành phần khác mà chúng ta thấy khi duyệt các trang web đều được viết bằng HTML và được trình bày thông qua trình duyệt.

Các tính năng mới của HTML5:

* Giới thiệu hỗ trợ đa phương tiện gốc (thẻ âm thanh và video)
* Giới thiệu nội dung có thể lập trình (thẻ canvas)
* Giới thiệu Web ngữ nghĩa (các thẻ như bài viết, sang một bên, chi tiết, hình ảnh, chân trang, tiêu đề, điều hướng, phần, tóm tắt, v.v.)
* Giới thiệu các điều khiển biểu mẫu mới (lịch, hộp thư, tìm kiếm, thanh trượt, v.v.)
* Giới thiệu hỗ trợ tốt hơn cho việc lưu trữ ngoại tuyến (localStorage và sessionStorage)
* Giới thiệu hỗ trợ cho việc định vị, kéo và thả, WebSocket, tác vụ nền và hơn thế nữa

Sử dụng thẻ HTML để mang nội dung trang:

* Kết cấu:

+ HTML: Mở đầu (tiêu đề, meta), thân

* Chữ:

+ Heading và đoạn paragraph: h1~h6, P

+ superscript và subscript: sup, sub

+ Trống (khoảng trắng được thu gọn)

+ break và horizontal ruler: br, hr

+ Thẻ ngữ nghĩa: Táo bạo và nhấn mạnh-strong, Trích dẫn-blockquote, Chữ viết tắt và từ viết tắt-abbr/acronym, trích dẫn-cite, Thông tin liên hệ chủ sở hữu-address, Sửa đổi nội dung-ins/del

* Danh sách:

+ Ordered list – ol/li

+ Unordered list – ul/li

+ Definition list – dl/dt/đ

* Liên kết: Trang liên kết, liên kết neo, liên kết chức năng
* Hình ảnh: Vị trí lưu trữ hình ảnh, hình ảnh, chiều rộng và chiều cao của nó, chọn định dạng hình ảnh phù hợp (JPEG, GIF, PNG), minh họa vector, thẻ ngữ nghĩa – figure / figcaption.
* Table:

+ Cấu trúc bảng cơ bản - table/tr/td/th

+ Tiêu đề của bảng - caption

+ Trên các hàng và cột - thuộc tính rowpan/thuộc tính colspan

+ Dạng dài - thead/tbody/tfoot

* Hình thức:

+ Các thuộc tính quan trọng - action / method / enctype

+ Điều khiển biểu mẫu (input) - type

Hộp văn bản - text / Hộp mật khẩu - password / Hộp số - number

Email - email / Điện thoại - tel / Ngày - date / Thanh trượt - range / URL - url / Tìm kiếm - search

Nút radio - radio / Nút kiểm tra - checkbox

Tải lên tệp - file / Miền ẩn - hidden

Nút gửi - submit / Nút hình ảnh - image / Nút đặt lại - reset

Danh sách thả xuống - select / option

Vùng văn bản- textarea

Kết hợp các phần tử biểu mẫu - fieldset / legend

* Âm thanh/video:

+ Các định dạng và trình phát video

+ Dịch vụ lưu trữ video

+ Đang chuẩn bị thêm video

+ Thẻ và thuộc tính video - autoplay / controls / loop / muted / preload / src

+ Thẻ và thuộc tính âm thanh- autoplay / controls / loop / muted / preload / src / width / height / poster

* Frame:

+ Bộ khung (lỗi thời, không được dùng nữa) - frameset / frame

+ Cửa sổ nội tuyến – iframe

* Khác:

+ Loại tài liệu

+ Bình luận

+ Thuộc tính

Id: mã định danh duy nhất

class: lớp mà phần tử thuộc về, dùng để phân biệt các phần tử khác nhau

title: thông tin bổ sung về phần tử (văn bản chú giải công cụ sẽ được hiển thị khi di chuột)

tabindex: Thứ tự chuyển đổi phím tab

contenteditable: phần tử có thể chỉnh sửa được không

draggable: phần tử có thể được kéo hay không

+ Phần tử cấp khối/phần tử cấp hàng

+ Thực thể ký tự (thực thể thay thế)

Kết xuất trang bằng CSS

* Giới thiệu: Vai trò của CSS; CSS hoạt động như thế nào; quy tắc, thuộc tính và giá trị; bộ chọn chung
* Màu sắc: cách chỉ định màu, thuật ngữ màu sắc và độ tương phản màu sắc, màu nền
* Văn bản (văn bản/phông chữ)

+ Cỡ chữ và phông chữ (font-size / font-family)

+ Trọng lượng, kiểu dáng, độ giãn và trang trí (font-weight / font-style / font-stretch / text-decoration)

+ Chiều cao dòng, khoảng cách chữ và khoảng cách từ

+ Căn chỉnh (text-align) và thụt lề (text-ident)

+ Kiểu liên kết (:link / :visited / :active / :hover)

+ Thuộc tính mới của CSS3

Hiệu ứng đổ bóng - đổ bóng văn bản

Chữ cái đầu tiên và dòng đầu tiên của văn bản (:first-letter / :first-line)

Trả lời người dùng

* Hộp (box model)

+ Kiểm soát kích thước hộp (width / height)

+ Đường viền, lề và phần đệm của hộp (border / margin / padding)

+ Hiển thị và ẩn các hộp (display / visibility)

+ Thuộc tính mới của CSS3

Hình ảnh viền (border-image)

Bóng (border-shadow)

Góc tròn (border-radius)

* Danh sách, bảng và biểu mẫu: Danh sách dấu đầu dòng (list-style), Đường viền và nền của bảng (border-collapse), Sự xuất hiện của các điều khiển biểu mẫu, Căn chỉnh các điều khiển biểu mẫu, Công cụ phát triển trình duyệt.
* Hình ảnh: Kiểm soát kích thước của hình ảnh (display: inline-block), Căn chỉnh hình ảnh, Hình nền (background/background-image/background-repeat/background-position)
* Cách trình bày:

+ Kiểm soát vị trí của phần tử (position / z-index)

Luồng thông thường

Định vị tương đối

Định vị tuyệt đối

Định vị cố định

Phần tử nổi (float/clear)

+ Bố cục trang web

Bố cục HTML5

+ Thích ứng với kích thước màn hình

Bố trí chiều rộng cố định

Bố trí chất lỏng

Lưới bố trí

Xử lý hành vi tương tác bằng JavaScript

* Cú pháp cơ bản của JavaScript

+ Tuyên bố và nhận xét

+ Biến và kiểu dữ liệu

Tuyên bố và phân công

Kiểu dữ liệu đơn giản và kiểu dữ liệu phức tạp

Quy tắc đặt tên biến

+ Biểu thức và toán tử

Toán tử gán

Toán tử số học

Toán tử so sánh

Các toán tử logic: &&, ||,!

+ Cấu trúc nhánh

if...else...

switch...cas...default...

+ Cấu trúc vòng lặp

For vòng lặp

While vòng lặp

do...while vòng lặp

+ Mảng

Tạo mảng

Thao tác các phần tử trong mảng

+ Chức năng

Khai báo hàm

Chức năng gọi

Tham số và giá trị trả về

Chức năng ẩn danh

Chức năng gọi ngay lập tức

* Hướng đối tượng

+ Khái niệm đối tượng

+ Cú pháp nguyên văn để tạo đối tượng

+ Truy cập toán tử thành viên

+ Cú pháp xây dựng để tạo đối tượng

thisTừ khóa

+ Thêm và xóa thuộc tính

deleteTừ khóa

+ Đối tượng tiêu chuẩn

Number/ String/ Boolean/ Symbol/ Array/Function

Date/ Error/ Math/ RegExp/ Object/ Map/Set

JSON/ Promise/ Generator/ Reflect/Proxy

* BOM

+ window Thuộc tính và phương thức của đối tượng

+ history sự vật

forward()/ back()/go()

+ location sự vật

+ navigator sự vật

+ screen sự vật

* DOM

+ Cây DOM

+ Phần tử truy cập

getElementById()/querySelector()

getElementsByClassName()/getElementsByTagName()/querySelectorAll()

parentNode/ previousSibling/ nextSibling/ children/ firstChild/lastChild

+ Yếu tố vận hành

nodeValue

innerHTML/ textContent/ createElement()/ createTextNode()/ appendChild()/ insertBefore()/ removeChild()

className/ id/ hasAttribute()/ getAttribute()/ setAttribute()/removeAttribute()

+ Xử lý sự kiện

Loại sự kiện

Sự kiện giao diện người dùng : load// unload/ error/resizescroll

Sự kiện bàn phím : keydown//keyupkeypress

Sự kiện chuột: click/ dbclick/ mousedown/ mouseup/ mousemove/ mouseover/mouseout

Sự kiện trọng tâm: focus/blur

Sự kiện biểu mẫu: input// change/ submit/ reset/ cut/ copy/ paste/select

Ràng buộc sự kiện

Trình xử lý sự kiện HTML (không được khuyến nghị vì cần tách thẻ khỏi mã)

Trình xử lý sự kiện DOM truyền thống (chỉ có thể đính kèm một chức năng gọi lại)

Trình xử lý sự kiện (không được hỗ trợ trong các trình duyệt cũ hơn)

Luồng sự kiện: nắm bắt sự kiện/sự kiện sủi bọt

Đối tượng sự kiện (window.event trong các phiên bản IE thấp hơn)

Target (Một số trình duyệt sử dụng srcElement)

type

cancelable

preventDefault()

stopPropagation() (hủyBubble trong các phiên bản IE thấp hơn)

Sự kiện chuột - nơi xảy ra sự kiện

Vị trí màn hình: screenXvàscreenY

Vị trí trang: pageXvàpageY

Vị trí của khách hàng: clientXvàclientY

Sự kiện bàn phím - phím nào được nhấn

keyCodethuộc tính (được sử dụng bởi một số trình duyệt which)

String.fromCharCode(event.keyCode)

Sự kiện HTML5

DOMContentLoaded

hashchange

beforeunload

* API JavaScript:

+ Lưu trữ khách hàng – localStorage và sessionStorage

+ Nhận thông tin vị trí –geolocation

+ Tìm nạp dữ liệu từ máy chủ - Fetch API

+ Vẽ đồ họa - <canvas>API

+ Âm thanh và video- <audio>và <video>API

Sử dụng jQuery

* Tổng quan về jQuery

+ Write Less Do More (Hoàn thành nhiều công việc hơn với ít mã hơn)

+ Sử dụng bộ chọn CSS để tìm các phần tử (dễ dàng và thuận tiện hơn)

+ Sử dụng các phương thức jQuery để thao tác các phần tử (giải quyết các vấn đề tương thích với trình duyệt, áp dụng cho tất cả các phần tử và áp dụng nhiều phương thức)

* Tìm phần tử

+ Bộ chọn lọc

\* / element / #id / .class / selector1, selector2

ancestor descendant / parent>child / previous+next / previous~siblings

+ Lọc

Các bộ lọc cơ bản: :not(selector) / :first / :last / :even / :odd / :eq(index) / :gt(index) / :lt(index) / :animated / :focus

Bộ lọc nội dung::contains('…') / :empty / :parent / :has(selector)

Bộ lọc hiển thị: :hidden / :visible

Bộ lọc nút con: :nth-child(expr) / :first-child / :last-child / :only-child

Bộ lọc thuộc tính: [attribute] / [attribute='value'] / [attribute!='value'] / [attribute^='value'] / [attribute$='value'] / [attribute|='value'] / [attribute~='value']

Ghi chú: :input / :text / :password / :radio / :checkbox / :submit / :image / :reset / :button / :file / :selected / :enabled / :disabled / :checked

* Thực hiện các hoạt động

+ Thao tác nội dung

Nhận / sửa đổi nội dung: html() /// text() replace With() remove()

Nhận/ đặt các phần tử: before() // after()/ prepend()/ append()/ remove()/ clone()/ unwrap()/ / detach()/ empty() add()

Nhận/ sửa đổi thuộc tính : attr()// removeAttr()/ addClass()/ removeClass() css()

Nhận/đặt giá trị biểu mẫu: val()

+ Tìm hoạt động

Phương pháp tìm kiếm: find()/ parent()/ children()/ siblings()/ next()/ nextAll()/ prev()/ prevAll()

Bộ lọc: filter()/ not()/ has()/ is()/ contains()

Số chỉ mục:eq()

+ Kích thước và vị trí

Kích thước liên quan: height()/ width()/ innerHeight()/ innerWidth()/ outerWidth()/ outerHeight()

Liên quan đến vị trí: offset()/ position()/ scrollLeft()/scrollTop()

+ Hiệu ứng đặc biệt và hoạt hình

Hoạt hình cơ bản : show() // hide() toggle()

Biến mất và xuất hiện: fadeIn()/ fadeOut()/ fadeTo()/fadeToggle()

Hiệu ứng trượt : slideDown()//slideUp()slideToggle()

Tùy chỉnh : delay()//stop()animate()

+ Sự kiện

Đang tải tài liệu: ready()/load()

Tương tác người dùng: on()/off()

* Vận hành dây chuyền
* Kiểm tra xem trang có sẵn không
* Plugin jQuery

+ Xác thực jQuery

+ Chế độ xem dạng cây jQuery

+ Tự động hoàn thành jQuery

+ Giao diện người dùng jQuery

* Tránh xung đột với các thư viện khác

+ Giới thiệu các thư viện khác trước rồi mới giới thiệu jQuery.

+ Giới thiệu jQuery trước và sau đó là các thư viện khác.

* Sử dụng Ajax

Ajax là công nghệ có thể cập nhật các phần của trang web mà không cần tải lại toàn bộ trang web.

+ Ajax bản địa

+ Ajax dựa trên jQuery

Tải nội dung

Gửi biểu mẫu

KHUNG GIAO DIỆN NGƯỜI DÙNG

Khung lũy tiến – Vue.js: Một framework bắt buộc phải có để phát triển tách biệt front-end và back-end (kết xuất front-end).

* Bắt đầu nhanh chóng

+ Giới thiệu về tệp JavaScript của Vue, nên tải nó từ máy chủ CDN.

+ Liên kết dữ liệu (kết xuất khai báo).

+ Điều kiện và vòng lặp.

+ Thuộc tính tính toán.

+ Xử lý các sự kiện.

+ Đầu vào của người dùng.

+ Tải dữ liệu JSON qua mạng.

* Sử dụng vue-cli: vue-cli giúp ta có thể lưu lại các bước cấu hình thủ công môi trường phát triển, môi trường test và môi trường chạy, để các nhà phát triển chỉ cần tập trung vào các vấn đề cần giải quyết

+ Lắp đặt giàn giáo.

+ Tạo dự án.

+ Cài đặt các gói phụ thuộc.

+ Chạy dự án.

Khung giao diện người dùng – Element: Thư viện thành phần máy tính để bàn dựa trên Vue 2.0, được sử dụng để xây dựng giao diện người dùng và hỗ trợ bố cục đáp ứng.

* Giới thiệu các tệp CSS và JavaScript của Element.
* Sử dụng các thành phần.

Khung báo cáo – Echarts: Một thư viện trực quan hóa nguồn mở do Baidu sản xuất, thư viện này thường được sử dụng để tạo nhiều loại báo cáo khác nhau.

Khung CSS dựa trên Flexbox – Bulma: Bulma là một CSS framework hiện đại dựa trên Flexbox. Nó có thiết kế mô-đun và có thể dễ dàng sử dụng để triển khai nhiều bố cục nội dung đơn giản hoặc phức tạp.

Khung bố cục đáp ứng – Bootstrap: Khung giao diện người dùng để phát triển nhanh các ứng dụng web, hỗ trợ bố cục đáp ứng.

* Đặc trưng

+ Hỗ trợ các trình duyệt và thiết bị di động chính

+ Dễ dàng và khéo léo

+ Thiết kế đáp ứng

* Nội dung

+ Hệ thống lưới điện

+ CSS đóng gói

+ Linh kiện làm sẵn

+ Plugin JavaScript

* Hình dung