Câu 1:

**Ứng dụng của Python**

Python thường được sử dụng để phát triển trang web và phần mềm, tự động hóa tác vụ, phân tích dữ liệu và trực quan hóa dữ liệu. Vì tương đối dễ học, Python đã được nhiều người không phải là lập trình viên như kế toán và nhà khoa học áp dụng cho nhiều công việc hàng ngày, chẳng hạn như tổ chức tài chính.

**Phân tích dữ liệu và học máy**

Python đã trở thành một yếu tố chính trong khoa học dữ liệu, cho phép các nhà phân tích dữ liệu và các chuyên gia khác sử dụng ngôn ngữ này để thực hiện các phép tính thống kê phức tạp, tạo trực quan hóa dữ liệu, xây dựng thuật toán [học máy](https://glints.com/vn/blog/hoc-machine-learning/" \t "_blank), thao tác và phân tích dữ liệu cũng như hoàn thành các nhiệm vụ khác liên quan đến dữ liệu.

Python có thể xây dựng nhiều dạng trực quan hóa dữ liệu khác nhau, chẳng hạn như biểu đồ đường và thanh, biểu đồ hình tròn, biểu đồ 3D. Python cũng có một số thư viện cho phép các lập trình viên viết chương trình để phân tích dữ liệu và học máy nhanh hơn và hiệu quả hơn, như TensorFlow và Keras.

### ****Phát triển web****

Python thường được sử dụng để phát triển back-end của trang web hoặc ứng dụng—những phần mà người dùng không nhìn thấy. Vai trò của Python trong phát triển web có thể bao gồm gửi dữ liệu đến và đi từ máy chủ, xử lý dữ liệu và giao tiếp với cơ sở dữ liệu, định tuyến URL và đảm bảo tính bảo mật. Python cung cấp một số khuôn khổ để phát triển web. Những cái thường được sử dụng bao gồm Django và Flask.

Một số công việc phát triển web sử dụng Python bao gồm kỹ sư phụ trợ, nhà phát triển Python, kỹ sư phần mềm và kỹ sư DevOps.

### ****Tự động hoá và phát triển phần mềm****

Nếu bạn thấy mình thực hiện một nhiệm vụ lặp đi lặp lại, bạn có thể làm việc hiệu quả hơn bằng cách tự động hóa nó bằng Python. Quá trình viết code được sử dụng để xây dựng các quy trình tự động này được gọi là viết script. Trong thế giới mã hóa, tự động hóa có thể được sử dụng để kiểm tra lỗi trên nhiều tệp, chuyển đổi tệp, thực hiện phép toán đơn giản và loại bỏ các bản sao trong dữ liệu.

Python thậm chí có thể được sử dụng bởi những người mới bắt đầu để tự động hóa các tác vụ đơn giản trên máy tính—chẳng hạn như đổi tên tệp, tìm và tải xuống nội dung trực tuyến hoặc gửi email hoặc văn bản theo khoảng thời gian mong muốn. Trong phát triển phần mềm, Python có thể hỗ trợ các tác vụ như kiểm soát bản dựng, theo dõi lỗi và thử nghiệm. Với Python, các nhà phát triển phần mềm có thể tự động kiểm tra các sản phẩm hoặc tính năng mới. Một số công cụ Python được sử dụng để kiểm thử phần mềm bao gồm Green và Requestium.

Source: <https://glints.com/vn/blog/ngon-ngu-lap-trinh-python-la-gi/#:~:text=Python%20th%C6%B0%E1%BB%9Dng%20%C4%91%C6%B0%E1%BB%A3c%20s%E1%BB%AD%20d%E1%BB%A5ng,nh%C6%B0%20t%E1%BB%95%20ch%E1%BB%A9c%20t%C3%A0i%20ch%C3%ADnh>.

Câu 2:

|  |  |
| --- | --- |
| CPP | PYTHON |
| Hầu hết mọi hoạt động của C++ đều được thực hiện dưới dạng ngôn ngữ biên dịch (Compiler), nên có thể chạy trên đa hệ điều hành. | Python là một ngôn ngữ thông dịch, điều này nghĩa là ngôn ngữ này trực tiếp chạy từng dòng mã. Nếu có lỗi trong mã chương trình, nó sẽ ngừng chạy. Do đó, lập trình viên có thể nhanh chóng tìm ra lỗi trong đoạn mã. |
| C++: C++ có cú pháp phức tạp hơn, với việc sử dụng dấu chấm phẩy và ngoặc nhọn để định rõ khối mã. | Python: Python có cú pháp đơn giản và dễ đọc, với việc sử dụng khoảng trắng để định rõ khối mã. Điều này giúp làm giảm số lỗi cú pháp và tạo mã dễ đọc hơn. |
| Các lập trình viên không cần phải khai báo loại biến khi viết mã bởi vì Python sẽ xác định chúng vào thời điểm chạy. Vì vậy, bạn có thể viết các chương trình Python một cách nhanh chóng hơn. | C++ là ngôn ngữ tĩnh kiểu, cần phải xác định kiểu dữ liệu khi khai báo biến. Điều này giúp kiểm soát kiểu và hiệu năng tốt hơn, nhưng đôi khi có thể cần nhiều công sức hơn trong việc quản lý kiểu. |
| Python là ngôn ngữ bậc cao gần gũi với ngôn ngữ con người hơn các ngôn ngữ lập trình khác. Do đó, các lập trình viên không cần phải lo lắng về những chức năng cơ bản của nó như kiến trúc và quản lý bộ nhớ. | C++ là một ngôn ngữ tầm trung, bạn hoàn toàn có thể sử dụng nó để phát triển các chương trình bậc thấp hay những chương trình bậc cao, mà vẫn hoạt động tốt trong phần cứng. |
| Python coi mọi thứ đều là đối tượng, nhưng ngôn ngữ này cũng hỗ trợ các phương thức lập trình khác như lập trình hàm và lập trình cấu trúc. | C++ là ngôn ngữ lập trình hướng đối tượng, sử dụng các Class và Object cùng các khái niệm như tính kế thừa, tính đa hình, tính đóng gói... để tạo ra các chương trình. |

Source: <https://tuhoclaptrinh.edu.vn/bai-viet/tim-hieu-ve-ngon-ngu-lap-trinh-c-422.html>

<https://aws.amazon.com/vi/what-is/python/#:~:text=Python%20l%C3%A0%20m%E1%BB%99t%20ng%C3%B4n%20ng%E1%BB%AF%20th%C3%B4ng%20d%E1%BB%8Bch%2C%20%C4%91i%E1%BB%81u%20n%C3%A0y%20ngh%C4%A9a,ra%20l%E1%BB%97i%20trong%20%C4%91o%E1%BA%A1n%20m%C3%A3>.

Câu 3:

Python là một ngôn ngữ thông dịch điển hình, vì vậy việc chạy các chương trình Python cần sự hỗ trợ của trình thông dịch. Chỉ cần bạn cài đặt các trình thông dịch khác nhau trên các nền tảng khác nhau, mã của bạn có thể chạy bất cứ lúc nào mà không cần lo lắng về bất kỳ vấn đề tương thích nào. Viết, chạy ở mọi nơi.

Mỗi khi chương trình được thực thi, nó cần được chuyển đổi và thực thi, mã nguồn được chuyển đổi thành mã máy nếu nó được sử dụng, và không có quá trình xử lý nào được thực hiện trên mã nguồn không sử dụng. Các chức năng khác nhau có thể được sử dụng mỗi khi chương trình được thực thi và mã nguồn cần được chuyển đổi sẽ khác nhau tại thời điểm này.

Bởi vì mã nguồn cần được chuyển đổi lại mỗi khi chương trình được thực thi, hiệu quả thực thi của các ngôn ngữ thông dịch vốn đã thấp hơn so với các ngôn ngữ biên dịch, và thậm chí còn có khoảng cách về độ lớn.

Source: <https://anonystick.com/blog-developer/so-sanh-thong-dich-va-bien-dich-2021061255936853>

Bài 5:

Hàm input có cú pháp như sau

input(prompt=None)

Có lúc bạn sẽ nhìn thấy cú pháp của nó là input(prompt=None,**/**). Cái phần thêm vào là kí tự**/**chỉ là một kí tự cho biết parameter prompt chỉ nhận giá trị dưới dạng**positional argument**. Nghĩa là khi bạn truyền vào cho hàm, bạn không được phép điền thêm chữ prompt.

Các loại kiểu dữ liệu

Những kiểu dữ liệu trong Python

Number

String

List

Tuple

Dictionary

1.Number

Number là một nhóm dữ liệu thường dùng trong Python, nó dùng để lưu trữ hầu hết các kiểu về số như:

Int # kiểu số nguyên vd: 1, 0, -2

Float # kiểu số thực vd: 1.3, -3.4

Complex # kiểu số phức vd: 3+4j

Để nhập kiểu số nguyên từ bàn phím ta làm như sau:

*# cú pháp nhập vaò 1 số nguyên*

a*=*int(input())

*# in ra a*

print(a)

Tương tự với số thực hay số phức ta chỉ cần thay đổi int thành float hoặc complex

2. String

String trong python sẽ biểu thị cho các chuỗi và được bao quanh bởi dấu ngoặc đơn hoặc

kép. Python cũng không có kiểu dữ liệu kí tự char (như trong c/c++)

String trong python là bất biến… chúng ta không thể thay đổi các kí tự trong string

3.List

List là kiểu dữ liệu tích hợp sẵn trong Python được sử dụng để lưu trữ các bộ dữ liệu.

List được sử dụng để lưu trữ nhiều items trong một biến duy nhất.

lst*=*[1,2,3,4,5,"haha",["hihi",29022004]]

4.Tuple

Tuple tương tự với list nhưng các items được sắp xếp theo thứ tự và không thểthay đổi.

Tuples được viết bằng dấu ngoặc tròn.

tpl*=*(1,2,3,4,5,"haha",["hihi",29022004])

5.Dictionary

Dictionary là một tập hợp được sắp xếp theo thứ tự , có thể thay đổi và không chophép trùng lặp.

Dictionary được sử dụng để lưu trữ các giá trị dữ liệu trong các cặp key: value.

dict*=*{

    "id":2022,

    "hovaten":"abc",

    "dtb":3.79

}

Các toán tử cơ bản trong python

1.Toán tử số học

|  |  |  |
| --- | --- | --- |
| Operator | Name | Example |
| + | Addition | x+y |
| - | Subtraction | x-y |
| \* | multiply | x\*y |
| / | Division | x/y |
| % | Modulus | x%y |
| \*\* | Exponentiation | x\*\*y |
| // | Floor division | x//y |

2. Toán tử so sánh

|  |  |  |
| --- | --- | --- |
| Operator | Name | Example |
| == | equal | x==y |
| != | not equal | x!=y |
| > | greater than | x<y |
| >= | greater or equal to | x>=y |
| <= | less than or equal to | x<=y |
| \*\* | Exponentiation | x\*\*y |
| // | Floor division | x//y |

3. Toán tử logic

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| and | return true when both are true | x<5 and x<10 |
| or | return true when one is true | x<5 or x<4 |
| not | reverse the result | not(x<5 and x<10) |
| >= | greater or equal to | x>=y |
| <= | less than or equal to | x<=y |
| \*\* | Exponentiation | x\*\*y |
| // | Floor division | x//y |

4. Toán tử thành viên (Membership Operators)

in

# Trả về true nếu biến đó ở trong một chuỗi, list,… chỉ định.

not in

#ngược lại với in

5.Toán tử gán (Assignment Operators)

= # Gán …

Các toán tử gán khác là: +=, -=, \*=, /=, %=,\*\*=, //=

6. Toán tử nhận dạng (Identity Operators)

is

# Trả về true nếu 2 biến ta so sánh có cùng trỏ đến một ô nhớ

is not

# Ngược lại vs is

7.Toán tử Bitwise (Bitwise Operators)

|  |  |
| --- | --- |
| Toán Tử | Ví dụ |
| & (Bitwise AND) | (a & b) = 12 (00001100) |
| | (Bitwise OR) | (a | b) = 14 (00001111) |
| ^ (Bitwise XOR) | (a ^ b) = 2 (00000010) |
| ~ (Bitwise NOT) | (-a) = -13 (00001101) |
| << (left shift) | a<<a = 49152 |
| >> (right shift) | a>>a = 0 |

(code demo trong file khác ))