Biến và những kiểu dữ liệu đơn giản

Nội dung giới thiệu

- Giới thiệu chung về biến (variables)
- Chuỗi (Strings)
- Số (Numbers)
- Chú thích (comments)

Biến

Thêm một dòng ở đầu tệp và sửa dòng thứ hai:

```
message = "Hello Python world!"
print(message)
```

```
Hello Python world!
```

Mở rộng chương trình với việc chỉnh sửa hello_world.py, để in ra thông điệp thứ hai. Thêm dòng trống vào tệp hello_world.py và thêm hai dòng code mới:

```
message = "Hello Python world!"
print(message)
message = "Hello Python Crash Course
world!"
print(message)
```

```
Hello Python world!
Hello Python Crash Course world!
```

Định danh và sử dụng các biến

Quy tắc và một số hướng dẫn:

- Tên biến chỉ có thể chứa các chữ cái, số và dấu gạch dưới. Chúng có thể bắt đầu bằng một chữ cái hoặc một dấu gạch dưới, nhưng không phải bằng một số.
- Không được phép sử dụng dấu cách trong tên biến, nhưng có thể sử dụng dấu gạch dưới để tách các từ trong tên biến.
- Tránh sử dụng các từ khóa Python và tên hàm làm tên biến
- Tên biến phải ngắn nhưng mang tính mô tả.
- Cẩn thận khi sử dụng chữ cái viết thường l và chữ cái viết hoa O vì chúng có thể bị nhầm lẫn với số 1 và số 0


message_1


1_message.


welcome_message


welcome message


print

student_name

name_length

s_n

length_of_persons_name

Tránh đặt tên lỗi khi sử dụng các biến

- Viết mã sau, bao gồm thông báo từ sai chính tả được in đậm:

```
message = "Hello Python Crash Course reader!"

print(mesage)
```

- Dấu vết (traceback) là một bản ghi về nơi trình thông dịch gặp sự cố khi cố gắng thực thi code đã viết.

```
Traceback (most recent call last):
① File "hello_world.py", line 2, in <module>
② print(mesage)
③ NameError: name 'mesage' is not defined
```

Ví dụ

- Xem xét trường hợp sau:

```
mesage = "Hello Python Crash Course reader!"
print(mesage)
```

- Kết quả:

```
Hello Python Crash Course reader!
```

Biến là các nhãn

- Các biến thường được mô tả như các hộp có thể lưu trữ các giá trị. Ý tưởng này có thể hữu ích khi mới sử dụng một vài biến, nhưng đó không phải là cách chính xác để mô tả cách các biến được biểu diễn bên trong bằng Python.
- Cách tốt hơn là nên coi các biến dưới dạng nhãn mà chúng ta có thể gán cho các giá trị. Ta có thể cũng nói rằng một biến tham chiếu đến một giá trị nhất định.

Chuỗi (Strings)

- Một chuỗi là một chuỗi các ký tự. Mọi thứ bên trong dấu ngoặc kép đều được xem xét một chuỗi với Python và ta có thể sử dụng dấu ngoặc đơn hoặc dấu ngoặc kép xung quanh chuỗi :

```
"This is a string."
'This is also a string.'
```

- Sự linh hoạt này cho phép ta sử dụng dấu nháy đơn hoặc nháy kép trong chuỗi.

```
'I told my friend, "Python is my favorite language!"'
"The language 'Python' is named after Monty Python, not the snake."
"One of Python's strengths is its diverse and supportive community."
```

Sử dụng chữ hoa trong chuỗi với các phương thức

- Phương thức title() thay đổi chữ thường thành chữ hoa như tiêu đề, do đó mỗi ký tự đầu chữ sẽ thành chữ hoa.

```
name = "ada lovelace"
print(name.title())
```

```
Ada Lovelace
```

- Phương thức upper() đổi hết thành chữ hoa
- Phương thức lower() đổi hết thành chữ thường

```
name = "Ada Lovelace"
print(name.upper())
print(name.lower())
```

```
ADA LOVELACE
ada lovelace
```

Sử dụng biến trong chuỗi

- Sử dụng giá trị của một biến bên trong một chuỗi. (f-strings)

```
first_name = "ada"
last_name = "lovelace"
①   full_name = f"{first_name} {last_name}"
print(full_name)
```

```
ada lovelace
```

- Sử dụng f-string để tạo thông điệp hoàn chỉnh

```
first_name = "ada"
last_name = "lovelace"
full_name = f"{first_name} {last_name}"
print(f"Hello, {full_name.title()}!")
```

```
Hello, Ada Lovelace!
```

- Sử dụng f-string để soạn thông điệp, sau đó gán toàn bộ thông điệp cho một biến

```
first_name = "ada"
last_name = "lovelace"
full_name = f"{first_name} {last_name}"
message = f"Hello, {full_name.title()}!"
print(message)
```

```
Hello, Ada Lovelace!
```

Thêm khoảng trắng vào chuỗi với Tab hoặc dòng mới

- Để thêm Tab vào chuỗi, sử dụng tổ hợp ký tự: \t

```
>>> print("Python")
Python
>>> print("\tPython")
    Python
```

- Để thêm một dòng mới trong một chuỗi, hãy sử dụng tổ hợp ký tự \ n

```
>>> print("Languages:\nPython\nC\nJavaScript")
Languages:
Python
C
JavaScript
```

Bỏ khoảng trắng

- Khoảng trắng thừa có thể gây nhầm lẫn, do cần loại bỏ trước khi thực hiện các tác vụ.
- Python giúp loại bỏ dễ dàng khoảng trắng thừa từ dữ liệu người dùng nhập vào bằng phương thức strip()

```
>>> favorite_language = 'python '
>>> favorite_language
'python '
>>> favorite_language.rstrip()
'python'
>>> favorite_language
'python '
```

- Lưu ý: Để xóa khoảng trắng vĩnh viễn, ta thực hiện phép gán:

```
>>> favorite_language = 'python '
>>> favorite_language = favorite_language.rstrip()
>>> favorite_language
'python'
```

Bỏ khoảng trắng (t)

- Xóa khoảng trắng bên phải: rstrip()
- Xóa khoảng trắng bên trái: lstrip()

```
>>> favorite_language = ' python '
>>> favorite_language.rstrip()
' python'
>>> favorite_language.lstrip()
'python '
>>> favorite_language.strip()
'python'
```

Tránh lỗi cú pháp với String

- Cách sử dụng chính xác dấu nháy kép và dấu nháy đơn:

```
message = "One of Python's strengths is its diverse community."
print(message)
```

```
One of Python's strengths is its diverse community.
```

- Nếu sử dụng dấu nháy đơn, Python không thể xác định vị trí chuỗi nên kết thúc:

```
message = 'One of Python's strengths is its diverse community.'
print(message)
```

```
File "apostrophe.py", line 1
message = 'One of Python's strengths is its diverse community.'
    SyntaxError: invalid syntax
```

Bài tập

TRY IT YOURSELF

Write a separate program to accomplish each of these exercises. Save each program with a filename that follows standard Python conventions, using lowercase letters and underscores, such as *simple_message.py* and *simple_messages.py*.

2-1. Simple Message: Assign a message to a variable, and then print that message.

2-2. Simple Messages: Assign a message to a variable, and print that message. Then change the value of the variable to a new message, and print the new message.

Số - Numbers

- Các con số được sử dụng khá thường xuyên trong lập trình để lưu điểm số trong trò chơi, đại diện cho dữ liệu trong trực quan hóa, lưu trữ thông tin trong các ứng dụng web và các ứng dụng khác. Python xử lý các số theo các cách khác nhau, dựa trên việc các số được sử dụng như thế nào.

Số nguyên - Integers

- Có thể cộng (+), trừ (-), nhân (*) và chia (/) số nguyên trong Python

```
>>> 2 + 3
5
>>> 3 - 2
1
>>> 2 * 3
6
>>> 3 / 2
1.5
```

- Trong một phiên đầu cuối, Python trả về kết quả của toán tử. Python sử dụng hai ký hiệu nhân để biểu diễn số mũ:

```
>>> 3 ** 2
9
>>> 3 ** 3
27
>>> 10 ** 6
1000000
```

Số nguyên – Integers (t)

- Python cũng hỗ trợ thứ tự các phép toán, có thể sử dụng nhiều các phép toán trong một biểu thức.

```
>>> 2 + 3*4
14
```

- Có thể sử dụng dấu ngoặc đơn để sửa đổi thứ tự của các phép toán để Python có thể đánh giá biểu thức theo thứ tự được chỉ định

```
>>> (2 + 3) * 4
20
```

Số thực - Floats

- Python gọi bất kỳ số nào có dấu thập phân là số thực (float)

- Chỉ cần nhập các số cần sử dụng và Python sẽ xử lý số đúng như mong đợi

```
>>> 0.1 + 0.1
0.2
>>> 0.2 + 0.2
0.4
>>> 2 * 0.1
0.2
>>> 2 * 0.2
0.4
```

Số nguyên và số thực

- Khi chia hai số bất kỳ, ngay cả khi chúng là số nguyên dẫn đến kết quả đều sẽ là số thực

```
>>> 4/2
2.0
```

- Nếu kết hợp số nguyên và số thực trong một phép toán, kết quả sẽ nhận được là số thực

```
>>> 1 + 2.0
3.0
>>> 2 * 3.0
6.0
>>> 3.0 ** 2
9.0
```

Các gạch dưới trong số

- Khi viết các số dài, ta có thể nhóm các chữ số bằng dấu gạch dưới để làm cho các số lớn dễ đọc hơn:

```
>>> universe_age = 14_000_000_000
```

- Khi in ra số dùng gạch dưới, Python chỉ in ra các số

```
>>> print(universe_age)
14000000000
```

Gán nhiều biến cùng lúc

- Gán giá trị cho nhiều biến chỉ bằng một dòng duy nhất, điều này làm cho chương trình ngắn hơn và dễ đọc hơn

- Ví dụ: đây là cách khởi tạo các biến x, y và z về 0

```
>>> x, y, z = 0, 0, 0
```

- Cần tách các tên biến bằng dấu phẩy và thực hiện tương tự với các giá trị và Python sẽ chỉ định từng giá trị tương ứng với biến theo vị trí

Cách viết các chú thích

- Trong Python, dấu thăng (#) chỉ ra một chú thích. Bất cứ điều gì theo sau dấu thăng trong mã bị trình thông dịch Python bỏ qua.

```
# Say hello to everyone.
print("Hello Python people!")
```

- Python sẽ bỏ qua dòng đầu tiên vì có gặp dấu #, chỉ thực thi dòng thứ hai

```
Hello Python people!
```

Loại chú thích nên viết

- Giải thích mã nguồn để làm gì và cách làm cho mã nguồn hoạt động
- Viết rõ ràng, ngắn gọn nhận xét trong mã nguồn
- Cách tiếp cận để tìm ra mã nguồn hợp lý

Bài tập

2-8. Number Eight: Write addition, subtraction, multiplication, and division operations that each result in the number 8. Be sure to enclose your operations in `print()` calls to see the results. You should create four lines that look like this:

```
print(5+3)
```

Your output should simply be four lines with the number 8 appearing once on each line.

2-9. Favorite Number: Use a variable to represent your favorite number. Then, using that variable, create a message that reveals your favorite number. Print that message.

Bài tập

Save each of the following exercises as a separate file with a name like *name_cases.py*. If you get stuck, take a break or see the suggestions in Appendix C.

2-3. Personal Message: Use a variable to represent a person's name, and print a message to that person. Your message should be simple, such as, "Hello Eric, would you like to learn some Python today?"

2-4. Name Cases: Use a variable to represent a person's name, and then print that person's name in lowercase, uppercase, and title case.

2-5. Famous Quote: Find a quote from a famous person you admire. Print the quote and the name of its author. Your output should look something like the following, including the quotation marks:

> *Albert Einstein once said, "A person who never made a mistake never tried anything new."*

Bài tập

2-6. Famous Quote 2: Repeat Exercise 2-5, but this time, represent the famous person's name using a variable called `famous_person`. Then compose your message and represent it with a new variable called `message`. Print your message.

2-7. Stripping Names: Use a variable to represent a person's name, and include some whitespace characters at the beginning and end of the name. Make sure you use each character combination, `"\t"` and `"\n"`, at least once.

Print the name once, so the whitespace around the name is displayed. Then print the name using each of the three stripping functions, `lstrip()`, `rstrip()`, and `strip()`.

XIN CẢM ƠN!