# THÀNH PHẦN

## khai báo biến trong Python

Trong bài học này chúng ta sẽ tìm hiểu cách khai báo biến trong Python, qua bài này bạn sẽ biết cách tạo và sử dụng biến cũng như gán dữ liệu cho biến bằng Python.

Nhắc lại về cú pháp thì Python là một ngôn ngữ đặc biệt, bạn không cần phải đặt dấu chấm phẩy (;) đằng sau mỗi dòng lệnh. Điều này đôi khi khiến những người học từ C/ C++ sang cảm thấy bỡ ngỡ.

Tuy nhiên bạn đừng lo lắng, bởi khi học Python thì bạn sẽ phải trình bày một cách sáng sủa về vị trí, khoảng trắng và xuống dòng. Nói thì dài dòng, sau đây ta sẽ học cách khai báo biến.

**Mục lục**

* [1. Biến trong Python là gì?](https://freetuts.net/cach-khai-bao-bien-trong-python-3422.html#goto-h2-0)
* [2. Cách khai báo biến trong Python](https://freetuts.net/cach-khai-bao-bien-trong-python-3422.html#goto-h2-1)
* [3. Cách thay đổi dữ liệu cho biến Python](https://freetuts.net/cach-khai-bao-bien-trong-python-3422.html#goto-h2-2)
* [4. Xóa biến trong Python](https://freetuts.net/cach-khai-bao-bien-trong-python-3422.html#goto-h2-3)

### Biến trong Python là gì?

Trong Python, biến là một định danh dùng để lưu trữ dữ liệu trong quá trình biên dịch mã code Python.

Biến đóng vai trò rất quan trọng, nếu không có biến thì bạn không thể tạo ra một chương trình hoàn hảo, nói cách khác bạn không thể giải quyết bất kì một bài toán nào cả, kể cả một ứng dụng đơn giản.

Khái niệm về biến không phải chỉ tồn tại ở Python, mà ngay cả những ngôn ngữ khác cũng có, vì vậy đây là bài đầu tiên mà bạn phải học khi tìm hiểu một ngôn ngữ lập trình bất kì.

### Cách khai báo biến trong Python

Để khai báo biến thì ta sử dụng cú pháp như sau:

|  |  |
| --- | --- |
| 1 | tenbien = giatri |

Điều đặc biệt ở trong Python là bạn phải gán một giá trị trong lúc khai báo biến, điều đó sẽ giúp Python nhận biết được biến đó sẽ lưu trữ kiểu dữ liệu gì.

**Ví dụ**: Khai báo biến domain.

|  |  |
| --- | --- |
| 1 | domain = "freetuts.net" |

Để in giá trị của biến ra màn hình thì ta sử dụng lệnh **print**.

|  |  |
| --- | --- |
| 1  2 | domain = "freetuts.net"  print(domain) |

### Cách thay đổi dữ liệu cho biến Python

Để thay đổi dữ liệu cho biến thì bạn chỉ việc sử dụng toán tử gán (ta sẽ được học ở bài tiếp theo).

|  |  |
| --- | --- |
| 1  2  3  4 | domain = "freetuts.net"  print(domain)  domain = "[https://freetuts.net](https://freetuts.net/)"  print(domain) |

Chạy chương trình ta thu được kết quả như sau:



### Xóa biến trong Python

Để xóa một biến trong Python để giải phóng bộ nhớ thì bạn sử dụng lệnh **del**.

Ví dụ dưới đây mình đã xóa biến domain nên dòng lệnh print sẽ bị lỗi.

|  |  |
| --- | --- |
| 1  2  3 | domain = "freetuts.net"  del domain  print(domain) |

Như vậy là mình đã hướng dẫn xong cách khai báo biến, cũng như cách thay đổi dữ liệu cho biến trong Python. Qua bài này bạn đã hiểu được quá trình chạy của một chương trình Python rồi đấy. Hẹn gặp lại bạn ở bài tiếp theo nhé.

## CHÚ THÍCH TRONG PYTHON

Trong bài này mình sẽ hướng dẫn các bạn cách tạo **comment trong Python**, đây là cách giúp bạn tạo ra những ghi chú trong lúc lập trình với ngôn ngữ Python.

Hầu hết các ngôn ngữ lập trình đều hỗ trợ tạo ghi chú chứ không riêng gì Python, vấn đề là cú pháp của mỗi ngôn ngữ sẽ khác nhau.

**Mục lục**

* [1. Comment trong Python là gì?](https://freetuts.net/comment-trong-python-195.html#goto-h2-0)
* [2. Cách tạo comment trong Python](https://freetuts.net/comment-trong-python-195.html#goto-h2-1)
  + [Tạo trên một dòng](https://freetuts.net/comment-trong-python-195.html#goto-h3-0)
  + [Tạo trên nhiều dòng](https://freetuts.net/comment-trong-python-195.html#goto-h3-1)

### Comment trong Python là gì?

Comment hay còn gọi là ghi chú, đây là những đoạn code mà trình biên dịch Python sẽ bỏ qua trong lúc chạy mã code.

Khi bạn phát triển dự án khổng lồ, hoặc bạn xây dựng một hàm nào đó thì cần lưu lại những lưu ý để sau này xem lại sẽ hiểu được những gì mà bạn đã code. Lúc này sử dụng cú pháp tạo comment sẽ giúp bạn giải quyết được vấn đề đó.

*Bài viết này được đăng tại [free tuts .net]*

### Cách tạo comment trong Python

Chúng ta có hai cách để tạo comment, đó là tạo trên một dòng và tạo trên nhiều dòng.

#### Tạo trên một dòng

Để tạo trên một dòng thì ta sử dụng dấu thăng #.

|  |  |
| --- | --- |
| 1 | # Nội dung comment |

Sau đây là một vài cách sử dụng.

|  |  |
| --- | --- |
| 1  2 | # Comment 1  print ("Say Hi!") # Comment 2 |

#### Tạo trên nhiều dòng

Tạo nhiều dòng dùng cho trường hợp bạn muốn comment trên hai dòng, lúc này có hai giải pháp cho bạn.

Một là sử dụng nhiều dấu thăng.

|  |  |
| --- | --- |
| 1  2  3 | # Dòng 1  # Dong 2  # Dòng 3 |

Hai là sử dụng dấu ba chấm để bao quanh các dòng lại.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | """  Đây là nội dung  comment  trên nhiều dòng  """  print("Hi Freetuts!") |

Việc tạo comment sẽ giúp chương trình dễ đọc hơn.

Bản thân mình cũng vậy, nhiều lúc quên comment và hôm sau xem lại code phải mất một thời gian đọc lại từ đầu. Vì vậy hãy tập thói quen comment khi lập trình với Python nhé.

## Các kiểu dữ liệu trong Python

Trong bài này mình sẽ giới thiệu các kiểu dữ liệu trong Python, qua bài này bạn sẽ hiểu được cách sử dụng các kiểu dữ liệu thường dùng nhất của Python.

Nói về kiểu dữ liệu thì Python rất đặc biệt, nó có rất nhiều kiểu dữ liệu về danh sách (collection), tuy nhiên trong bài này mình sẽ không trình bày về những kiểu phức tạp đó mà chỉ nói đến những kiểu đơn giản thôi nhé.

Về cơ bản thì trong Python có các kiểu dữ liệu chuẩn như sau: Numbers, String, List, Tuple, Dictionary.

**Mục lục**

* [1. Kiểu dữ liệu Numbers trong Python](https://freetuts.net/cac-kieu-du-lieu-trong-python-3421.html#goto-h2-0)
* [2. Kiểu dữ liệu String trong Python](https://freetuts.net/cac-kieu-du-lieu-trong-python-3421.html#goto-h2-1)
* [3. Kiểu dữ liệu List trong Python](https://freetuts.net/cac-kieu-du-lieu-trong-python-3421.html#goto-h2-2)
* [4. Kiểu dữ liệu Tuple trong Python](https://freetuts.net/cac-kieu-du-lieu-trong-python-3421.html#goto-h2-3)
* [5. Kiểu dữ liệu Dictionary trong Python](https://freetuts.net/cac-kieu-du-lieu-trong-python-3421.html#goto-h2-4)

### Kiểu dữ liệu Numbers trong Python

Kiểu dữ liệu number dùng để lưu trữ những con số, và một biến được xác đinh là kiểu number khi bạn gán dữ liệu cho nó là một number.

Ví dụ dưới đây mình khai báo hai biến age và counter là kiểu number.

|  |  |
| --- | --- |
| 1  2 | age = 30  counter = 60 |

Kiểu number được phân tác ra nhiều loại nhỏ khác nhau như: Các số nguyên, số phẩy động ...

* int (số nguyên)
* long (số nguyên lớn)
* float (số thực)
* complex (complex numbers)

Xem thêm kiểu [number trong Python](https://freetuts.net/kieu-du-lieu-number-trong-python-1709.html).

### Kiểu dữ liệu String trong Python

Kiểu chuỗi (string) trong Python được hiểu là tập hợp của nhiều ký tự ghép lại với nhau như một mảng.

Dữ liệu chuỗi được bao quanh trong **dấu nháy đơn** hoặc **dấu nháy kép**.

|  |  |
| --- | --- |
| 1  2 | string1 = "freetuts.net"  string2 = 'cuong' |

Vì được xem là tập hợp của nhiều ký tự nên mỗi ký tự trong chuỗi được đánh một số chỉ mục và bắt đầu tư 0.

* Ký tự đầu tiên có chỉ mục = 0
* Ký tự cuối cùng có chỉ mục bằng tổng ký tự trừ đi 1.

Ngoài ra bạn có thể sử dụng các toán tử sau trên chuỗi:

* Bạn có thấy sử dụng cặp ngoặc vuông [] để lấy các ký tự trong chuỗi.
* Sử dụng dấu sao \* để lặp lại chuỗi.
* Sử dụng dấu cộng + để nối chuỗi.

**Ví dụ**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | str = 'Hello World!'    print(str)          # In ra toàn bộ chuỗi  print(str[0])       # In phân tử đầu tiên của chuỗi  print(str[2:5])     # In phần tử thứ 3 đến thứ 6, tại vì chỉ mục bắt đầu từ 0  print(str[2:])      # In từ vị trí thứ 3 đến cuối chuỗi  print(str \* 2)      # In chuỗi hai lần  print(str + "TEST") # Nối chuỗi |

Xem thêm kiểu [string trong Python](https://freetuts.net/kieu-du-lieu-string-trong-python-1711.html).

### Kiểu dữ liệu List trong Python

Trong Python thì list có thể được xem là một mảng danh sách các phần tử được sắp xếp với nhau theo một trật tự. Phần tử đầu tiên được đánh dấu là 0, phần tử cuối cùng được đánh dấu là tổng số phần tử trừ đi một.

Để khai báo kiểu List thì ta sử dụng cặp ngoặc vuông [], và mỗi phần tử sẽ được ngăn cách nhau bởi dấu phẩy.

**Ví dụ**

|  |  |
| --- | --- |
| 1 | domains = ["freetuts.net", "kephimonline.com", "zip.freeetuts.net"] |

Kiểu dữ liệu của List không nhất thiết là phải giống nhau. Như ví dụ dưới đây mình vừa khai báo kiểm number vừa kiểu string cho hai phần tử.

|  |  |
| --- | --- |
| 1 | infor = ["cuong", 31] |

Quay trở lại ví dụ ở phần 2 kiểu chuỗi, thực tế thì chuỗi trong Python được lưu trữ thành dạng List.

Xem kỹ hơn tại bài [List trong Python](https://freetuts.net/list-mang-trong-python-1690.html).

### Kiểu dữ liệu Tuple trong Python

Tuple cũng là một kiểu dữ liệu dạng danh sách giống như kiểu List, điểm khác biệt duy nhất là Tuple sử dụng **dấu ngoặc đơn**, còn List sử dụng **dấu ngoặc vuông**.

Ví dụ dưới đây mình khai báo một kiểu Tuple.

|  |  |
| --- | --- |
| 1 | domains = ("freetuts.net", "kephimonline.com", "zip.freeetuts.net") |

Điểm khác biệt thứ hai nữa đó là dữ liệu trong Tuple không thể thay đổi, nó giống như một hằng số vậy. Tuy nhiên nếu một phần tử của Typle là kiểu List thì ta có thể thay đổi dữ liệu cho phần tử đó.

Xem thêm tại bài [Tuple trong Python](https://freetuts.net/kieu-du-lieu-tuple-trong-python-1710.html).

### Kiểu dữ liệu Dictionary trong Python

Kiểu từ điển (dictionary) là một loại bảng băm (hash table), nó sẽ lưu trữ dữ liệu ở dạng key => value nên việc truy xuất cực kì dễ dàng.

Nếu bạn đã từng học qua PHP thì có thể thấy nó giống với kiểu [mảng kết hợp](https://freetuts.net/cac-kieu-du-lieu-trong-php-3.html), nó cũng giống với cấu trúc của [JSON](https://freetuts.net/csdl-can-ban/hoc-json) và [Object trong Javascript](https://freetuts.net/doi-tuong-object-trong-javascript-408.html).

**Ví dụ**: Khai báo thông tin cá nhân của mình.

|  |  |
| --- | --- |
| 1 | infor = {'name': 'Cường','domain': 'freetuts.net', 'age': 31} |

Vì các key được đánh theo số hoặc chuỗi nên việc sắp xếp theo đúng thứ tự trong dictionary là không có. Nói cách khác là các phần tử trong dictionary là không có thứ tự.

Xem thêm kiểu dữ liệu [Dictoinary trong Python](https://freetuts.net/dictionary-trong-python-1961.html).

Trên phần tổng hợp một số kiểu dữ liệu trong Python, chúc bạn học thật tốt nhé.

## Cách ép kiểu dữ liệu trong Python

Trong bài này mình sẽ hướng dẫn các bạn cách chuyển đổi và ép kiểu trong Python, đây là thao tác thường được sử dụng trong thực tế khi làm việc với Python. Trước khi vào bài mới thì bạn hãy quay lại bài cũ để xem [các kiểu dữ liệu trong Python](https://freetuts.net/cac-kieu-du-lieu-trong-python-3421.html) đã nhé, sau đó chúng ta sẽ tiếp tục với phần dưới đây.

À để kiểm tra kiểu dữ liệu của một biến thì ta dùng hàm type nhé các bạn, cú pháp như sau:

|  |  |
| --- | --- |
| 1 | type(variable) |

**Mục lục**

* [1. Ép kiểu dữ liệu ngầm trong Python](https://freetuts.net/ep-kieu-du-lieu-trong-python-1702.html#goto-h2-0)
* [2. Ép kiểu trong Python tường minh](https://freetuts.net/ep-kieu-du-lieu-trong-python-1702.html#goto-h2-1)
* [3. Bảng các hàm ép kiểu dữ liệu trong Python](https://freetuts.net/ep-kieu-du-lieu-trong-python-1702.html#goto-h2-2)
* [4. Lời kết](https://freetuts.net/ep-kieu-du-lieu-trong-python-1702.html#goto-h2-3)

### Ép kiểu dữ liệu ngầm trong Python

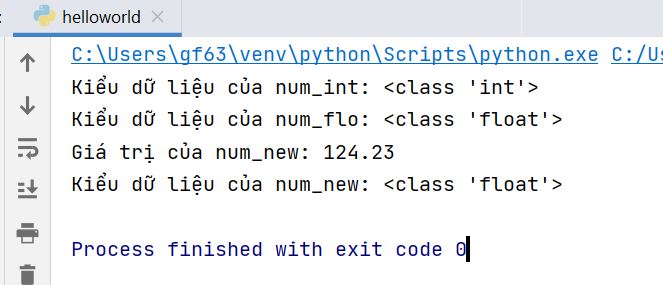
Đây là thao tác tự động chuyển đổi một loại dữ liệu sang loại dữ liệu khác của Python, quá trình này không cần bất kỳ sự tham gia của lập trình viên.

Chúng ta hãy xem ví dụ dưới đây, trong đó Python chuyển đổi kiểu dữ liệu thấp hơn (số nguyên) sang kiểu dữ liệu cao hơn (float) để tránh mất dữ liệu.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | num\_int = 123  num\_flo = 1.23    num\_new = num\_int + num\_flo    print("Kiểu dữ liệu của num\_int:",type(num\_int))  print("Kiểu dữ liệu của num\_flo:",type(num\_flo))    print("Giá trị của num\_new:",num\_new)  print("Kiểu dữ liệu của num\_new:",type(num\_new)) |

Kết quả của chương trình này như sau:

|  |  |
| --- | --- |
| 1  2  3  4 | Kiểu dữ liệu của num\_int: <class 'int'>  Kiểu dữ liệu của num\_flo: <class 'float'>  Giá trị của num\_new: 124.23  Kiểu dữ liệu của num\_new: <class 'float'> |



Trong chương trình trên thì:

* Mình đã định nghĩa hai biến num\_int và num\_flo, sau đó tạo một biến num\_new để lưu trữ tổng của hai biến đó.
* Tiếp theo sẽ dùng hàm type để kiểm tra kiểu dữ liệu của cả ba biến, thật bất ngờ vì num\_new đã mang kiểu **float**vì đây là kiểu số lớn hơn kiểu **int**. Như vậy biến num\_new đã được chuyển đổi ngầm.

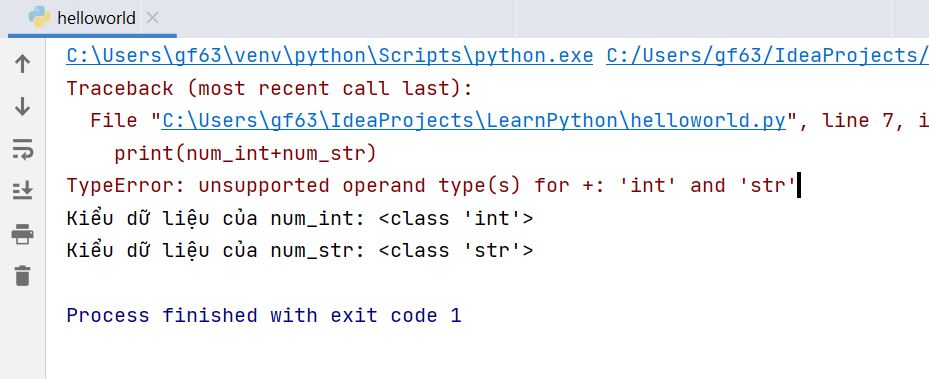
Bây giờ, hãy thử thêm một chuỗi và một số nguyên và xem Python xử lý thế nào.

**Ví dụ**: Bổ sung kiểu dữ liệu chuỗi (cao hơn) và kiểu dữ liệu số nguyên (thấp hơn)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | num\_int = 123  num\_str = "456"    print("Kiểu dữ liệu của num\_int:",type(num\_int))  print("Kiểu dữ liệu của num\_str:",type(num\_str))    # Dòng này sẽ lỗi vì string và number không chuyển ngầm được  print(num\_int+num\_str) |

Khi chạy chương trình trên, kết quả sẽ là:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | Traceback (most recent call last):    File "C:\Users\gf63\IdeaProjects\LearnPython\helloworld.py", line 7, in <module>      print(num\_int+num\_str)  TypeError: unsupported operand type(s) for +: 'int' and 'str'  Kiểu dữ liệu của num\_int: <class 'int'>  Kiểu dữ liệu của num\_str: <class 'str'> |



Như vậy mặc định Python không thể tự động chuyển đổi ngầm giữa string và number.

### Ép kiểu trong Python tường minh

Trong ép kiểu tường minh thì lập trình viên sẽ chuyển đổi loại dữ liệu của một đối tượng thành loại dữ liệu cần thiết theo nhu cầu của họ.

Chúng ta sử dụng các hàm được xác định trước như int(), float(), str(), v.v để thực hiện chuyển đổi loại rõ ràng.

Chuyển đổi loại này cũng được gọi là **typecasting** vì người dùng ép (thay đổi) kiểu dữ liệu của các đối tượng. Cú pháp như sau:

|  |  |
| --- | --- |
| 1 | (required\_datatype)(expression) |

Ví dụ dưới đây sẽ thể hiện việc ép kiểu từ **string**sang **int**trong Python, chương trình sẽ không báo lỗi.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | num\_int = 123  num\_str = "456"    print("Kiểu dữ liệu của num\_int:",type(num\_int))  print("Kiểu dữ liệu của num\_str trước khi ép kiểu:",type(num\_str))    num\_str = int(num\_str)  print("Kiểu dữ liệu của num\_str sau khi ép kiểu:",type(num\_str))    num\_sum = num\_int + num\_str    print("Tổng của num\_int và num\_str:",num\_sum)  print("Kiểu dữ liệu của sum:",type(num\_sum)) |

Chạy lên kế quả sẽ như sau:

|  |  |
| --- | --- |
| 1  2  3  4  5 | Kiểu dữ liệu của num\_int: <class 'int'>  Kiểu dữ liệu của num\_str trước khi ép kiểu: <class 'str'>  Kiểu dữ liệu của num\_str sau khi ép kiểu: <class 'int'>  Tổng của num\_int và num\_str: 579  Kiểu dữ liệu của sum: <class 'int'> |

Trong ví dụ trên thì bạn thấy mình đã sử dụng hàm int() để thực hiện chuyển đổi, ép kiểu một cách rõ ràng.

|  |  |
| --- | --- |
| 1 | num\_str = int(num\_str) |

### Bảng các hàm ép kiểu dữ liệu trong Python

Sau đây là bảng danh sách các hàm được sử dụng để kép kiểu trong Python, mời các bạn theo dõi.

|  |  |
| --- | --- |
| **STT** | **Function & Description** |
| 1 | int(x [,base]) ép kiểu int |
| 2 | long(x [,base] ) ép kiểu long int. |
| 3 | float(x) ép kiểu float. |
| 4 | complex(real [,imag]) ép kiểu complex number. |
| 5 | str(x) ép kiểu string. |
| 6 | repr(x) ép thành chuỗi biểu thức. |
| 7 | eval(str) ép chuỗi sang object. |
| 8 | tuple(s) ép kiểu tuple. |
| 9 | list(s) ép kiểu list. |
| 10 | set(s) ép kiểu set. |
| 11 | dict(d) ép kiểu dictionary. |
| 12 | frozenset(s) ép kiểu frozen set. |
| 13 | chr(x) ép kiểu in sang kiểu char |
| 14 | unichr(x) ép kiểu int sang Unicode character. |
| 15 | ord(x) ép ký tự sang kiểu int. |
| 16 | hex(x) ép kiểu integer sang chuỗi thập lục phân. |
| 17 | oct(x) ép kiểu integer chuỗi bát phân. |

### Lời kết

Như vậy là bạn đã tìm hiểu xong việc chuyển đổi, ép kiểu trong Python, đây là kiến thức căn bản nên bạn cần phải nắm vững để sau này có thể học nâng cao lên nhé. Sau đây là một vài lời tổng kết:

* Chuyển đổi kiểu dữ liệu là chuyển đổi đối tượng từ một kiểu dữ liệu này sang kiểu dữ liệu khác.
* Chuyển đổi kiểu ngầm định được **trình thông dịch Python** thực hiện tự động.
* Python tự động chọn kiểu dữ liệu cao hơn để tránh mất dữ liệu trong chuyển đổi loại ngầm định.
* Chuyển đổi loại rõ ràng cũng được gọi là **ép kiểu**, các loại dữ liệu của đối tượng được chuyển đổi bằng cách sử dụng các hàm của người dùng.
* Trong ép kiểu việc mất dữ liệu có thể xảy ra khi chúng ta thực thi đối tượng theo kiểu dữ liệu cụ thể.

## Các toán tử trong Python

Trong bài này mình sẽ giới thiệu đến các bạn danh sách các toán tử trong Python, đây là những toán tử được sử dụng rất thường xuyên khi lập trình Python.

Vấn đề muôn thuở, khi nói đến toán tử của các ngôn ngữ lập trình thì ta thường nhắc đến toán tử toán học, toán tử so sánh, toán tử gán, toán tử logic, toán tử quan hệ .. và trong bài này ta sẽ sẽ tìm hiểu đến chúng.

Thông qua bảng danh sách các toán tử trong Python này sẽ giúp bạn hiểu rõ hơn về cách sử dụng.

**Mục lục**

* [1. Toán tử toán học trong Python](https://freetuts.net/cac-toan-tu-trong-python-3423.html#goto-h2-0)
* [2. Toán tử so sánh trong Python](https://freetuts.net/cac-toan-tu-trong-python-3423.html#goto-h2-1)
* [3. Toán tử gán trong Python](https://freetuts.net/cac-toan-tu-trong-python-3423.html#goto-h2-2)
* [4. Toán tử logic trong Python](https://freetuts.net/cac-toan-tu-trong-python-3423.html#goto-h2-3)

### Toán tử toán học trong Python

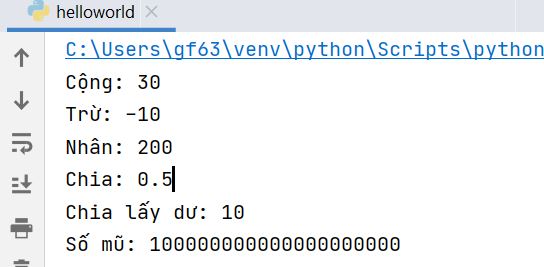
Toán tử toán học là những toán tử về cộng / trừ / nhân / chia ... trong toán học.

|  |  |
| --- | --- |
| **Toán tử** | **Ý nghĩa** |
| + | Cộng |
| - | Trừ |
| \* | Nhân |
| / | Chia |
| % | Chia lấy dư |
| \*\* | Tính lũy thừa |

**Ví dụ**: Hãy chạy ví dụ sau để hiểu rõ hơn về toán tử toán học.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | print("Cộng: " + str(a + b))  print("Trừ: " + str(a - b))  print("Nhân: " + str(a \* b))  print("Chia: " + str(a / b))  print("Chia lấy dư: " + str(a % b))  print("Số mũ: " + str(a \*\* b)) |

Kết quả như sau:



### Toán tử so sánh trong Python

**Lưu ý**: Trong Python thì giá trị True hoặc False phải ghi hoa chữ cái đầu tiên, nếu không Python sẽ hiểu nhầm đó là một biến.

Toán tử so sánh được dùng để so sánh mối quan hệ giữa hai toán hàng, kết quả nó trả về là một kiểu boolean (True hoặc False).

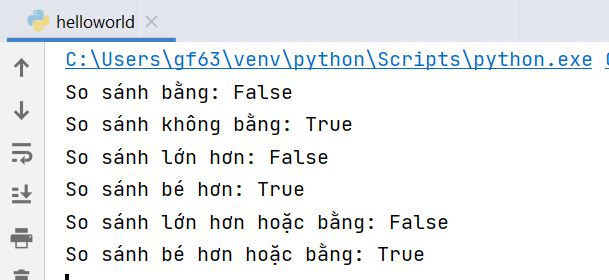
Giả sử ta có hai toán hạng A và B, lúc này các toán tử so sánh dưới đây sẽ có dạng A toan\_tu B.

|  |  |
| --- | --- |
| **Toán tử** | **Ý nghĩa** |
| == | Trả về True nếu A bằng B, False nếu a khác B |
| != | Trả về True nếu A khác B, False nếu A bằng B |
| <> | Giống với toán tử != |
| > | Trả về True nếu A > B, False nếu A bé hơn hoặc bằng B |
| < | Trả về True nếu A < B, False nếu A lớn hơn hoặc bằng B |
| >= | Trả vè True nếu A lớn hơn hoặc bằng B, ngược lại trả về False |
| <= | Trả về True nếu A bé hơn hoặc bằng B, ngược lại trả về False |

Xem ví dụ dưới đây.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | a = 10  b = 20    print("So sánh bằng: "              + str(a == b))  print("So sánh không bằng: "        + str(a != b))  print("So sánh lớn hơn: "           + str(a > b))  print("So sánh bé hơn: "            + str(a < b))  print("So sánh lớn hơn hoặc bằng: " + str(a >= b))  print("So sánh bé hơn hoặc bằng: "  + str(a <= b)) |

Chạy lên kết quả sẽ như sau:



### Toán tử gán trong Python

Chúng ta đã từng sử dụng toán tử gán ở những bài học trước rồi đấy các bạn, đó là toán tử =.

Toán tử gán được dùng để gán dữ liệu cho một biến nào đó.

Toán tử gán có thể được biến thể ra nhiều thể loại khác nhau.

|  |  |
| --- | --- |
| **Toán tử** | **Ý nghĩa** |
| c = a + b | Gán giá trị a + b vào biến c |
| a += b | Tương đương với a = a + b |
| a -= b | Tương đương với a = a - b |
| a \*= b | Tương đương với a = a \* b |
| a /= b | Tương đương với a = a / b |
| a %= b | Tương đương với a = a % b |
| a \*\*= b | Tương đương với a = a \*\* b |

|  |  |
| --- | --- |
| **Toán tử** | **Ý nghĩa** |
| c = a + b | Gán giá trị a + b vào biến c |
| a += b | Tương đương với a = a + b |
| a -= b | Tương đương với a = a - b |
| a \*= b | Tương đương với a = a \* b |
| a /= b | Tương đương với a = a / b |
| a %= b | Tương đương với a = a % b |
| a \*\*= b | Tương đương với a = a \*\* b |

Xét ví dụ dưới đây.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | a = 4  b = 2    c = a + b   # c = 6  a += b      # a = a + b = 6  a -= b      # a = a - b = -2  a \*= b      # a = a \* b = 8  a /= b      # a = a / b = 2  a %= b      # a = a % b = 0  a \*\*= b     # a = a \*\* b = 16 |

### Toán tử logic trong Python

Toán tử logic là những loại toán tử được dùng để kiểm tra sự liên kết giữa hai biểu thức, kết quả nó trả về là True hoặc False.

Chúng ta có ba toán tử thôn dụng gồm:

|  |  |
| --- | --- |
| Toán tử | Ý nghĩa |
| A and B | Trả về True nếu cả A và B có giá trị True, ngược lại trả về False |
| A or B | Trả về True nếu một trong hai A hoặc B có giá trị True, ngược lại trả về False |
| not(A) | Trả về True nếu A là False, và trả về False nếu A là True. Đây là toán tử phủ định |

**Ví dụ**: Bạn hãy thử chạy các lện sau để xem kết quả.

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | A = True  B = False    print (A and B) # False  print (A or B)  # True  print (not A)   # False |

Trên là tổng hợp 4 nhóm toán tử trong Python được sử dụng nhiều nhất. Thực tế vẫn còn rất nhiều loại toán tử khác, nhưng do chúng ít được sử dụng nên mình sẽ không liệt kê ra ở đây nhé.