



Lập trình Python cơ bản

Bài 2: Biến và các kiểu dữ liệu cơ sở



Phòng Lập Trình - Mạng

Nội dung

1. Định danh (Identifier)
2. Biến (Variable)
3. Các kiểu dữ liệu (Data type)
4. Chuyển đổi kiểu dữ liệu
5. Chú thích trong Python (comment)
6. Nhập/Xuất dữ liệu trên shell (Input/Output)

1. Định danh

- ❑ Trong lập trình, cần đặt tên cho biến (variable), phương thức/hàm (function), lớp (class), module và các đối tượng khác.
- ❑ Việc đặt tên được gọi là định danh (Identifier)
- ❑ Quy ước:
 - Tên biến phải **bắt đầu bằng chữ cái** (các ký tự A-Z, a-z) hoặc **ký tự _** (underscore). Tiếp đó là các **ký tự chữ, ký tự số** (0-9).
 - Định danh (Identifier) có phân biệt chữ HOA, chữ thường.
 - Không sử dụng các ký tự dấu câu (@, #, \$, %,...) để đặt tên.

1. Định danh

Ví dụ:

Tên đặt hợp lệ	Tên đặt không hợp lệ
x5	9x
_x	x#
Spam	%LoiNhuan
spam	Nam Nhuan
spAm	
total_of_eggs	
Total_Of_Eggs	

1. Định danh

❑ Một số quy tắc đặt tên (Identifier):

- **Tên class** bắt đầu bằng chữ hoa. Tất cả các identifier khác bắt đầu bằng chữ thường.
- **Tên function** viết thường, các từ nối với nhau bằng dấu _
- **Không** sử dụng các từ khóa (keyword) trong Python khi đặt tên cho bất cứ identifier nào.



1. Định danh

❑ Các từ khóa trong Python

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

Nội dung

1. Định danh (Identifier)
2. Biến (Variable)
3. Các kiểu dữ liệu (Data type)
4. Chuyển đổi kiểu dữ liệu
5. Chú thích trong Python (comment)
6. Nhập/Xuất dữ liệu trên shell (Input/Output)

2. Biến

□ Khái niệm

- Là một đơn vị lưu trữ trên bộ nhớ của máy tính, lưu trữ các giá trị có thể được dùng để tính toán xử lý.
- Biến có thể lưu trữ dữ liệu dạng chuỗi, dạng số,...
- Bằng cách gán các kiểu dữ liệu khác nhau cho biến, ta tạo ra các biến kiểu số nguyên, số thập phân, chuỗi...
- Cần phải khai báo biến khi sử dụng.
- Cú pháp:
`tên_var = <giá_trị>`
hoặc
`tên_var_1, tên_var_2, tên_var_3,... = <giá_trị_1>, <giá_trị_2>, <giá_trị_3>,...`

2. Biến

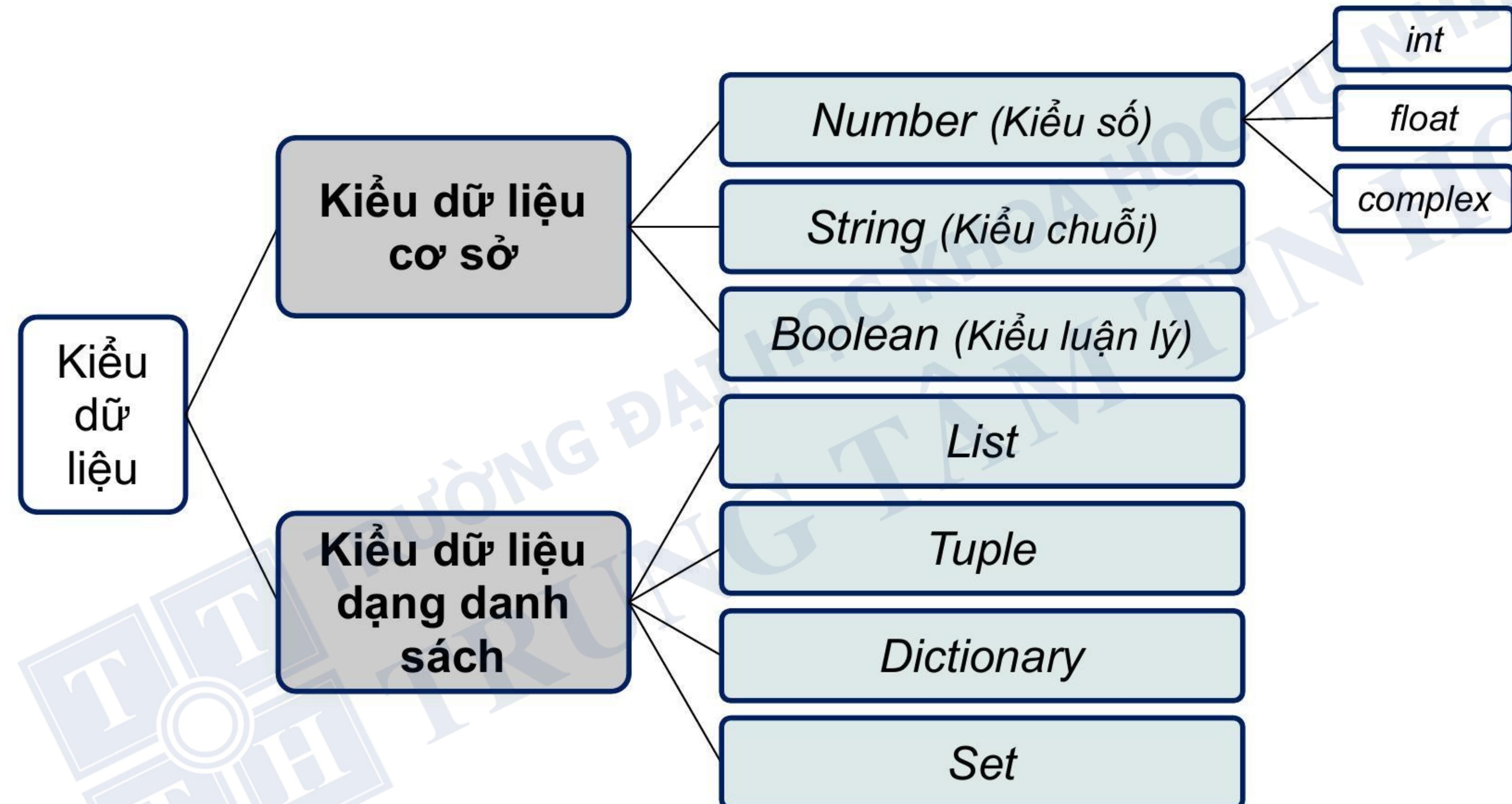
□ Ví dụ

```
ho_ten = 'Nguyễn Văn A'          # string  
tuoi = 18                         # int  
gioi_tinh = True                  # boolean  
toan, ly, hoa = 8.0, 7.5, 9.0    # float  
print('ĐTB: %.1f + %.1f + %.1f)/3 = %.1f' % (toan, ly, hoa, (toan+ly+hoa)/3))
```

Nội dung

1. Định danh (Identifier)
2. Biến (Variable)
3. Các kiểu dữ liệu (Data type)
4. Chuyển đổi kiểu dữ liệu
5. Chú thích trong Python (comment)
6. Nhập/Xuất dữ liệu trên shell (Input/Output)

3. Các kiểu dữ liệu



3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - Number (Kiểu số)

- ***int*** (số nguyên):

- Số lượng ký số của kiểu này là không giới hạn, chỉ bị phụ thuộc vào dung lượng của bộ nhớ.
- Python hỗ trợ biểu diễn số nguyên dưới 3 dạng cơ số

Cơ số	Tiền tố	Ví dụ	Ghi chú
Decimal		5, -89	Mặc định
Octal	0o	-0o490	
Hexadecimal	0x	0x5Cb7, -0X3A8f	Không phân biệt HOA, thường

3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - Number (Kiểu số)

- **float** (số thực):

- Có tối đa 15 số lẻ.
- Ví dụ: 0.0, -12.38, 7.52+e5, -98.71e100

- **complex** (số phức):

- Là 1 cặp số có thứ tự các số thực (real floating point) ký hiệu là **x + yj**, với x là **real** và y là **imag**.
- Ví dụ: 3+4j, 3.14j, 3e+26j, 9.123456e-17j

3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - Boolean (Kiểu luận lý)

- Chỉ có 2 giá trị: **True** hoặc **False**
- Ví dụ:

```
result = True  
print(result)                                # True  
  
not_result = not result  
print(not_result)                            # False
```

3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - String (Kiểu chuỗi)

- Là một chuỗi ký tự được đặt trong nháy kép (" ") hoặc nháy đơn ('')

- Khai báo và khởi tạo chuỗi:

```
tên_biến = <giá_trị>
```

- Ví dụ:

- name = "Sài gòn"

hoặc

- name = 'Sài gòn'

3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - String (Kiểu chuỗi)

- Có thể sử dụng 3 dấu nháy (nháy đơn hoặc nháy đôi) để khai báo chuỗi nhiều dòng.
- Ví dụ:

```
hello = """  
**      **  *****  **      **  *****  
**      **  **      **      **  **  **  
*****  *****  **      **  **  **  
**      **  **      **  **  **  **  
**      **  *****  *****  *****  *****  
***  ***
```

3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - String (Kiểu chuỗi)

- Truy xuất chuỗi: Sử dụng [index] hoặc [from:to]

- index (chỉ mục): **from: stop**

	Từ trái sang phải	Từ phải sang trái
Bắt đầu	0	-1
Kết thúc	chiều dài chuỗi - 1	- chiều dài chuỗi

- Ví dụ: greeting = 'Hello Python'

Length	1	2	3	4	5	6	7	8	9	10	11	12
Index	H	e	I	I	O	P	y	t	h	o	n	
	0	1	2	3	4	5	6	7	8	9	10	11
Negative Index	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - String (Kiểu chuỗi)

- Truy xuất chuỗi: Sử dụng [index] hoặc [from:to]

```
greeting = 'Hello Python'
```

- Truy xuất 1 ký tự trong chuỗi: `print(greeting[4])` => o
- Truy xuất chuỗi con:
 - Lấy theo khoảng index: `print(greeting[2:5])` => llo
 - Lấy 3 ký tự đầu tiên: `print(greeting[:3])` => Hel
 - Lấy 3 ký tự cuối cùng: `print(greeting[-3:])` => hon

3. Các kiểu dữ liệu

□ Kiểu dữ liệu cơ sở - String (Kiểu chuỗi)

- Lấy chiều dài chuỗi: Sử dụng hàm **len()**

```
greeting = 'Hello Python'
```

```
print(len(greeting)) => 12
```

- Nối chuỗi: sử dụng toán tử **+**

```
chuoi_1 = 'Trung tâm'
```

```
chuoi_2 = 'Tin học'
```

```
print(chuoi_1 + ' ' + chuoi_2) => Trung tâm Tin học
```

- Lặp chuỗi: sử dụng toán tử *****

```
print('-' * 20) => -----
```

Nội dung

1. Định danh (Identifier)
2. Biến (Variable)
3. Các kiểu dữ liệu (Data type)
4. Chuyển đổi kiểu dữ liệu
5. Chú thích trong Python (comment)
6. Nhập/Xuất dữ liệu trên shell (Input/Output)

4. Chuyển đổi kiểu dữ liệu

❑ Các phương thức chuyển đổi kiểu dữ liệu

Phương thức	Mô tả	Ví dụ
<code>int(x [,base])</code>	Chuyển đổi giá trị x thành số nguyên (integer) theo cơ số chỉ định (nếu có)	<pre>a = '10' b = 15 print(int(a) + b)</pre> <p>Kết quả: 25</p>
<code>float(x)</code>	Chuyển đổi giá trị x thành số thực (float)	<pre>a = '10.5' b = 15 print(float(a) + b)</pre> <p>Kết quả: 25.5</p>
<code>str(x)</code>	Chuyển đổi giá trị x thành chuỗi (string)	<pre>a = 10 b = 15 print(str(a) + str(b))</pre> <p>Kết quả: 1015</p>
<code>eval(str)</code>	Đánh giá kiểu dữ liệu của một chuỗi và trả về một object	<pre>c = '10 + 10' print(eval(c))</pre> <p>Kết quả: 20</p>

4. Chuyển đổi kiểu dữ liệu

❑ Các phương thức chuyển đổi kiểu dữ liệu

Phương thức	Mô tả	Ví dụ
<code>complex(real [,imag])</code>	Chuyển đổi một số thành complex number	<pre>real = 25 imag = 3.2 print(complex(real, imag))</pre> <p>Kết quả: (25+3.2j)</p>
<code>repr(x)</code>	Chuyển đổi tương x - thành 1 chuỗi expression string	<pre>lst1 = [1, 2, 3, 4, 5] lst2 = repr(lst1) print(lst2) print(type(lst2))</pre> <p>Kết quả:</p> <pre>[1, 2, 3, 4, 5] <class 'str'></pre>

4. Chuyển đổi kiểu dữ liệu

❑ Các phương thức chuyển đổi kiểu dữ liệu

Phương thức	Mô tả	Ví dụ
chr(x)	Chuyển một integer x thành ký tự tương ứng (mã ASCII)	print(chr (98)) Kết quả: 'b'
ord(x)	Chuyển ký tự x thành giá trị mã ASCII của ký tự	print(ord ('c')) Kết quả: 99
hex(x)	Chuyển một integer x thành chuỗi hexadecimal	print(hex (100)) Kết quả: 0x64
oct(x)	Chuyển một integer x thành chuỗi octal	print(oct (100)) Kết quả: 0o144

Nội dung

1. Định danh (Identifier)
2. Biến (Variable)
3. Các kiểu dữ liệu (Data type)
4. Chuyển đổi kiểu dữ liệu
5. Chú thích trong Python (comment)
6. Nhập/Xuất dữ liệu trên shell (Input/Output)

5. Chú thích trong Python

- ❑ **Chú thích (comment) là một hoặc nhiều dòng chứa một trong những nội dung sau:**
 - Ghi chú riêng
 - Phần chú thích có thể ghi thông tin tác giả, ngày viết, version
 - Che đoạn mã lệnh tạm thời không sử dụng
 - Giải thích cách xử lý của một đoạn chương trình, ...
- ❑ **Khi chạy chương trình, trình biên dịch sẽ không biên dịch phần chú thích này.**

5. Chú thích trong Python

□ **# (hash)**: Các ký tự đi sau dấu **#** trên cùng dòng sẽ được xem là chú thích.

- Ví dụ: `print(10 + 10) # 20`

□ **“nội dung ghi chú”** hoặc **““nội dung ghi chú””** cho phép ghi chú trên nhiều dòng

- Ví dụ: `"""`

LDS1

Python cơ bản

Fundamentals of Python

`"""`

Nội dung

1. Định danh (Identifier)
2. Biến (Variable)
3. Các kiểu dữ liệu (Data type)
4. Chuyển đổi kiểu dữ liệu
5. Chú thích trong Python (comment)
6. Nhập/Xuất dữ liệu trên shell (Input/Output)

6. Nhập/xuất dữ liệu trên shell

□ Ứng dụng trên shell (console):

- Là ứng dụng nhập xuất ở chế độ văn bản tương tự như màn hình Console của hệ điều hành MS-DOS.
- Các ứng dụng kiểu shell thường được dùng để minh họa các ví dụ cơ bản liên quan đến cú pháp ngôn ngữ, các thuật toán, và các chương trình ứng dụng không cần thiết đến giao diện người dùng đồ họa.

6. Nhập/xuất dữ liệu trên shell

❑ Xuất dữ liệu:

- Định dạng xuất sử dụng dấu phần trăm (percent sign - %)

Ký tự	Ý nghĩa
%c	Ký tự đơn
%d, %i	Số nguyên thập phân có dấu X
%e	Số chấm động (ký hiệu có số mũ – with lowercase letters)
%E	Số chấm động (ký hiệu có số mũ – with UPPERcase letters)
%f	Số chấm động (ký hiệu thập phân) X
%g	Định dạng ngắn gọn của %f và %e
%G	Định dạng ngắn gọn của %f và %E
%o	Số nguyên hệ bát phân (Octal integer)

Ký tự	Ý nghĩa
%p	Con trỏ (pointer)
%r	String (chuyển đổi bất kỳ đối tượng của Python bằng cách sử dụng hàm repr()). X
%s	String (chuyển đổi bất kỳ đối tượng của Python bằng cách sử dụng str()). X
%u	Số nguyên không dấu
%x	Số nguyên hệ thập lục (Hexadecimal – with lowercase letters)
%X	Số nguyên hệ thập lục (Hexadecimal – with UPPERcase letters)
%%	Xuất ra ký hiệu phần trăm (%)

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

- Định dạng xuất sử dụng dấu phần trăm (percent sign - %)

```
# Float
num = 1234.567
print('%f' % num)          # 1234.567000
print('%.0f' % num)         # 1235
print('%.2f' % num)         # 1234.57
print('%10.2f' % num)       # 1234.57

# String
str_num = '1234567890'
print('%s' % str_num)       # 1234567890
print('%.6s' % str_num)     # 123456
print('%.11s' % str_num)    # 1234567890
```

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

- Xuất chuỗi sử dụng dấu phẩy (,) hoặc dấu cộng (+) để nối chuỗi trong hàm `print()`.

Dấu phẩy (,)

```
print('Trung tâm', 'Tin học')
print('Năm:', 2022)
```

Trung tâm Tin học

Năm: 2022

Dấu cộng (+)

```
print('Trung tâm' + ' ' + 'Tin học')
print('Năm: ' + str(2022))
```

Trung tâm Tin học

Năm: 2022

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

- Xuất chuỗi sử dụng dấu ngoặc nhọn (`{}`) và phương thức `format`.

`"{①:>②}" .format(variables)`

Trong đó:

- Cho phép các cặp ngoặc nhọn được lồng nhau.
- ① nếu có cho biết thứ tự các đối số được cung cấp trong danh sách `variables`, thứ tự này được đánh số từ 0.
- ② có thể dùng 1 trong các dạng sau:
 - `{:>number}` : khoảng cách để in giá trị của biến là `number`
 - `{:.Xf}` : lấy X số lẻ của số thực cần in.
 - `{:W.Xf}` : in số thực trong độ rộng = W với X số lẻ.

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

- Xuất chuỗi sử dụng dấu ngoặc nhọn (`{}`) và phương thức `format`.

`"{①:>②}" .format(variables)`

Trong đó:

- ② có thể dùng 1 trong các dạng sau:
 - `{:+.xf}` : lấy X số lẻ của số thực cần in và có in dấu (âm hay dương) trước giá trị của biến.
 - `{:<xd|f}` : in cạnh trái. Ký hiệu `d|f` tùy thuộc kiểu dữ liệu là `int` hay `float`.
 - `{:xd|f}` : in cạnh phải. Ký hiệu `d|f` tùy thuộc kiểu dữ liệu là `int` hay `float`.

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

- Xuất chuỗi sử dụng dấu ngoặc nhọn (`{}`) và phương thức `format`.

`"{①:>②}" .format(variables)`

Trong đó:

- ② có thể dùng 1 trong các dạng sau:
 - `{ :^xd|f }` : in canh giữa. Ký hiệu `d|f` tùy thuộc kiểu dữ liệu là `int` hay `float`.
 - `{ :0>xd }` : in số nguyên sau khi chèn các số `0` bên trái sao cho đủ chiều dài `X`.
 - `{ :c<xd }` : in số nguyên sau đó in thêm các ký tự `C` bên phải của số nguyên cho đến khi đủ chiều dài `X`.

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

- Xuất chuỗi sử dụng dấu ngoặc nhọn (`{}`) và phương thức `format`.

`"{①:>②}" .format(variables)`

Trong đó:

- ② có thể dùng 1 trong các dạng sau:
 - `{ :, }` : in dấu phân cách phần ngàn. Chỉ chấp nhận dấu phẩy (,) làm dấu ngăn cách phần ngàn.
 - `{ :.X% }` : định dạng in số dưới dạng phần trăm (%) với X số lẻ.

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

Mã lệnh	Kết quả
Cho a, b = 5, -3.333	
print('{} + {} = {}'.format(a,b,a+b))	5 + -3.333 = 1.667
print('{x} + {y} = {z}'.format(x=a,y=b,z=a+b))	5 + -3.333 = 1.667
print('{} {:.2f}={:.0f}'.format(a,b,a+b))	5 -3.33 = 2
print('{} {:.2f}={:+.0f}'.format(a,b,a+b))	5 -3.33 = +2
print('{0:>5} + {1:>8}={2:>8}'.format(a,b,a+b))	5 + -3.333 = 1.666999999999998

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

Mã lệnh	Kết quả
n=2.345 print ('{:<10.2f}'.format(n))	2.34_____
print ('{:10.2f}'.format(n))	_____2.34____
print ('{:^10.2f}'.format(n))	_____2.34_____
print ('{:0>4d}'.format(a))	0005
print ('{@<4d}'.format(a))	5@0000
n=1234567.892 print ('{,.1f}'.format(n))	1,234,567.9

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

Mã lệnh	Kết quả
<pre>n=1234.567 print ('{10:,.2f}'.format(n))</pre>	1,234.57
<pre>n=12.345 print ('{:.2%}'.format(n))</pre>	1,234.50%
<pre>dientich = 1256.66 decimals = 2 print("Diện tích hình chữ nhật = {0:.{1}f}cm\u00b2".format(dientich, decimals))</pre>	Diện tích hình chữ nhật = 1256.66cm ²
<pre>thetich = 1254.725 decimals = 3 print("Thể tích hình trụ= {0:.{1}f}cm\u00b3".format(thetich, decimals))</pre>	Thể tích hình trụ= 1254.725cm ³

6. Nhập/xuất dữ liệu trên shell

□ Xuất dữ liệu:

- Các định dạng xuất dùng với Backslash sign (dấu \):
 - Chuỗi “\n” sử dụng trong hàm **print** để xuống dòng văn bản.
 - Sử dụng \ trước ký tự đặc biệt. Ví dụ: `print("What\'s your age?")`

6. Nhập/xuất dữ liệu trên shell

❑ Xuất dữ liệu:

- Các định dạng xuất dùng với Backslash sign (dấu \)

Định dạng	Ý nghĩa
\\	Backslash (\)
\'	Single quote (')
\"	Double quote (")
\a	ASCII Bell (BEL)
\b	ASCII Backspace (BS)
\f	ASCII Formfeed (FF)

Định dạng	Ý nghĩa
\n	ASCII Linefeed (LF)
\r	ASCII Carriage Return (CR)
\t	ASCII Horizontal Tab (TAB)
\v	ASCII Vertical Tab (VT)
\ooo	ASCII character with octal value ooo
\xhh...	ASCII character with hex value hh...

6. Nhập/xuất dữ liệu trên shell

□ Nhập dữ liệu:

- Hàm **input** cho phép đợi người dùng nhập dữ liệu và kết thúc việc nhập khi phím ENTER được nhấn. Hàm trả về chuỗi do người dùng nhập vào (kể cả khi toàn bộ dữ liệu người dùng nhập vào đều là ký số).
- Cú pháp: **input** (*prompt*)

Trong đó: *prompt* là chuỗi thông báo sẽ được in ra màn hình.

- Ví dụ:

```
name = input("What is your name: ")  
print("Your name is: " + name)
```



TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
TIT TRUNG TÂM TIN HỌC
TTH