Kiểu dữ liệu trong Python

Kiểu dữ liệu cơ bản (Built-in Data Types)

1. Kiểu số (Numeric Types):

```
# int: Số nguyên (ví dụ: 5, -10, 0)
# float: Số thực (ví dụ: 3.14, -2.5, 1.0)
# complex: Số phức (ví dụ: 3+4j, 2-1j)
```

int - Số nguyên

```
# Các cách khai báo số nguyên
age = 25            # Số nguyên dương
tuoi_nhanvien = 30  # Số nguyên dương
staff_age = 45    # Số nguyên dương
StaffAge = 50    # Số nguyên dương
temperature = -5    # Số nguyên âm
zero = 0            # Số không
full_age = int(input("Nhập tuổi:"))  # Sử dụng hàm int() để chuyển đổi
print("Tuổi của bạn là:", full_age)
# Hệ số khác nhau
binary = 0b1010      # Hệ nhị phân (10 trong thập phân)
octal = 0o12         # Hệ bát phân (10 trong thập phân)
hexadecimal = 0xa    # Hệ thập lục phân (10 trong thập phân)

# Số nguyên lớn (không giới hạn kích thước)
big_number = 12345678901234567890
```

float - Số thực

```
# Các cách khai báo số thực
pi = 3.14159
temperature = -15.5
zero_float = 0.0
height = float(input("Nhập chiều cao:"))  # Sử dụng hàm float() để chuyển đổi
print("Chiều cao của bạn là:", height)
weight = 70.0
chieu_cao = 1.75
ChieuCao = 1.80
CanNang = 65.5
can_nang = 80.2
# Ký hiệu khoa học
speed_of_light = 3e8      # 300,000,000
small_number = 1.5e-10    # 0.00000000015

# Giá trị đặc biệt
positive_infinity = float('inf')
negative_infinity = float('-inf')
not_a_number = float('nan')
```

complex - Số phức

```
# Các cách tạo số phức
z1 = 3 + 4j
z2 = complex(2, -1)       # 2 - 1j
z3 = complex('5+3j')      # Từ chuỗi

# Truy xuất phần thực và ảo
print(z1.real)            # 3.0
print(z1.imag)            # 4.0
print(abs(z1))            # 5.0 (độ lớn)
```

2. KIỂU VĂN BẢN (TEXT TYPE)

```
# str: Chuỗi ký tự (ví dụ: "Hello", 'Python', """Nhiều dòng""")
```

```
# Các cách tạo chuỗi
name = "Nguyễn Văn An"
ten_nhanvien = 'Trần Thị B'
StaffName = "Lê Văn Cường"
message = 'Hello World'
multiline = """Đây là
chuỗi nhiều dòng
với nội dung dài"""

print(multiline)
# Raw string (không xử lý escape characters)
path = r"C:\Users\name\Documents"

# F-string (formatted string)
age = 25
intro = f"Tôi tên {name}, {age} tuổi"

# Các phương thức chuỗi thường dùng
text = "  Python Programming  "
print(text.upper())           # "  PYTHON PROGRAMMING  "
print(text.lower())           # "  python programming  "
print(text.strip())           # "Python Programming"
print(text.replace("Python", "Java"))  # "  Java Programming  "
print(text.split())           # ['Python', 'Programming']
print(len(text))              # 21

# Slicing (cắt chuỗi)
word = "Python"
print(word[0])                # 'P'
print(word[1:4])              # 'yth'
print(word[::-1])             # 'nohtyP' (đảo ngược)
```

3. Kiểu logic (Boolean Type):

```
# bool: Giá trị logic True hoặc False
```

```
# Giá trị boolean
is_student = True
is_working = False

# Chuyển đổi sang boolean
print(bool(1))                # True
print(bool(0))                # False
print(bool(""))               # False (chuỗi rỗng)
print(bool("Hello"))          # True
print(bool([]))               # False (list rỗng)
print(bool([1, 2]))           # True

# Các phép toán logic
result1 = True and False      # False
result2 = True or False       # True
result3 = not True            # False
result4 = (5 > 3) and (2 < 4)   # True
result5 = (5 == 5) or (3 != 3)  # True
print(result1, result2, result3)
```

```
text = "học học lập trinh python rất dê"
index1 = text.find("python")  # Tìm vị trí của từ python
len_pyhton = len("python")  # Độ dài của từ python
index2 = index1 + len_pyhton  # Vị trí kết thúc của từ python
trich_xuat_chuoi = text[index1:index2]  # Trích xuất chuỗi từ vị trí index1 đến index2
print(trich_xuat_chuoi)
```

```
BaiTho ='''Kiếp con người mỏng manh như là gió
Sống trên đời có được mấy lần vui
Sao phải đau mà không thể mỉm cười
```

```
Gắng buông nỗi ngậm ngùi nơi quá khứ

Nếu có thể sao ta không làm thử
Để tâm hồn khắc hai chữ bình an
Cho đôi chân bước thanh thản nhẹ nhàng
Dù hướng đời có muôn ngàn đá sỏi
'''
# KetQua = BaiTho.find("con người")
# print(KetQua)
# KetQua = "con người" in BaiTho
# if KetQua == True:
#     print("Tìm thấy cụm từ 'Con người' trong bài thơ")
# else:
#     print("Không tìm thấy cụm từ 'Con người' trong bài thơ")
# TuCanTrichXuat = input("Nhập từ cần trích xuất:")
# index1 = BaiTho.find(TuCanTrichXuat)  # Tìm vị trí của từ con người
# len_tu = len(TuCanTrichXuat)  # Độ dài của từ con người
# index2 = index1 + len_tu  # Vị trí kết thúc của từ con người
# trich_xuat_chuoi = BaiTho[index1:index2]  # Trích xuất chuỗi từ vị trí index1 đến index2
# print(trich_xuat_chuoi.title())  # In chữ hoa chữ cái đầu mỗi từ

# BaiThoMoi = BaiTho.replace("bình an", "hạnh phúc")
# print(BaiThoMoi)
# BaiThoMoi = '"...' +"\n"+BaiTho + '..."'
# print(BaiThoMoi)  # In chuỗi nguyên bản
text = "bạn có khỏe không "
text1 = 0
text2 = text + str(text1)
print(text2)
```

```
bạn có khỏe không 0
```

4. Kiểu dữ liệu dạng chuỗi (Sequence Types)

```
# list: Danh sách có thể thay đổi (ví dụ: [1, 2, 3, "Python"])
# tuple: Danh sách không thể thay đổi (ví dụ: (1, 2, 3))
# range: Dãy số (ví dụ: range(0, 10))
```

```
# Tạo danh sách
numbers = [1, 2, 3, 4, 5]
mixed_list = [1, "hello", 3.14, True]
nested_list = [[1, 2], [3, 4], [5, 6]]

# Truy xuất phần tử
print(numbers[0])             # 1
print(numbers[-1])            # 5 (phần tử cuối)
print(numbers[1:4])           # [2, 3, 4]

# Thay đổi danh sách
numbers.append(6)             # Thêm vào cuối
numbers.insert(0, 0)          # Chèn vào vị trí 0
numbers.remove(3)             # Xóa giá trị 3
popped = numbers.pop()        # Lấy và xóa phần tử cuối

# Các phương thức khác
print(len(numbers))           # Độ dài
print(max(numbers))           # Giá trị lớn nhất
print(min(numbers))           # Giá trị nhỏ nhất
print(sum(numbers))           # Tổng các phần tử
numbers.sort()                # Sắp xếp
numbers.reverse()             # Đảo ngược

# List comprehension
squares = [x**2 for x in range(5)]  # [0, 1, 4, 9, 16]
evens = [x for x in range(10) if x % 2 == 0]  # [0, 2, 4, 6, 8]
```

```
# Tạo tuple
coordinates = (3, 5)
colors = ("red", "green", "blue")
single_item = (42,)           # Tuple một phần tử cần dấu phẩy
print(single_item)
# Truy xuất như list nhưng không thay đổi được
print(coordinates[0])         # 3
```

```
print(coordinates[1])        # 5

# Unpacking tuple
x, y = coordinates
print(x, y)                  # 3 5

# Tuple lồng nhau
nested = ((1, 2), (3, 4), (5, 6))
```

```
# Các cách tạo range
r1 = range(5)                # 0, 1, 2, 3, 4
r2 = range(2, 8)             # 2, 3, 4, 5, 6, 7
r3 = range(0, 10, 2)         # 0, 2, 4, 6, 8
r4 = range(10, 0, -1)        # 10, 9, 8, 7, 6, 5, 4, 3, 2, 1

# Chuyển đổi thành list
numbers = list(range(5))     # [0, 1, 2, 3, 4]

# Sử dụng trong vòng lặp
for i in range(3):
    print(f"Lần thứ {i}")
```

5. KIỂU DỮ LIỆU ÁNH XẠ (MAPPING TYPE)

```
# dict - Từ điển: Cặp khóa-giá trị (ví dụ: {"name": "Alice", "age": 30})
```

```
# Tạo dictionary
student = {
    "name": "Nguyễn Văn An",
    "age": 20,
    "major": "Computer Science",
    "gpa": 3.75
}

# Cách tạo khác
student2 = dict(name="Trần Thị Bình", age=19)
empty_dict = {}

# Truy xuất và thay đổi
print(student["name"])       # "Nguyễn Văn An"
print(student.get("age"))    # 20
print(student.get("phone", "Không có"))  # "Không có"

student["age"] = 21          # Thay đổi giá trị
student["phone"] = "0123456789"  # Thêm key mới

# Các phương thức dictionary
print(student.keys())        # dict_keys(['name', 'age', 'major', 'gpa', 'phone'])
print(student.values())      # dict_values([...])
print(student.items())       # dict_items([...])

# Xóa phần tử
del student["phone"]         # Xóa key
popped_value = student.pop("gpa", 0.0)  # Lấy và xóa

# Dictionary comprehension
squares_dict = {x: x**2 for x in range(5)}  # {0: 0, 1: 1, 2: 4, 3: 9, 4: 16}

# Nested dictionary
company = {
    "employees": {
        "engineering": ["An", "Bình", "Cường"],
        "marketing": ["Dung", "Em"]
    },
    "founded": 2020
}
```

6. KIỂU DỮ LIỆU TẬP HỢP (SET TYPES)

6.1 set - Tập hợp

```
# Tạo set
numbers = {1, 2, 3, 4, 5}
colors = {"red", "green", "blue"}
empty_set = set()             # Không thể dùng {} vì đó là dict rỗng

# Từ list sang set (loại bỏ trùng lặp)
duplicates = [1, 2, 2, 3, 3, 3, 4]
unique_numbers = set(duplicates)  # {1, 2, 3, 4}

# Thêm và xóa phần tử
numbers.add(6)                # Thêm phần tử
numbers.remove(3)             # Xóa phần tử (lỗi nếu không tồn tại)
numbers.discard(10)           # Xóa phần tử (không lỗi nếu không tồn tại)

# Các phép toán tập hợp
set1 = {1, 2, 3, 4}
set2 = {3, 4, 5, 6}

union = set1 | set2           # {1, 2, 3, 4, 5, 6} - Hợp
intersection = set1 & set2    # {3, 4} - Giao
difference = set1 - set2      # {1, 2} - Hiệu
symmetric_diff = set1 ^ set2  # {1, 2, 5, 6} - Hiệu đối xứng

# Kiểm tra quan hệ tập hợp
print({1, 2}.issubset({1, 2, 3}))      # True
print({1, 2, 3}.issuperset({1, 2}))    # True
print({1, 2}.isdisjoint({3, 4}))       # True (không giao nhau)
```

6.2 frozenset - Tập hợp bất biến

```
# Tạo bytes
data1 = b"Hello"              # Từ chuỗi literal
data2 = bytes([65, 66, 67])   # Từ danh sách số (ABC)
data3 = bytes(5)              # 5 bytes với giá trị 0
data4 = "Hello".encode('utf-8')  # Từ string

# Truy xuất
print(data1[0])               # 72 (mã ASCII của 'H')
print(data1[1:4])             # b'ell'

# Các phương thức
print(data1.decode('utf-8'))  # "Hello"
print(data1.upper())          # b'HELLO'
print(len(data1))             # 5
```

7. KIỂU DỮ LIỆU NHỊ PHÂN (BINARY TYPES)

CHUYỂN ĐỔI KIỂU DỮ LIỆU (TYPE CONVERSION)

```
# Chuyển đổi số
x = int("123")                # str -> int: 123
y = float("3.14")             # str -> float: 3.14
z = str(456)                  # int -> str: "456"

# Chuyển đổi collection
numbers_list = [1, 2, 3, 4, 5]
numbers_tuple = tuple(numbers_list)    # list -> tuple
numbers_set = set(numbers_list)        # list -> set
back_to_list = list(numbers_set)       # set -> list

# Chuyển đổi string
text = "Hello"
text_bytes = text.encode('utf-8')      # str -> bytes
back_to_str = text_bytes.decode('utf-8') # bytes -> str
```

KIỂM TRA KIỂU DỮ LIỆU

```
# Sử dụng type()
print(type(42))              # <class 'int'>
print(type("hello"))         # <class 'str'>
print(type([1, 2, 3]))       # <class 'list'>

# Sử dụng isinstance()
x = 42
print(isinstance(x, int))    # True
print(isinstance(x, str))    # False
print(isinstance(x, (int, float)))  # True (kiểm tra nhiều kiểu)

# Kiểm tra callable
def my_function():
    pass

print(callable(my_function))  # True
print(callable(42))           # False
```

Thao tác xử lý chuỗi trong python

1. TẠO CHUỖI

```
# Dấu nháy đơn và đôi
name = 'Nguyễn Văn An'
message = "Xin chào!"
quote = "Anh ấy nói: 'Tôi học Python'"
quote2 = 'Cô ấy nói: "Python rất hay"'

# Chuỗi nhiều dòng
poem = """Đây là bài thơ
Có nhiều dòng
Rất hay"""

paragraph = '''Đoạn văn này
cũng có nhiều dòng
và rất dài'''

# Raw string (không xử lý escape characters)
path = r"C:\Users\Name\Documents\file.txt"
regex = r"\d+\.\d+"

# Chuỗi Unicode
vietnamese = "Tiếng Việt có dấu"
emoji = "😜 😊 🎉"
```

```
BaiTho ='''Kiếp con người mỏng manh như là gió
Sống trên đời có được mấy lần vui
Sao phải đau mà không thể mỉm cười
Gắng buông nỗi ngậm ngùi nơi quá khứ

Nếu có thể sao ta không làm thử
Để tâm hồn khắc hai chữ bình an
Cho đôi chân bước thanh thản nhẹ nhàng
Dù hướng đời có muôn ngàn đá sỏi
'''
# tim_kim = "con người" in BaiTho
# print("từ con người có trong bài thơ:", tim_kim)  # True
# index1 = BaiTho.find("con người")  # Tìm vị trí từ "con người"
# index2 = index1 + len("con người")
# tu_tim_kim = BaiTho[index1:index2]
# print("Từ tìm kiếm:", tu_tim_kim)  # "con người"

tu_tim_kim = input("Nhập từ cần tìm:")
index1 = BaiTho.find(tu_tim_kim)  # Tìm vị trí từ
index2 = index1 + len(tu_tim_kim)
tu_tim_kim_kq = BaiTho[index1:index2]
print("Từ tìm kiếm:", tu_tim_kim_kq)
```

```
Từ tìm kiếm:
```

2. ĐỊNH DẠNG CHUỖI (STRING FORMATTING)

F-string (Python 3.6+) - Cách hiện đại nhất

```
name = "An"
age = 25
height = 1.75

# Cơ bản
greeting = f"Xin chào {name}, {age} tuổi"
info = f"Chiều cao: {height}m"

# Biểu thức trong f-string
calculation = f"2 + 3 = {2 + 3}"
upper_name = f"Tên viết hoa: {name.upper()}"

# Định dạng số
price = 1234567.89
formatted_price = f"Giá: {price:,.2f} VND"      # "Giá: 1,234,567.89 VND"
percentage = 0.75
percent_str = f"Tỉ lệ: {percentage:.1%}"        # "Tỉ lệ: 75.0%"

# Căn lề
text = f"|{name:<10}|{age:>5}|{height:^8.2f}|"  # Trái, phải, giữa
print(text)  # |An        |   25|  1.75  |
```

```
|An        |   25|  1.75  |
```

format() method

```
template = "Tên: {}, tuổi: {}, điểm: {:.1f}"
result = template.format("Bình", 22, 8.75)

# Với tên tham số
template2 = "Tên: {name}, tuổi: {age}, điểm: {score:.1f}"
result2 = template2.format(name="Cường", age=20, score=9.25)

# Với chỉ số
template3 = "Họ: {1}, tên: {0}"
result3 = template3.format("An", "Nguyễn")
```

% formatting (cách cũ)

```
name = "Dung"
score = 8.5
old_format = "Học sinh %s có điểm %.1f" % (name, score)
```

3. TRUY XUẤT VÀ CẮT CHUỖI (INDEXING & SLICING)

```
text = "Python Programming"

# Indexing (truy xuất từng ký tự)
first_char = text[0]          # 'P'
last_char = text[-1]          # 'g'
sixth_char = text[5]          # 'n'

# Slicing (cắt chuỗi)
first_6 = text[0:6]
print(first_6)           # 'Python'
first_6_short = text[:6]      # 'Python' (bỏ qua 0)
last_11 = text[7:]            # 'Programming' (từ vị trí 7 đến cuối)
middle = text[3:10]           # 'hon Pro'
```

```
# Slicing với bước
every_2nd = text[::2]         # 'Pto rgamn' (lấy mỗi ký tự thứ 2)
reverse = text[::-1]          # 'gnimmargorP nohtyP' (đảo ngược)
reverse_slice = text[15:5:-2] # 'gmrr h' (đảo ngược với bước 2)
```

Python

```
BaiTho ='''Kiếp con người mỏng manh như là gió
Sống trên đời có được mấy lần vui
Sao phải đau mà không thể mỉm cười
Gắng buông nỗi ngậm ngùi nơi quá khứ

Nếu có thể sao ta không làm thử
Để tâm hồn khắc hai chữ bình an
Cho đôi chân bước thanh thản nhẹ nhàng
Dù hướng đời có muôn ngàn đá sỏi
'''
# timkiem = BaiTho.find("con người") #5
# print(timkiem)
# timkiem1 = "con người" in BaiTho
# print(timkiem1)

# ketthuc = len("con người")
# print(ketthuc)
# BaiTho[timkiem:ketthuc + timkiem]
# print(BaiTho)
chuoi_tinkiem = input("Nhập chuỗi cần tìm:")
chieu_dai = len(chuoi_tinkiem)
index_dautien = BaiTho.find(chuoi_tinkiem)
index_thuhai =index_dautien + chieu_dai
print(BaiTho[index_dautien:index_thuhai])
```

4. CÁC PHƯƠNG THỨC KIỂM TRA CHUỖI

Kiểm tra nội dung

```
text = "Python123"
empty = ""
space = "   "

# Kiểm tra ký tự
print(text.isalnum())         # True (chữ và số)
print(text.isalpha())         # False (có số)
print(text.isdigit())         # False (có chữ)
print("123".isdigit())        # True
print(text.islower())         # False
print(text.isupper())         # False
print("python".islower())     # True

# Kiểm tra định dạng
print("Python".istitle())     # True (Title Case)
print(" ".isspace())          # True (chỉ có khoảng trắng)
print(empty.isspace())        # False (chuỗi rỗng)

# Kiểm tra nội dung đặc biệt
print("print".isidentifier()) # True (tên biến hợp lệ)
print("123abc".isidentifier()) # False
print("hello world".isprintable()) # True (in được)
```

Kiểm tra bắt đầu và kết thúc

```
filename = "document.pdf"
email = "user@example.com"

# startswith và endswith
print(filename.startswith("doc"))      # True
print(filename.endswith(".pdf"))       # True
print(filename.endswith((".pdf", ".doc"))) # True (nhiều lựa chọn)

print(email.startswith(("admin", "user"))) # True
```

5. TÌM KIẾM VÀ THAY THẾ

Tìm kiếm

```
text = "Python is great. Python is powerful."

# find() - trả về vị trí đầu tiên, -1 nếu không tìm thấy
pos1 = text.find("Python")           # 0
pos2 = text.find("Python", 1)       # 17 (tìm từ vị trí 1)
pos3 = text.find("Java")             # -1
print(pos1)
# rfind() - tìm từ cuối
last_pos = text.rfind("Python")      # 17

# index() - giống find() nhưng báo lỗi nếu không tìm thấy
try:
    pos = text.index("Java")
except ValueError:
    print("Không tìm thấy 'Java'")

# count() - đếm số lần xuất hiện
count = text.count("Python")         # 2
count_is = text.count("is")          # 2

# in operator - kiểm tra có tồn tại không
print("Python" in text)             # True
print("Java" in text)               # False
print("PYTHON" in text.upper())     # True
```

```
0
Không tìm thấy 'Java'
True
False
True
```

Thay thế

```
bai_tho='''
'''
```

```
text = "I love Java. Java is great."

# replace() - thay thế tất cả
new_text = text.replace("Java", "Python")
# "I love Python. Python is great."

# Giới hạn số lần thay thế
limited = text.replace("Java", "Python", 1)
# "I love Python. Java is great."

# Thay thế nhiều ký tự
messy = "a-b-c-d-e"
clean = messy.replace("-", " ")      # "a b c d e"

# Xóa ký tự (thay bằng chuỗi rỗng)
no_spaces = "a b c d e".replace(" ", "")  # "abcde"
```

6. THAO TÁC CASE (HOA/THƯỜNG)

```
text = "PyThOn PrOgRaMmInG"

# Chuyển đổi case
print(text.lower())          # "python programming"
print(text.upper())          # "PYTHON PROGRAMMING"
print(text.title())          # "Python Programming"
print(text.capitalize())     # "Python programming"
print(text.swapcase())       # "pYtHoN pROGRAmmIng"
```

```
# Các ví dụ thực tế
name = "nguyễn VĂN an"
proper_name = name.title()   # "Nguyễn Văn An"

email = "USER@EXAMPLE.COM"
clean_email = email.lower()  # "user@example.com"
```

7. LOẠI BỎ KHOẢNG TRẮNG

```
messy_text = "   Python Programming   "
left_space = "   Python"
right_space = "Python   "

# Loại bỏ khoảng trắng
print(messy_text.strip())     # "Python Programming"
print(left_space.lstrip())    # "Python"
print(right_space.rstrip())   # "Python"

# Loại bỏ ký tự tùy chỉnh
dotted = "...Python..."
print(dotted.strip("."))      # "Python"

mixed = "***Python###"
print(mixed.strip("*#"))      # "Python"

# Ví dụ thực tế
user_input = "  hello world  \n"
cleaned = user_input.strip()  # "hello world"
```

8. TÁCH VÀ GHÉP CHUỖI

Tách chuỗi (split)

```
sentence = "Python is amazing"
words = sentence.split()      # ['Python', 'is', 'amazing']

# Tách với delimiter tùy chỉnh
csv_data = "An,25,Hà Nội"
fields = csv_data.split(",")  # ['An', '25', 'Hà Nội']

# Giới hạn số lần tách
email = "user@sub.example.com"
parts = email.split(".", 1)   # ['user@sub', 'example.com']

# rsplit() - tách từ phải
path = "folder/subfolder/file.txt"
parts = path.rsplit("/", 1)   # ['folder/subfolder', 'file.txt']

# splitlines() - tách theo dòng
text = """Dòng 1
Dòng 2
Dòng 3"""
lines = text.splitlines()     # ['Dòng 1', 'Dòng 2', 'Dòng 3']

# partition() - tách thành 3 phần
email = "user@example.com"
parts = email.partition("@")  # ('user', '@', 'example.com')
```

Ghép chuỗi (join)

```
# join() - ghép danh sách thành chuỗi
words = ['Python', 'is', 'great']
sentence = " ".join(words)    # "Python is great"
sentence2 = "-".join(words)   # "Python-is-great"

# Ghép số (cần chuyển sang string)
numbers = [1, 2, 3, 4, 5]
number_str = ",".join(str(x) for x in numbers)  # "1,2,3,4,5"
# Hoặc
```

```
number_str2 = ",".join(map(str, numbers))       # "1,2,3,4,5"

# Ví dụ thực tế
path_parts = ['home', 'user', 'documents', 'file.txt']
full_path = "/".join(path_parts)  # "home/user/documents/file.txt"
```

```
text = "Học lập trình python rất thú vị"
text1 ="tôi cũng muốn học python"
# text[index1:index2]
index1 = text.find("pithon")
len_python = len("python")
index2 = index1 + len_python
trich_xuat_python = text[index1:index2]
print("Từ trích xuất:", trich_xuat_python)  # "python"
```

```
Từ trích xuất:
```

```
text = "Học lập trình python rất thú vị"
text1 ="tôi cũng muốn học python"
text2 =text +"\n"+text1
print(text2)
```

```
Học lập trình python rất thú vị
tôi cũng muốn học python
```

9. CĂNG CHỈNH VÀ ĐỆM CHUỖI

```
text = "Python"

# Căn giữa, trái, phải
print(text.center(20))        # "       Python       "
print(text.center(20, "*"))   # "*******Python*******"
print(text.ljust(10))         # "Python    "
print(text.rjust(10))         # "    Python"
print(text.rjust(10, "0"))    # "0000Python"

# zfill() - đệm số 0 phía trước
number = "42"
padded = number.zfill(5)      # "00042"

# Ví dụ thực tế - tạo bảng
items = ["Tên", "Tuổi", "Điểm"]
print("|".join(item.center(10) for item in items))
# "    Tên    |   Tuổi    |   Điểm    "
```

10. ENCODING VÀ DECODING

```
# Chuyển đổi string sang bytes
text = "Xin chào Việt Nam!"
utf8_bytes = text.encode('utf-8')
ascii_bytes = text.encode('ascii', errors='ignore')

# Chuyển đổi bytes sang string
decoded_text = utf8_bytes.decode('utf-8')

# Xử lý lỗi encoding
problematic = "Café"
safe_ascii = problematic.encode('ascii', errors='replace')  # b'Caf?'
ignore_ascii = problematic.encode('ascii', errors='ignore') # b'Caf'

print(utf8_bytes)             # b'Xin ch\xc3\xa0o Vi\xe1\xbb\x87t Nam!'
print(decoded_text)           # "Xin chào Việt Nam!"
```

11. THAO TÁC VỚI REGEX (CẦN IMPORT RE)

```
import re

text = "Số điện thoại: 0123-456-789 hoặc 0987.654.321"
```

```
# Tìm kiếm pattern
phone_pattern = r'\d{4}[-\.]\d{3}[-\.]\d{3}'
phones = re.findall(phone_pattern, text)
print(phones)
# ['0123-456-789', '0987.654.321']

# Thay thế với regex
cleaned = re.sub(r'[-\.]', '', text)
# "Số điện thoại: 0123456789 hoặc 0987654321"

# Tách với regex
parts = re.split(r'[-\.]', "2023-12-25")
# ['2023', '12', '25']
```

```
['0123-456-789', '0987.654.321']
```

13. VÍ DỤ THỰC TẾ

Xử lý dữ liệu người dùng

```
def clean_user_input(user_input):
    """Làm sạch input từ người dùng"""
    # Loại bỏ khoảng trắng thừa
    cleaned = user_input.strip()

    # Chuyển về chữ thường
    cleaned = cleaned.lower()

    # Loại bỏ ký tự đặc biệt
    cleaned = re.sub(r'[^a-zA-Z0-9\s]', '', cleaned)

    return cleaned

# Sử dụng
user_data = "  NgUyễn VĂn An!!!  "
clean_name = clean_user_input(user_data)
print(clean_name)  # "nguyễn văn an"
```

```
nguyn vn an
```

Xử lý email

```
def validate_email(email):
    """Kiểm tra email hợp lệ"""
    email = email.strip().lower()

    # Kiểm tra format cơ bản
    if '@' not in email or email.count('@') != 1:
        return False

    username, domain = email.split('@')

    # Kiểm tra username và domain
    if not username or not domain:
        return False

    if not domain.count('.') >= 1:
        return False

    return True

# Test
emails = ["test@example.com", "invalid.email", "user@domain.co.uk"]
for email in emails:
    print(f"{email}: {validate_email(email)}")
```

```
test@example.com: True
invalid.email: False
user@domain.co.uk: True
```

Format tiền tệ

```
def format_currency(amount):
    """Format số tiền theo kiểu Việt Nam"""
    # Chuyển thành chuỗi với 2 chữ số thập phân
    formatted = f"{amount:,.2f}"

    # Thay dấu phẩy và chấm (nếu cần)
    # formatted = formatted.replace(',', 'X').replace('.', ',').replace('X', '.')

    return formatted + " VND"

# Sử dụng
prices = [1000, 1500.5, 1234567.89]
for price in prices:
    print(format_currency(price))
# 1,000.00 VND
# 1,500.50 VND
# 1,234,567.89 VND
```

```
1,000.00 VND
1,500.50 VND
1,234,567.89 VND
```

1. CẤU TRÚC CỦA LỆNH IF

Cú pháp cơ bản

```
# if điều_kiện:
#     # Khối lệnh thực hiện khi điều kiện đúng
#     lệnh_1
#     lệnh_2

# Lưu ý quan trọng:

# Dấu hai chấm : sau điều kiện là bắt buộc
# Khối lệnh phải được thụt lề (indentation) - thường là 4 spaces hoặc 1 tab
# Python dựa vào thụt lề để xác định khối lệnh
```

```
age =19
if age >= 18:
    print("Bạn đã đủ tuổi để bỏ phiếu.")
    print("Hãy tham gia bầu cử để góp phần xây dựng đất nước.")
    print("bạn có thể lái xe.")
```

```
so1 = float(input("Nhập số thứ nhất:"))
so2 = float(input("Nhập số thứ hai:"))
toantu = input("Nhập phép toán (+, -, *, /):")
if toantu == "+":
    ketqua = so1 + so2
    print(f"Kết quả: {so1} + {so2} = {ketqua}")
if toantu == "-":
    ketqua = so1 - so2
    print(f"Kết quả: {so1} - {so2} = {ketqua}")
if toantu == "*":
    ketqua = so1 * so2
    print(f"Kết quả: {so1} * {so2} = {ketqua}")
if toantu == "/" and so2 != 0:
    ketqua = so1 / so2
    print(f"Kết quả: {so1} / {so2} = {ketqua}")
```

```
so1 = float(input("Nhập số thứ nhất:"))
so2 = float(input("Nhập số thứ hai:"))
toantu = input("Nhập phép toán (+, -, *, /):")
if toantu == "+":
    ketqua = so1 + so2
    print(f"Kết quả: {so1} + {so2} = {ketqua}")
elif toantu == "-":
    ketqua = so1 - so2
    print(f"Kết quả: {so1} - {so2} = {ketqua}")
```

```
elif toantu == "*":
    ketqua = so1 * so2
    print(f"Kết quả: {so1} * {so2} = {ketqua}")
elif toantu == "/":
    if so2!=0:
        ketqua = so1 / so2
        print(f"Kết quả: {so1} / {so2} = {ketqua}")
    else:
        print("Lỗi: Không thể chia cho 0.")
else:
    print("Phép toán không hợp lệ.")
```

```
Phép toán không hợp lệ.
```

2. CÁC DẠNG CẤU TRÚC IF

Lệnh if đơn giản

```
age = 18

if age >= 18:
    print("Bạn đã đủ tuổi trưởng thành")
    print("Có thể làm căn cước công dân")

# Ví dụ khác
score = 85
if score >= 80:
    print("Bạn đạt điểm giỏi!")
```

```
age = int(input("Nhập tuổi của bạn: "))
if age >= 18:
    print("Bạn đã đủ tuổi trưởng thành")
    print("Bạn có thể tham gia bầu cử")
    print("Được uống rượu bia")
else: # chính là age <18
    print("Bạn chưa đủ tuổi trưởng thành")
    print("Chưa được tham gia bầu cử")
    print("Không được uống rượu bia")
```

```
age = 18
if age >= 18:
    pass
```

```
number1 = float(input("Nhập  số 1: "))
number2 = float(input("Nhập số 2: "))
operator = input("Nhập phép toán (+, -, *, /): ")
if operator == "+":
    result = number1 + number2
    print(f"Kết quả: {number1} + {number2} = {result}")
if operator == "-":
    result = number1 - number2
    print(f"Kết quả: {number1} - {number2} = {result}")
if operator == "*":
    result = number1 * number2
    print(f"Kết quả: {number1} * {number2} = {result}")
if operator == "/" and number2 != 0:
    result = number1 / number2
    print(f"Kết quả: {number1} / {number2} = {result}")
```

```
so1 = float(input("Nhập số 1: "))
so2 = float(input("Nhập số 2: "))
toantu = input("Nhập phép toán (+, -, *, /): ")
if toantu == "+":
    ketqua = so1 + so2
    print(f"Kết quả: {so1} + {so2} = {ketqua}")
if toantu == "-":
    ketqua = so1 - so2
    print(f"Kết quả: {so1} - {so2} = {ketqua}")
if toantu == "*":
```

```
    ketqua = so1 * so2
    print(f"Kết quả: {so1} * {so2} = {ketqua}")
if toantu == "/" and so2 != 0:
    ketqua = so1 / so2
    print(f"Kết quả: {so1} / {so2} = {ketqua}")
```

Lệnh if-else

```
age = 16

if age >= 18:
    print("Bạn đã đủ tuổi trưởng thành")
else:
    print("Bạn chưa đủ tuổi trưởng thành")

# Ví dụ với số
number = 7
if number % 2 == 0:
    print(f"{number} là số chẵn")
else:
    print(f"{number} là số lẻ")
```

```
import math
# tínhdelta = b**2 - 4*a*c
a = float(input("Nhập a: "))
b = float(input("Nhập b: "))
c = float(input("Nhập c: "))
delta = b**2 - 4*a*c
if delta >=0:
    canbachai = math.sqrt(delta)
    print("Căn bậc hai của delta là:", canbachai)
else:
    print("Delta âm, không có căn bậc hai thực")
```

```
# Giả phuong trình bật nhất
b = float(input("Nhập a: "))
c = float(input("Nhập b: "))

# bx + c = 0
if b == 0:
    if c ==0: # 0x + 0 =0
        print("Phuong trình bật nhất có vsn")
    else: # 0x + b =0 -> sai
        print("Phương trình bậc nhất vô nghiệm")
else: # a !=0
    x= -c/b
    print("Phương trình có nghiệm x =", x)
```

```
# giải phuong trình bac 2:
# ax^2 + bx + c =0
import math
a = float(input("nhập a:"))
b = float(input("Nhập b: "))
c = float(input("Nhập c: "))

if a == 0: # bx +c =0
    if b == 0:
        if c ==0: # 0x + 0 =0
            print("Phuong trình bật nhất có vsn")
        else: # 0x + b =0 -> sai
            print("Phương trình bậc nhất vô nghiệm")
    else: # a !=0
        x= -c/b
        print("Phương trình có nghiệm x =", x)
else: # a !=0 ax^2 + bx + c =0
    tinh_delta = b*b - 4*a*c
    if tinh_delta <0:
        print("Vo Nghiệm")
    elif tinh_delta == 0:
        x1 = x2 = -b/(2*a)
        print(f"nghiệm kép x1 = x2 = {x1} ")
    else: # tinh_delta >0
```

```
        can_bac_hai = math.sqrt(tinh_delta)
        x1 = (-b + can_bac_hai)/(2*a)
        x2 =(-b - can_bac_hai)/(2*a)
        print(f"nghiệm x1 ={x1}, x2 = {x2}")
```

```
nghiệm x1 =-0.3333333333333333, x2 = -1.0
```

Lệnh if-elif-else

```
score = 75

if score >= 90:
    grade = "Xuất sắc"
elif score >= 80:
    grade = "Giỏi"
elif score >= 70:
    grade = "Khá"
elif score >= 50:
    grade = "Trung bình"
else:
    grade = "Yếu"

print(f"Điểm {score} - Xếp loại: {grade}")

# Ví dụ với nhiều điều kiện
temperature = 25

if temperature < 0:
    print("Trời rất lạnh, có thể có băng tuyết")
elif temperature < 10:
    print("Trời lạnh")
elif temperature < 20:
    print("Trời mát")
elif temperature < 30:
    print("Trời ấm")
elif temperature < 40:
    print("Trời nóng")
else:
    print("Trời rất nóng")
```

```
so1 = float(input("Nhập số 1: "))
so2 = float(input("Nhập số 2: "))
toantu = input("Nhập phép toán (+, -, *, /): ")
if toantu == "+":
    ketqua =   so1 + so2
    print(f" tong 2 so {so1} va {so2} = {ketqua}")
elif toantu =='-':
    ketqua = so1 - so2
    print(f"trừ 2 số {so1} - {so2} = {ketqua}")
elif toantu == "*":
    ketqua = so1*so2
    print("Nhân 2 số {so1}*{so2} = {ketqua}")
elif toantu == '/':
    if so2 !=0:
        ketqua = so1/so2
        print("chia  số {so1} / {so2} ={ ketqua}")
    else:
        print("phép chia không hợp lệ")
else:
    print("khong thưc hiện phép chia nào ca")
```

3. CÁC TOÁN TỬ SO SÁNH

```
a = 10
b = 5

# Toán tử so sánh cơ bản
print(a == b)    # False - bằng
print(a != b)    # True - khác
```

```
print(a > b)     # True - lớn hơn
print(a < b)     # False - nhỏ hơn
print(a >= b)    # True - lớn hơn hoặc bằng
print(a <= b)    # False - nhỏ hơn hoặc bằng

# Ví dụ trong if
if a > b:
    print("a lớn hơn b")

if a == 10:
    print("a bằng 10")

# So sánh chuỗi
name1 = "An"
name2 = "Bình"

if name1 == name2:
    print("Tên giống nhau")
else:
    print("Tên khác nhau")

# So sánh với None
value = None
if value is None:
    print("Giá trị là None")

if value is not None:
    print("Giá trị không phải None")
```

4. TOÁN TỬ LOGIC

Toán tử and

```
age = 20
has_license = True

# Cả hai điều kiện phải đúng
if age >= 18 and has_license:
    print("Có thể lái xe")
else:
    print("Không thể lái xe")

# Ví dụ với số
x = 15
if x > 10 and x < 20:
    print("x nằm trong khoảng 10-20")

# Có thể viết ngắn gọn hơn
if 10 < x < 20:
    print("x nằm trong khoảng 10-20")
```

Toán tử or

```
day = "Thứ 7"

# Một trong hai điều kiện đúng là đủ
if day == "Thứ 7" or day == "Chủ nhật":
    print("Cuối tuần")
else:
    print("Ngày trong tuần")

# Ví dụ khác
weather = "mưa"
if weather == "mưa" or weather == "bão":
    print("Nên ở nhà")
else:
    print("Có thể ra ngoài")
```

Toán tử not

```
is_raining = False

if not is_raining:
    print("Không mưa, có thể ra ngoài")
else:
    print("Đang mưa, nên ở nhà")

# Ví dụ với list
my_list = []
if not my_list:  # List rỗng được coi là False
    print("List rỗng")

# Ví dụ với string
name = ""
if not name:  # Chuỗi rỗng được coi là False
    print("Tên trống")
```

Kết hợp các toán tử logic

```
age = 25
income = 50000
has_job = True

# Kết hợp and và or
if (age >= 18 and age <= 65) and (income > 30000 or has_job):
    print("Đủ điều kiện vay ngân hàng")
else:
    print("Không đủ điều kiện")

# Sử dụng not với and/or
if not (age < 18 or age > 65):
    print("Trong độ tuổi lao động")
```

5. KIỂM TRA MEMBERSHIP (in, not in)

```
# Kiểm tra trong list
fruits = ["táo", "cam", "chuối", "xoài"]
fruit = "cam"

if fruit in fruits:
    print(f"{fruit} có trong danh sách")
else:
    print(f"{fruit} không có trong danh sách")

# Kiểm tra trong string
text = "Học Python rất thú vị"
if "Python" in text:
    print("Có chứa từ Python")

# Kiểm tra không có trong
forbidden_words = ["spam", "hack", "virus"]
message = "Xin chào mọi người"

if not any(word in message.lower() for word in forbidden_words):
    print("Tin nhắn sạch")
else:
    print("Tin nhắn chứa từ cấm")

# Cách đơn giản hơn
has_forbidden = False
for word in forbidden_words:
    if word in message.lower():
        has_forbidden = True
        break

if not has_forbidden:
    print("Tin nhắn sạch")
```

6. LỆNH IF LỒNG NHAU (NESTED IF)

```
age = 20
has_license = True
has_car = False

if age >= 18:
    print("Đủ tuổi lái xe")
    if has_license:
        print("Có bằng lái")
        if has_car:
            print("Có thể lái xe ngay")
        else:
            print("Cần mượn hoặc thuê xe")
    else:
        print("Cần thi bằng lái")
else:
    print("Chưa đủ tuổi lái xe")

# Ví dụ phân loại học sinh
math_score = 85
english_score = 75

if math_score >= 80:
    if english_score >= 80:
        print("Học sinh xuất sắc cả hai môn")
    elif english_score >= 70:
        print("Giỏi Toán, khá Tiếng Anh")
    else:
        print("Chỉ giỏi Toán")
else:
    if english_score >= 80:
        print("Chỉ giỏi Tiếng Anh")
    else:
        print("Cần cố gắng thêm")
```

7. CONDITIONAL EXPRESSION (TERNARY OPERATOR)

```
# Cú pháp: giá_trị_nếu_đúng if điều_kiện else giá_trị_nếu_sai

# age = 20
# if age >=18:
#     status = "Trưởng thành"
# else:
#     status = "Chưa trưởng thành"
# print(status)

# status = "Trường thành" if age >=18 else " chua trưởng thành"

# age =int(input("Nhập tuổi của bạn: "))
# if age >=18:
#     print("Bạn đã đủ tuổi trưởng thành")
#     print("Bạn có thể tham gia bầu cử")
#     print("Được uống rượu bia")
# else: # chính là age <18
#     print("Bạn chưa đủ tuổi trưởng thành")
#     print("Chưa được tham gia bầu cử")
#     print("Không được uống rượu bia")
so1 = float(input("Nhập số 1: "))
so2 = float(input("Nhập số 2: "))
# if so1 > so2:
#     max_value = so1
# else:
#     max_value = so2
# print(f"Giá trị lớn hơn là: {max_value}")
max_value = so1 if so1 > so2 else so2
print(f"Giá trị lớn hơn là: {max_value}")

# status = "Trưởng thành" if age >= 18 else "Chưa trưởng thành"
# print(status)

# # Ví dụ với số
# x = 5
# y = 10
# max_value = x if x > y else y
# print(f"Giá trị lớn hơn: {max_value}")
```

```
# # Ví dụ trong function
# def get_grade(score):
#     return "Đậu" if score >= 50 else "Rớt"

# print(get_grade(75))  # "Đậu"
# print(get_grade(30))  # "Rớt"

# # Lồng nhau (không nên lạm dụng)
# score = 85
# grade = "Xuất sắc" if score >= 90 else "Giỏi" if score >= 80 else "Khá" if score >= 70 else "Trung bình"
```

```
Giá trị lớn hơn là: 6.0
```

```
# tìm số lớn nhát trong 2 số
so1 = float(input("Nhập số 1: "))
so2 = float(input("Nhập số 2: "))
# if so1 > so2:
#     print(f"Số lớn nhất là: {so1}")
# else:
#     print(f"Số lớn nhất là: {so2}")
max_2so = so1 if so1 >so2 else so2
print(f"Số lớn nhất là: {max_2so}")
```