TRƯỜNG ĐẠI HỌC
VĂN LANG
Đạo đức - Ý chí - Sáng tạo
KHOA CÔNG NGHỆ THÔNG TIN

KỸ THUẬT LẬP TRÌNH

Chương : TUPLE (DÃY)

GVGD: Ths. Nguyễn Minh Tân

HỌC KỲ I – NĂM HỌC 2020-2021

KHÓA 24T-IT

NỘI DUNG

1. Giới thiệu Tuple

- Kiểu dữ liệu Tuple trong Python được gọi là dãy số. Kiểu Tuple cũng được định nghĩa khá giống với List (danh sách).
- Điều khác duy nhất là: Tuple ko thể thay đổi các phần tử trong nó còn List thì ngược lại.

2. Khai báo và khởi tạo Tuple

- Cú pháp: <biến nhớ> = (<giá trị 1>,<giá trị 2>,…<giá trị N>)
- Cách tạo Tuple: Đặt các phần tử trong cặp ngoặc tròn () và cách nhau bởi dấu phẩy
- Ngoặc tròn có thể được bỏ qua, nhưng mới bắt đầu thì nên tuân thủ đúng điều trên
- empty_tuple = () # Tuple rỗng
- print(empty_tuple)
- my_tuple = (-1, 2, 0, 3) # Tuple chứa các số nguyên
- print(my_tuple)
- your_tuple = (1, 'Hi', 'Night!', 2.718, 1+3j) # Tuple chứa phần tử mang nhiều kiểu dữ liệu
- print(your_tuple)
- our_tuple = (1992, [1, 0, 3, -5], (), ('Py', 'js', 0, 3.14)) # Tuple dạng lồng nhau
- print(our_tuple)
- Tup(50,)

2. So sánh Tuple và List

- List và Tuple đều là một dãy (sequence) các phần tử. Chúng có các khác biệt sau:
- Khi viết một List bạn sử dụng cặp dấu ngặc vuông [ ], trong khi viết một Tuple bạn sử dụng dấu ngặc tròn ( ).
- Ví dụ:
- # Đây là một Tuple.
- aTuple = ("apple", "apricot", "banana")
-
- # Đây là một List (Danh sách).
- aList = ["apple", "apricot", "banana"]

2. So sánh Tuple và List

- List là kiểu dữ liệu có thể biến đổi (mutable), bạn có thể sử dụng phương thức như **append()** để thêm phần tử vào List, hoặc sử dụng phương thức **remove()** để xóa các phần tử ra khỏi List mà không làm tạo ra thêm một thực thể 'List' khác trên bộ nhớ.

2. So sánh Tuple và List

- list1 = [1990, 1991, 1992]
-
- print ("list1: ", list1)
- # Địa chỉ của list1 trên bộ nhớ.
- list1Address = hex (id(list1))
- print ("Address of list1: ", list1Address)
- print ("\n")
- print ("Append element 2001 to list1")
- # Nối (append) một phần tử vào list1.
- list1.append(2001)
- print ("list1 (After append): ", list1)
- # Địa chỉ của list1 trên bộ nhớ.
- list1Address = hex (id(list1))
- print ("Address of list1 (After append): ", list1Address)

2. So sánh Tuple và List

- **Tuple** là một đối tượng bất biến (immutable), nó không có các phương thức **append()**, **remove()**,.. như list. Một số phương thức, hoặc toán tử mà bạn nghĩ rằng nó dùng để cập nhập **Tuple**, nhưng không phải vậy, nó dựa trên Tuple ban đầu để tạo ra một Tuple mới.

2. So sánh Tuple và List

- tuple1 = (1990, 1991, 1992)
-
- # Địa chỉ của tuple1 trên bộ nhớ.
- tuple1Address = hex (id(tuple1))
- print ("Address of tuple1: ", tuple1Address)
-
- # Nối một tuple vào tuple1.
- tuple1 = tuple1 + (2001, 2002)
-
- # Địa chỉ của tuple1 trên bộ nhớ.
- tuple1Address = hex (id(tuple1))
- print ("Address of tuple1 (After concat): ", tuple1Address)
- .

2. Khai báo và khởi tạo Tuple

- # Tạo Tuple không cần dùng cặp ngoặc tròn => Kỹ thuật Tuple packing
- mine_tuple = 'cat', 1999, 'pig', 1996
- print(mine_tuple)
- # => unpacking
- a, b, c, d = mine_tuple
- print(f"a = {a}, b = {b}, c = {c}, d = {d}")

2. Khai báo và khởi tạo Tuple

- # Việc tạo một Tuple có 1 phần tử hơi khó khăn chút và dễ bị nhầm lẫn
- one_tuple = ('one element') # có gợi ý từ IDE, và đang hiểu đây là String ko phải Tuple
- print(one_tuple)
- a_tuple = ('one element',) # như này mới hiểu là Tuple 1 phần tử, chú ý dấu phẩy
- print(a_tuple)
- tuple_01 = 'one element',   # đây cũng được hiểu là Tuple 1 phần tử với việc không dùng ngoặc tròn
- print(tuple_01)

4. Truy cập đến các phần tử trong Tuple

- """ Truy cập vào trong Tuple
- - Bằng chỉ số dương: Dùng toán tử [] và đưa chỉ số phần tử muốn truy cập. Chỉ số bắt đầu từ 0 đến (số phần tử - 1).
- Với Tuple lồng nhau thì dùng chỉ số lồng nhau.
- - Bằng chỉ số âm: Dùng toán tử [] và đưa chỉ số dạng âm của phần tử muốn truy cập.
- Chỉ số âm đánh từ phải sang trái, bắt đầu là -1 cho phần tử cuối cùng, và ngược đến -(số phần tử) cho phần tử đầu tiên
- - Đoạn cắt: Dùng toán tử [] kết hợp với toán tử : để có thể lấy ra 1 đoạn các phần tử của tuple
- """

4. Truy cập đến các phần tử trong Tuple

```
your_tuple = (1, 'Hi', 'Night!', 2.718, 1+3j)
print(len(your_tuple))  # Số lượng phần tử của your_tuple
print(your_tuple[4])
print(your_tuple[0])

our_tuple = (1992, [1, 0, 3, -5], (), ('Py', 'js', 0, 3.14))
print(our_tuple[1])
print(our_tuple[1][2])
print(our_tuple[3][1])

my_tuple = ('py', 'thon', 'java', 'script', 'Ox', 'Oy')
print(my_tuple[-2])
print(my_tuple[-6])

my_tuple = ('py', 'thon', 'java', 'script', 'Ox', 'Oy')
print(my_tuple[:4])
print(my_tuple[2:])
print(my_tuple[1:4])
print(my_tuple[:-5])
```

5. Thao tác với Tuple

- """ Thay đổi, cập nhật Tuple. Không giống như List, Tuple là kiểu dữ liệu không cho phép thay đổi (giống String).
- - Điều này có nghĩa là các phần tử không thể bị thay đổi.
- Nhưng nếu phần tử trong một Tuple lồng mà là dạng dữ liệu có thể thay đổi thì ta có thể thực hiện thay đổi được (<= Khá dị)
- - Có thể thực hiện gán giá trị mới cho biến Tuple cũ (reassignment)
- - Ghép 2 Tuple bằng cách dùng dấu +
- - Nhân 1 Tuple với 1 số nguyên dương n => lặp lại Tuple đó lên n lần
- """

Ghép 2 tuple

- tup1 = (12, 34.56)
- tup2 = ('abc', 'xyz')
-
  # Following action is not valid for tuples
- # tup1[0] = 100;
-
  # So let's create a new tuple as follows
- tup3 = tup1 + tup2
- print(tup3)

5. Thao tác với Tuple

- my_tuple = (1992, [1, 0, 3, -5], (26, 10, 1999))
- # my_tuple[2] = 2000 # Lỗi: TypeError: 'tuple' object does not support item assignment
- my_tuple[1][3] = 5
- print(my_tuple)

- my_tuple = ('re', 'assignment', 'reassignment')
- print(my_tuple)

- tuple_01 = (1, 2, 3)
- tuple_02 = (4, 5, 6)
- tuple_03 = (7, 8, 9)
- print(tuple_01 + tuple_02 + tuple_03)
- print(tuple_01 * 2)

5. Thao tác với Tuple

- """ Xóa Tuple
- - Như trên đã nói, chúng ta ko thể thay đổi các phần tử trong Tuple => không thể xóa phần tử trong Tuple
- - Tuy nhiên, có thể xóa hoàn toàn 1 tuple bằng từ khóa del
- """
- my_tuple = (0, 1, 2, 3, 4, 5)
- # del my_tuple[3]
- del my_tuple
- # print(my_tuple)

6. Các phương thức của Tuple

- """ - Do không phải kiểu dữ liệu có thể thay đổi => add và remove không thể tồn tại trong kiểu dữ liệu Tuple
- - Chỉ còn 1 phương thức
- - tuple.count(element): Đếm số lượng phần tử element trong tuple
- - tuple.index(element): Trả lại chỉ số của phần tử element đầu tiên từ trái sang phải
- """

```
my_tuple = ('d', 'o', 'g', 'p', 'i', 'g', 'd', 'u', 'c', 'k')
print(my_tuple.count('g'))
print(my_tuple.index('d'))
```

6. Các phương thức của Tuple

Hàm	Mô tả
~~cmp(list1, list2)~~	So sánh các phần tử của cả 2 Tuple. Hàm này đã bị loại bỏ khỏi **Python3**.
len(list)	Trả về số phần tử của danh sách
max(list)	Trả về phần tử trong Tuple với giá trị lớn nhất.
min(list)	Trả về phần tử trong Tuple với giá trị nhỏ nhất.
tuple(seq)	Chuyển đổi một List thành một Tuple.

```
tuple1 = (1991, 1994, 1992)

tuple2 = (1991, 1994, 2000, 1992)

print ("tuple1: ", tuple1)
print ("tuple2: ", tuple2)

# Trả về số lượng phần tử trong danh sách.
print ("len(tuple1): ", len(tuple1) )
print ("len(tuple2): ", len(tuple2) )

# Giá trị lớn nhất trong Tuple.
maxValue = max(tuple1)

print ("Max value of tuple1: ", maxValue)

# Giá trị nhỏ nhất trong Tuple.
minValue = min(tuple1)

print ("Min value of tuple1: ", minValue)

# (all)
# List
list3 = [2001, 2005, 2012]

print ("list3: ", list3)

# Chuyển đổi một List thành tuple.
tuple3 = tuple (list3)

print ("tuple3: ", tuple3)
```

7. Các toán tử khác của Tuple

- Cũng giống như String, Tuple có 3 toán tử **+**, ***** , **in**.

Toán tử	Mô tả	Ví dụ
+	Toán tử dùng để nối (concatenate) 2 Tuple để tạo ra một Tuple mới	(1, 2, 3) + ("One","Two") --> (1, 2, 3, "One", "Two")
*	Toán tử dùng để nối nhiều bản sao chép của cùng một Tuple. Và tạo ra một Tuple mới	(1, 2) * 3 --> (1, 2, 1, 2, 1, 2)
in	Kiểm tra một phần tử nằm trong một Tuple hay không, kết quả trả về True hoặc False.	"Abc" in ("One", "Abc") --> True

7. Các toán tử khác của Tuple

- """ - Kiểm tra có thuộc Tuple hay không: in hoặc not in
- - Duyệt một Tuple với for và in
- """

```
my_tuple = ('d', 'o', 'g', 'p', 'i', 'g')
print('a' in my_tuple)
print('i' in my_tuple)
print('h' not in my_tuple)

love_tuple = ("HAT", 'MT', 'HNH', 'TH')
for singer in love_tuple:
    print(f"I like {singer}")

for i in range(len(love_tuple)):
    print(f"I like", love_tuple[i])
```

8. Ưu điểm của Tuple

Một số ƯU ĐIỂM của Tuple so với List.

- Theo các thứ đã trao đổi, Tuple khá giống với List và chúng thường được sử dụng trong các tình huống tương tự nhau.
- Tuy nhiên, vẫn có một số ưu điểm khi sử dụng Tuple so với List như sau:
- - Thường dùng Tuple chứa các phần tử với các kiểu dữ liệu không đồng nhất (khác nhau)
- Còn List chứa các phần tử với kiểu dữ liệu đồng nhất (giống nhau)
- - Tuple là kiểu dữ liệu immutable (bất biến/ko thay đổi được), thì duyệt nó sẽ nhanh hơn List => Tăng cường hiệu năng
- - Tuple có chứa các phần tử immutable, có thể dùng làm key cho dictionary, với List thì không thể
- - Nếu có dữ liệu không được thay đổi, dùng Tuple sẽ đảm bảo được tính chất đó, nó sẽ chống ghi đè

8. Comparing tuples

- a=(5,6)
- b=(1,4)
- if (a>b):print("a is bigger")
- else: print("b is bigger")

8. Comparing tuples

- #case 2
- a=(5,6)
- b=(5,4)
- if (a>b):print("a is bigger")
- else: print ("b is bigger")

8. Comparing tuples

- a=(5,6)
- b=(6,4)
- if (a>b):print("a is bigger")
- else: print("b is bigger")

8. Case

- Case1: Comparison starts with a first element of each tuple. In this case 5>1, so the output a is bigger

- Case 2: Comparison starts with a first element of each tuple. In this case 5>5 which is inconclusive. So it proceeds to the next element. 6>4, so the output a is bigger

- Case 3: Comparison starts with a first element of each tuple. In this case 5>6 which is false. So it goes into the else block and prints "b is bigger."

8. Củng cố bài học

- Tìm các cách khởi tạo List hợp lệ dưới đấy
- A. tup = tuple((1,2, 3) + [3, 4])
- B. tup = (1)
- C. tup = 1
- D. tup = 1, 2

8. Củng cố bài học

- Dự đoán kết quả của chương đoạn code dưới đây
- >>> tup = (1, 2, [3, 4])
- >>> tup[2] += [50, 60]

8. Củng cố bài học

- Lựa chọn phương án đúng
- A. tup = (1, 2, [3, 4, 50, 60])
- B. TypeError: ‘tuple’ object does not support item assignment
- C. a và b đúng

9. Bài tập thực hành

Bài 00: Viết chương trình sinh một tuple chứa các phần tử có các kiểu dữ liệu khác nhau.
Sau đó, unpack các phần tử trong một tuple.

Bài 01: Viết chương trình chuyển một tuple sang thành list và ngược lại từ list sang tuple

Bài 02: Viết chương trình đảo ngược một tuple.

Bài 03: Viết chương trình đếm số lượng các phần tử trong một list đến khi gặp một phần tử kiểu tuple.

Bài 04: Cho 1 list chứa các tuple không rỗng.
Viết chương trình sắp xếp list đó theo chiều tăng dần của phần tử cuối trong mỗi tuple.
Ví dụ: [(2, 5), (4, 1), (0, 0)] => [(0, 0), (4, 1), (2, 5)]

Bài 05: Viết chương trình tìm ra tuple có phần tử thứ 2 là nhỏ nhất từ một list chứa các tuple.

Bài 06: Viết chương trình in ra phần tử thứ 4 và phần tử thứ 4 từ cuối lên trong một tuple.

Bài 07: Viết chương trình kiểm tra 2 tuple có phần tử chung hay không.

Bài 08: Viết chương trình kiểm tra xem tất cả các phần tử trong tuple có giống nhau hay không.

Bài 09: Viết chương trình tính tổng và tìm giá trị lớn nhất trong tuple chứa các số thực.

Bài 10: Cho list sau: ["www.hust.edu.vn", "www.wikipedia.org", "www.asp.net", "www.amazon.com"]
Viết chương trình để in ra hậu tố (vn, org, net, com) trong các tên miền website trong list trên.

Bài 11: Viết chương trình tìm từ dài nhất trong một câu nhập vào từ bàn phím.

LỊCH HỌC BÙ THỨ 7: CA1,CA2 14/11/2020

- **LÝ DO**: KHOA CNTT TỔ CHỨC LỄ NHẬP MÔN
- **LỊCH BÙ THỨ 5:** CA2: TIẾT 4,5,6 NGÀY 12/11
- **LỊCH BÙ THỨ 6:** CA1: TIẾT 1,2,3 NGÀY 11/11
- LỊCH TẠM THỜI LÀ NHƯ VẬY CHỨ THẦY CŨNG CHƯA CÓ CHẮC. VÌ THẦY PHẢI MAIL LÊN XIN NỮA XEM HỌ CÓ ĐỒNG Ý HAY KHÔNG. NẾU ĐƯỢC THÌ THẦY SẼ MAIL CHO CÁC EM PHÒNG HỌC CỤ THỂ.