Python Starter

Послідовності

План заняття

1. Що таке послідовність?
2. Операції над послідовностями
3. Перевірка входження елемента
4. Порівняння послідовностей
5. Кортежі (tuple)
6. Іменовані кортежі (namedtuple)

Після уроку обов'язково

Повторіть цей урок у відео форматі на ITVDN.com

Доступ можна отримати через керівництво вашого навчального центру

Перевірте, як Ви засвоїли цей матеріал на TestProvider.com

Тема

Послідовності

Поняття послідовності

- Послідовністю в Python називається ітерабельний об'єкт, який підтримує ефективний доступ до елементів з використанням цілочисельних індексів через спеціальний метод __getitem__() і підтримує метод __len__(), який повертає довжину послідовності. До основних вбудованих типів послідовностей відносяться list (розглянемо на наступному занятті), tuple, range, str та bytes.
- Послідовності також опціонально можуть реалізовувати методи count(), index(), __contains__(), __reversed__() та інші

Загальні операції

Операція	Опис
x in s, x not in s	чи знаходиться елемент x у послідовності s
s + t	конкатенація послідовностей
s * n, n * s	конкатенація n неповних копій послідовності s
s[i]	i-й елемент послідовності s
s[i:j], s[i:j:k]	зріз послідовності s від i до j із кроком k
len(s)	довжина послідовності
min(s)	мінімальний елемент послідовності
max(s)	максимальний елемент послідовності
s.index(x[, i[, j]])	індекс першого входження x (опціонально – починаючи з позиції i до позиції j)
s.count(x)	кількість елементів x у послідовності s
sum(s)	sum(s) – сума елементів послідовності

Перевірка входження елемента

```
my_digit = 14
# TypeError: argument of type 'int' is not iterable
# print(4 in my_digit)

my_string = 'test'
print('e' in my_string)

my_tuple = 10, 12, 14
print(12 in my_tuple)

my_list = [10, 12, 14]
print(12 in my_list)
```

Порівняння послідовностей

- Дві послідовності рівні, якщо вони мають однаковий тип, рівну довжину та відповідні елементи обох послідовностей рівні.
- Послідовності однакових типів можна порівнювати. Порівняння відбуваються в лексикографічному порядку: послідовність меншої довжини менша, ніж послідовність більшої довжини; якщо ж їх довжини рівні, то результат порівняння дорівнює результату порівняння перших елементів, що відрізняються.

Кортежі

- **Кортежі** – це незмінні послідовності, які зазвичай використовуються, щоб зберігати різнотипні дані. Представлені класом tuple.
- Підтримують усі загальні для послідовностей операції.

```
Створення кортежів:
my_tuple = ()
my_tuple = (1,)
my_tuple = 1,
my_tuple = (1, 2, 'a string')
my_tuple = 1, 2, 'a string'
my_tuple = tuple(range(8))
```

Індексування кортежів

```
x = (10, 20, 30)
x[0]  # 10
x[1]  # 20
x[2]  # 30
x[3]  # IndexError: tuple index out of range
```

Індексування з негативними числами почнеться з останнього елемента як -1:

```
x[-1] # 30
x[-2] # 20
x[-3] # 10
x[-4] # IndexError: tuple index out of range
```

Індексування ряду елементів:

```
print(x[:-1])   # (10, 20)
print(x[-1:])   # (30,)
print(x[1:3])   # (20, 30)
```

Функції із довільною кількістю аргументів. Розпакування аргументів функції

- Функція може мати довільну кількість аргументів. Після всіх позиційних параметрів функції або замість них (але перед тими, які передбачається використовувати як іменовані) у її сигнатурі можна вказати спеціальний аргумент із символом * перед ім'ям. Тоді фактичні параметри, що залишилися, зберігаються в кортежі з цим ім'ям.
- Також є і зворотна можливість. Якщо під час виклику функції перед ім'ям ітерабельного об'єкта поставити символ *, його елементи розпаковуються в позиційні аргументи.

Розпакування кортежів

```
a, b, c = 1, 2, 3
# змінюємо місцями значення двох змінних:
a, b = b, a
a, b, c = iterable
head, *tail = iterable
for index, value in enumerate(sequence):
        pass
```

Іменовані кортежі

Іменовані кортежі (Namedtuple) - це заміна звичайних кортежів, вони виконують ті ж функції, але покращують читаність коду. Навіть якщо програміст призупинить роботу над програмою на кілька місяців, синтаксис або значення кортежу будуть говорити самі за себе. Іменовані кортежі можуть бути відмінною альтернативою визначення класів і вони мають деякі інші цікаві особливості.

Іменований кортеж можна створити за допомогою будь-якої структури даних, що підтримує ітерацію. Мета іменованих кортежів – вирішити дві проблеми:

- іменовані кортежі є незмінними подібно до звичайних кортежів. Не можна змінювати їх після того, як ви щось помістили у них. Інакше можна отримати виняток: AttributeError: can't set attribute
- кожен об'єкт, збережений в іменованих кортежах, може бути доступний через унікальний, зручний для читання людиною ідентифікатор. Це звільняє вас від запам'ятовування цілочисельних індексів або вигадування обхідних шляхів типу визначення цілочисельних констант як мнемонік для ваших індексів.

Синтаксис namedtuple в Python:

```
import collections
collections.namedtuple(typename, field_names, *, rename=False, defaults=None, module=None)
Або:
from collections import namedtuple
# Іменований кортеж
Person = namedtuple('Person', 'name surname’)
named_tuple = Person(‘Петро’, ‘Петренко')
print(named_tuple)
```

Відмінність іменованого кортежу (Namedtuple) від словника

Namedtuple - це заміна звичайних кортежів. Вони виконують ті ж функції, але покращують читаність коду. Навіть якщо програміст призупинить роботу над програмою на кілька місяців, синтаксис або значення самого кортежу будуть говорити самі за себе. Тепер час поговорити про синтаксис таких кортежів.

Іменований кортеж – не частина стандартної бібліотеки, він доступний у модулі **collections**.

Незважаючи на схожість **namedtuple** і словника, між ними є відмінності:

- до словника можна додавати нові значення, а в іменований кортеж - ні!
- словники вважаються більш інформативними та наочними;
- для створення словника потрібно менше рядків коду;
- до значення словника отримуємо доступ через ключ, а в іменованому кортежі через точкову нотацію чи індекс.

Іменовані кортежі покращують читаність коду та окремих функцій. З кодом зручно працювати, якщо пам'ятати призначення кортежу. Проте, швидше за все, після його створення все забудеться вже за кілька днів.

Фабрика іменованих кортежів

Фабрика іменованих кортежів – оголошення іменованого кортежу, яке може використовуватися для створення нескінченної кількості аналогічних структур.

```
Marks = namedtuple('Marks', 'Physics Chemistry Math English’)

# Створення іменованого кортежу
marks = Marks(90, 85, 95, 100)
print(marks)
```

Доступ до імен полів namedtuple в Python

Іменовані кортежі (namedtuple) – це підклас кортежів у Python. У них ті ж функції, що й у звичайних, але їх значення можна отримувати як за допомогою імені (через точку, наприклад, .name), так і за допомогою індексу (наприклад, [0]). У цьому матеріалі на прикладах розберемо синтаксис та функції цих кортежів.

Для цього є декілька способів:

- доступ до полів можна отримати тим же способом, що і в кортежах – за допомогою індексу.
- через точку.
- getattr().

Доступ до полів можна отримати тим самим способом, що й у кортежах – за допомогою індексу.

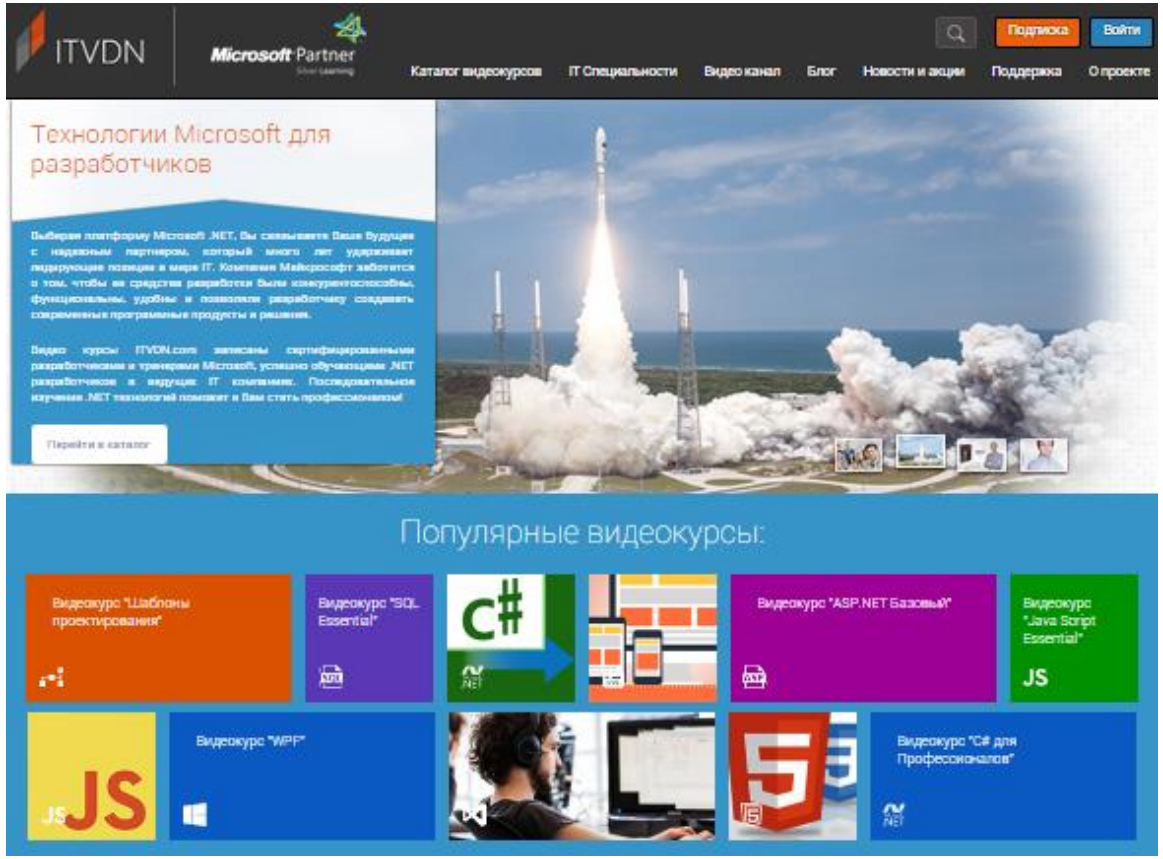
Зміна значень в іменованих кортежах

Функція _replace() приймає ім'я поля та значення, яке потрібно замінити, а повертає іменований кортеж зі зміненим значенням. Таким чином, цей тип виявляється змінюваним за допомогою одного простого трюку.

```
marks = marks._replace(Physics=99)
```

Дивіться наші уроки у відео форматі

ITVDN.com



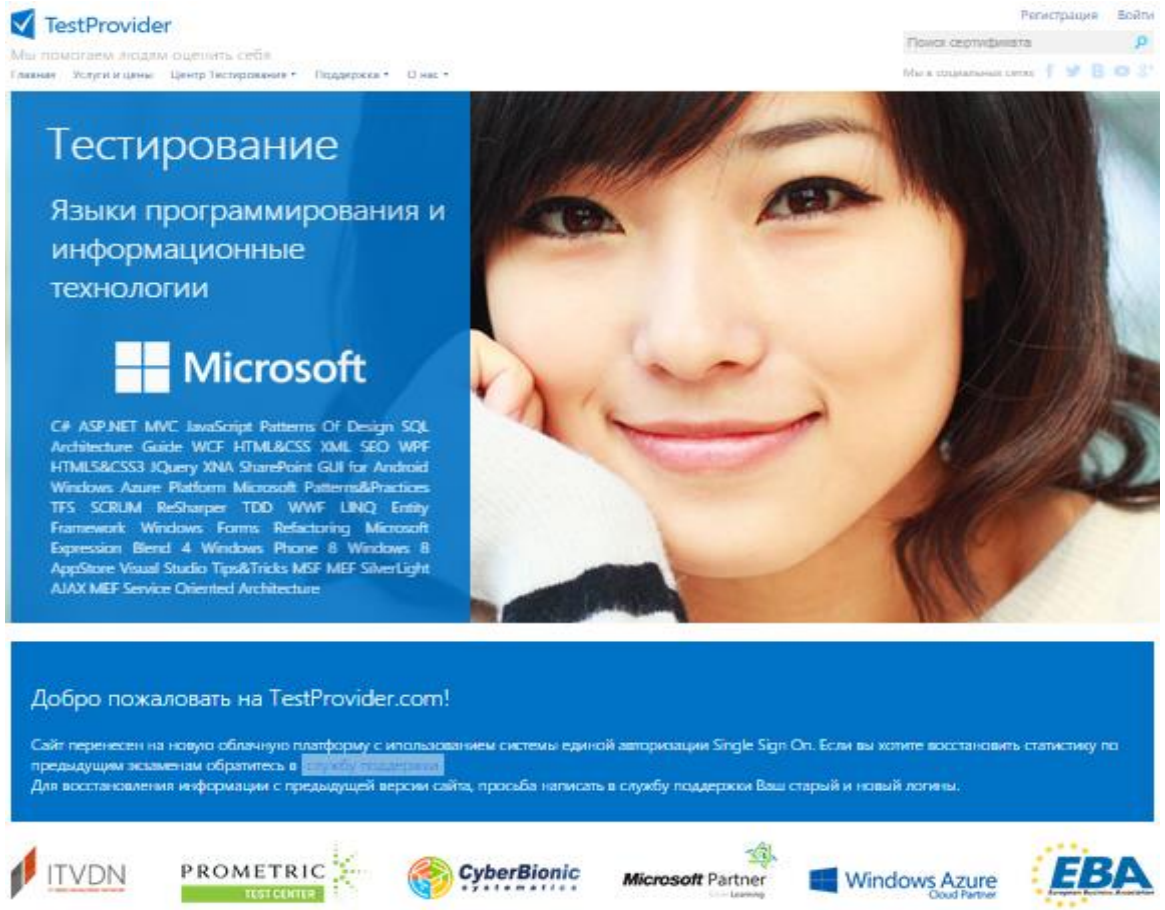
Перегляньте цей урок у відео форматі на освітньому порталі ITVDN.com
для закріплення пройденого матеріалу.

Усі курси записані сертифікованими тренерами, які працюють у навчальному центрі CyberBionic Systematics

ITVDN
IT VIDEO DEVELOPERS NETWORK

Перевірка знань

TestProvider.com



TestProvider – це online сервіс перевірки знань з інформаційних технологій. За його допомогою Ви можете оцінити Ваш рівень та виявити слабкі місця. Він буде корисним як у процесі вивчення технології, так і для загальної оцінки знань IT спеціаліста.

Після кожного уроку проходьте тестування для перевірки знань на TestProvider.com

Успішне проходження фінального тестування дозволить Вам отримати відповідний Сертифікат.

TestProvider

Q&A

Дякую за увагу!