

# Python Starter

Функції. Частина 2

# Python Starter

## План заняття

1. Документаційні рядки (docstrings)
2. Що таке простір імен?
3. Види областей видимості
4. Локальні та глобальні змінні
5. Вкладена (нелокальна) та вбудована області видимості
6. Вбудовані функції
7. Оператори `nonlocal` і `global`
8. Рекурсія

# Python Starter

Після уроку обов'язково



Повторіть цей урок у відео форматі на [ITVDN.com](http://itvdn.com)

Доступ можна отримати через керівництво вашого навчального центру



Перевірте, як Ви засвоїли цей матеріал на [TestProvider.com](http://testprovider.com)

# Python Starter

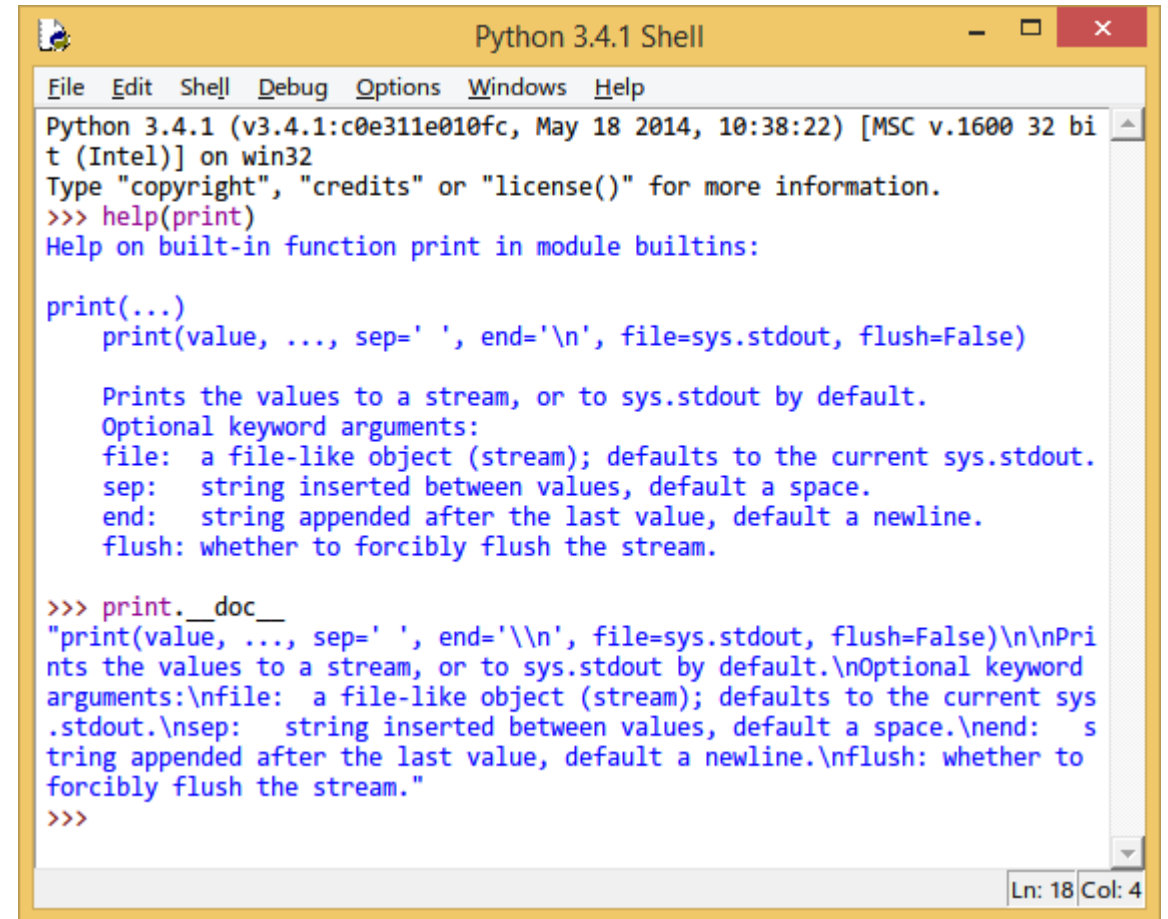
Тема

Функції

# Python Starter

## Документаційні рядки

- Рядок, що стоїть на початку функції (а також модуля, класу або методу), відіграє роль особливого виду коментарів - документаційного рядка (docstring).
- На відміну від звичайних коментарів, до документаційних рядків можна отримати доступ під час виконання програми.
- Доступ напряму до документаційних рядків здійснюється шляхом звернення до поля `__doc__` відповідних об'єктів.
- Працюючи з інтерпретатором в інтерактивному режимі зручно використовувати функцію `help`.



```
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> help(print)
Help on built-in function print in module builtins:

print(...)
    print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

    Prints the values to a stream, or to sys.stdout by default.
    Optional keyword arguments:
    file: a file-like object (stream); defaults to the current sys.stdout.
    sep: string inserted between values, default a space.
    end: string appended after the last value, default a newline.
    flush: whether to forcibly flush the stream.

>>> print.__doc__
"print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)\n\nPrints the values to a stream, or to sys.stdout by default.\nOptional keyword arguments:\nfile: a file-like object (stream); defaults to the current sys.stdout.\nsep: string inserted between values, default a space.\nend: string appended after the last value, default a newline.\nflush: whether to forcibly flush the stream."
>>>
```

# Python Starter

## Стандартні функції

<https://docs.python.org/3/library/functions.html>

<a href="#"><u>abs()</u></a>	<a href="#"><u>dict()</u></a>	<a href="#"><u>help()</u></a>	<a href="#"><u>min()</u></a>	<a href="#"><u>setattr()</u></a>
<a href="#"><u>all()</u></a>	<a href="#"><u>dir()</u></a>	<a href="#"><u>hex()</u></a>	<a href="#"><u>next()</u></a>	<a href="#"><u>slice()</u></a>
<a href="#"><u>any()</u></a>	<a href="#"><u>divmod()</u></a>	<a href="#"><u>id()</u></a>	<a href="#"><u>object()</u></a>	<a href="#"><u>sorted()</u></a>
<a href="#"><u>ascii()</u></a>	<a href="#"><u>enumerate()</u></a>	<a href="#"><u>input()</u></a>	<a href="#"><u>oct()</u></a>	<a href="#"><u>staticmethod()</u></a>
<a href="#"><u>bin()</u></a>	<a href="#"><u>eval()</u></a>	<a href="#"><u>int()</u></a>	<a href="#"><u>open()</u></a>	<a href="#"><u>str()</u></a>
<a href="#"><u>bool()</u></a>	<a href="#"><u>exec()</u></a>	<a href="#"><u>isinstance()</u></a>	<a href="#"><u>ord()</u></a>	<a href="#"><u>sum()</u></a>
<a href="#"><u>bytearray()</u></a>	<a href="#"><u>filter()</u></a>	<a href="#"><u>issubclass()</u></a>	<a href="#"><u>pow()</u></a>	<a href="#"><u>super()</u></a>
<a href="#"><u>bytes()</u></a>	<a href="#"><u>float()</u></a>	<a href="#"><u>iter()</u></a>	<a href="#"><u>print()</u></a>	<a href="#"><u>tuple()</u></a>
<a href="#"><u>callable()</u></a>	<a href="#"><u>format()</u></a>	<a href="#"><u>len()</u></a>	<a href="#"><u>property()</u></a>	<a href="#"><u>type()</u></a>
<a href="#"><u>chr()</u></a>	<a href="#"><u>frozenset()</u></a>	<a href="#"><u>list()</u></a>	<a href="#"><u>range()</u></a>	<a href="#"><u>vars()</u></a>
<a href="#"><u>classmethod()</u></a>	<a href="#"><u>getattr()</u></a>	<a href="#"><u>locals()</u></a>	<a href="#"><u>repr()</u></a>	<a href="#"><u>zip()</u></a>
<a href="#"><u>compile()</u></a>	<a href="#"><u>globals()</u></a>	<a href="#"><u>map()</u></a>	<a href="#"><u>reversed()</u></a>	<a href="#"><u>__import__()</u></a>
<a href="#"><u>complex()</u></a>	<a href="#"><u>hasattr()</u></a>	<a href="#"><u>max()</u></a>	<a href="#"><u>round()</u></a>	
<a href="#"><u>delattr()</u></a>	<a href="#"><u>hash()</u></a>	<a href="#"><u>memoryview()</u></a>	<a href="#"><u>set()</u></a>	

# Python Starter

## Простір імен

- Простір імен Python – це контейнери для імен відображення об'єктів. У Python все — це об'єкт, і ми вказуємо ім'я на об'єкт, щоб ми могли отримати доступ до нього в разі потреби.
- Можна уявити простір імен у якості словника, де ключ – ім'я змінної, а значення – це об'єкт, що з ним пов'язаний.

```
namespace = {"name1":object1, "name2":object2}
```

- У Python кілька незалежних просторів імен можуть існувати одночасно. Імена змінної можуть бути використані у цих просторах імен.

```
function_namespace = {"name1":object1, "name2":object2}
```

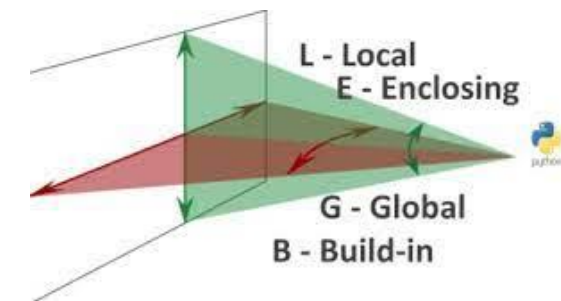
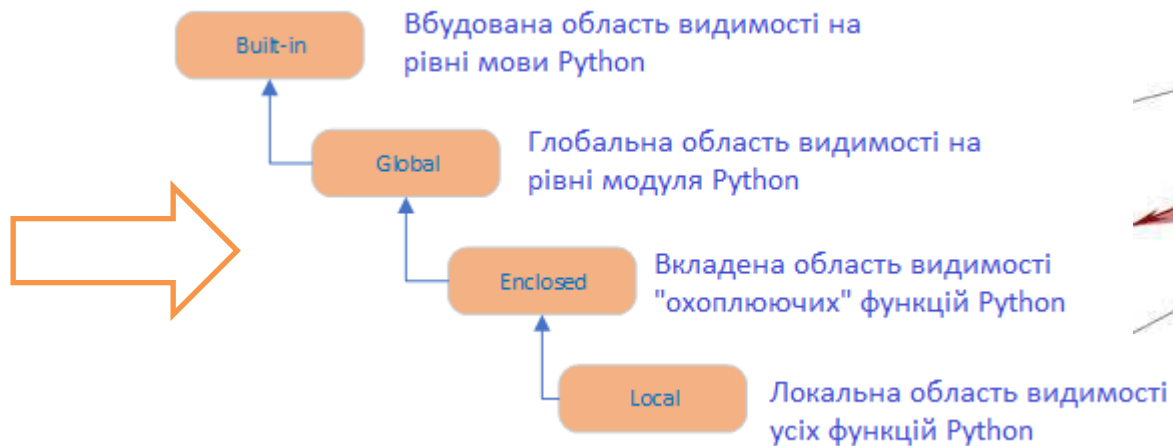
```
for_loop_namespace = {"name1":object3, "name2":object4}
```

# Python Starter

## Область видимості

- Область видимості (англ. scope) позначає область програми, у межах якої ідентифікатор (ім'я) деякої змінної продовжує бути пов'язаним із цією змінною і повертати її значення. За межами області видимості той самий ідентифікатор може бути пов'язаний з іншою змінною, або бути вільним (не пов'язаним з жодною з них).
- У мовах, що підтримують структурне програмування, змінні зазвичай поділяються на два типи за областю видимості :

- глобальні змінні;
- локальні змінні;
- але в мові Python їх більше:





# Python Starter

## Локальні змінні

- Локальні змінні оголошуються всередині функції та недоступні зовні.
- Змінні локальної області видимості використовуються, щоб уникнути проблем із побічними ефектами, які можуть статися з глобальними змінними.
- У Python областю видимості локальної змінної є функція. У деяких мовах будь-який блок коду може мати свої локальні змінні.
- Операція присвоєння функції створює локальну змінну. Якщо потрібно змінити значення змінної з іншої області видимості, слід скористатися операторами `global` або `nonlocal`.

```
def function():  
    variable = 42
```

←  
локальна змінна

# Python Starter

## Вкладені функції

У Python можна оголошувати функції всередині інших функцій. Вони мають доступ до змінних та аргументів зовнішньої функції та недоступні за межами тих функцій, у яких були визначені.

```
def outer_function():  
  
    def inner_function():  
        print('Внутрішня функція')  
  
    print('Зовнішня функція')  
    inner_function()
```

# Python Starter

## Вкладені змінні

- Вкладені змінні оголошуються всередині зовнішніх функцій і доступні в межах існування зовнішньої функції

```
def outer_function():  
    variable = 42  
  
    def inner_function():  
        print('Внутрішня функція')  
        print(variable)  
  
    print('Зовнішня функція')  
    inner_function()
```

← вкладена змінна

# Python Starter

## Глобальні змінні

- Глобальні змінні оголошуються поза всіма функціями і доступні всюди.
- Використання глобальних змінних має недоліки: глобальна змінна може бути змінена у будь-якій точці програми, що може вплинути на роботу інших частин програми. З цієї причини глобальні змінні мають необмежений потенціал для створення взаємних залежностей, що призводить до ускладнення програми.

глобальна змінна



```
variable = 42
```

```
def function():  
    pass
```

# Python Starter

## Built-in

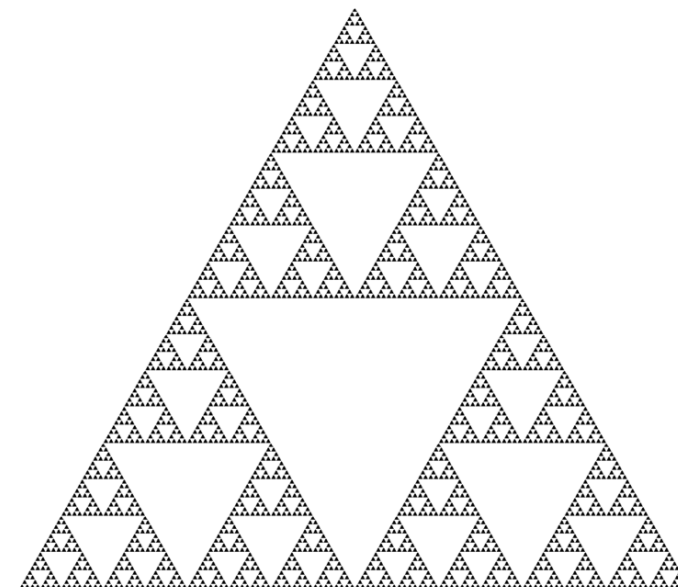
- Вбудовані функції в інтерпретатор доступні всюди.
- Для їх використання не потрібно імпортувати модулі (див. слайд 6).

```
ws.on("message", m => {  
  let a = m.split(" ")  
  switch(a[0]){  
    case "connect":  
      if(a[1]){  
        if(clients.has(a[1])){  
          ws.send("connected");  
          ws.id = a[1];  
        }else{  
          ws.id = a[1]  
          clients.set(a[1], {clients: (clients) => {  
            ws.send("connected")
```

# Python Starter

## Рекурсія

- Пам'ять під локальні змінні виділяється під час кожного виклику функції. Це уможливорює рекурсію.
- Рекурсія - виклик функції з неї ж самої, безпосередньо (проста рекурсія) або через інші функції (складна або непряма рекурсія), наприклад, функція A викликає функцію B, а функція B - функцію A.
- Кількість вкладених викликів функції або процедури називається глибиною рекурсії.
- Рекурсивна програма дозволяє описати повторюване і навіть потенційно нескінченне обчислення, причому без явних повторень частин програми та використання циклів.



# Дивіться наші уроки у відео форматі

## ITVDN.com



Перегляньте цей урок у відео форматі на освітньому порталі [ITVDN.com](http://itvdn.com) для закріплення пройденого матеріалу.

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



# Перевірка знань

TestProvider.com

TestProvider

Мы помогаем людям оценить себя

Регистрация Войти

Поиск сертификата

Главная Услуги и цены Центр Тестирования Поддержка О нас

Мы в социальных сетях

## Тестирование

Языки программирования и информационные технологии

**Microsoft**

C# ASP.NET MVC JavaScript Patterns OF Design SQL Architecture Guide WCF HTML&CSS XML SEO WPF HTML5&CSS3 JQuery XNA SharePoint GUI for Android Windows Azure Platform Microsoft Patterns&Practices TFS SCRUM ReSharper TDD WWF LINQ Entity Framework Windows Forms Refactoring Microsoft Expression Blend 4 Windows Phone 8 Windows 8 AppStore Visual Studio Tips&Tricks MSF MEF SilverLight AJAX MEF Service Oriented Architecture

Добро пожаловать на TestProvider.com!

Сайт перенесен на новую облачную платформу с использованием системы единой авторизации Single Sign On. Если вы хотите восстановить статистику по предыдущим экзаменам обратитесь в [службу поддержки](#). Для восстановления информации с предыдущей версии сайта, просба написать в службу поддержки Ваш старый и новый логины.

ITVDN PROMETRIC TEST CENTER CyberBionic Microsoft Partner Windows Azure Cloud Partner EBA

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 проходите тестування для перевірки знань на [TestProvider.com](http://TestProvider.com)

Успішне проходження фінального тестування дозволить Вам отримати відповідний Сертифікат.





# Python Starter

Q&A

# Python Starter

Дякую за увагу!

# Інформаційний відеосервіс для розробників програмного забезпечення

