Функції (частина 2)

№ уроку: 9 **Курс:** Python Starter

Засоби навчання: PyCharm

Огляд, мета та призначення уроку

Після завершення уроку учні матимуть уявлення про більш просунуте використання функцій, розглянуть поняття рекурсії, а також розглянуть на практиці деякі стандартні функції мови Python.

Вивчивши матеріал даного заняття, учень зможе:

- Створювати особливий вид коментарів – документаційні рядки
- Використовувати стандартні функції мови Python
- Розуміти різницю між локальними та глобальними змінними
- Використовувати рекурсію

Зміст уроку

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

Резюме

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.
Напряму доступ до документаційних рядків здійснюється шляхом звернення до поля __doc__ відповідних об'єктів:

ім'я_функції.__doc__

Зверніть увагу на відсутність круглих дужок після імені функції. Круглі дужки – це операція виклику функції, але за їх відсутності сама функція сприймається як деякий об'єкт.
Працюючи з інтерпретатором в інтерактивному режимі зручно використовувати функцію help.
Стандартна бібліотека Python містить величезну кількість різних модулів та бібліотек. Але є й особливо важливі стандартні функції, які вбудовані в мову і доступні завжди без імпортування.
Деякі з них:

- abs(x) – модуль числа x
- bin(x) – подання числа x у двійковій системі числення
- bool(x) – створити значення типу bool із x
- callable(f) – перевірити, чи можна викликати f як функцію
- chr(code) – символ із кодом code
- complex(real, imag) – створити комплексне число
- dir(obj) – вивести список полів та методів об'єкту obj
- float(x) – створити дійсне число з x
- format(x, fmt) – повертає рядок, що представляє значення x, відформатоване відповідно до форматного рядка fmt

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

- help(obj) – виводить довідку по об'єкту obj
- hex(x) – шістнадцяткове представлення числа x
- id(obj) – значення, унікальне для кожного об'єкта (у CPython – адреса об'єкта в пам'яті)
- input() – введення даних
- int(x) – створити ціле число
- len(s) – довжина рядка або будь-якої іншої послідовності
- max(arg1, arg2, …) – максимальне число серед заданих
- min(arg1, arg2, …) – мінімальне число серед заданих
- oct(x) – подання числа x у восьмеричній системі числення
- ord(c) – код символу c
- pow(x, n) – число x у ступені n
- print() – виведення даних
- range() – послідовність цілих чисел (див. урок про цикли)
- repr(obj) – рядкове представлення об'єкта
- reversed(iterable) – обхід послідовності у зворотному порядку
- round(number, ndigits) – округлення числа
- sorted(iterable) – сортує послідовність у порядку зростання чи спадання значень
- str(x) – створення рядка з x

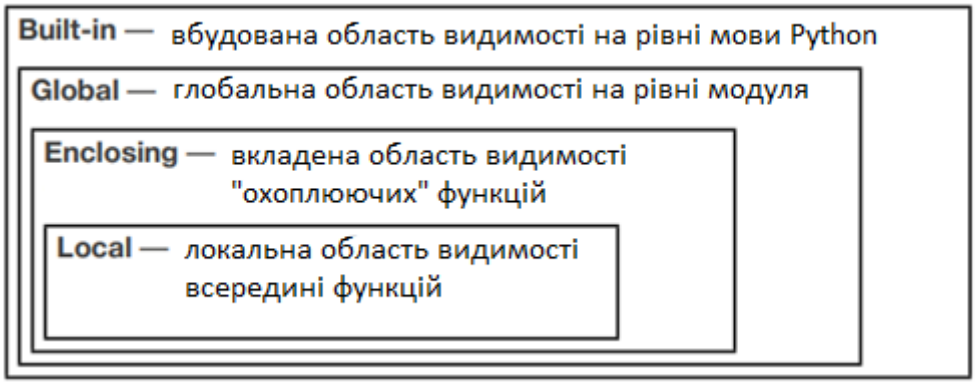
Повний список можна знайти в документації у розділі Built-in Functions розділу Library Reference. У Python можна оголошувати функції всередині функцій.

В момент використання ідентифікатора змінних у програмі, Python щоразу відшукує, створює / змінює ці імена у відповідному просторі імен. Простір імен, який доступний у кожний момент, залежить від області, де знаходиться код.

Область видимості (англ. scope) позначає область програми, у межах якої ідентифікатор (ім'я) деякою змінною 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:

- **локальні змінні** — оголошуються всередині функції та недоступні зовні;
- **глобальні змінні** — оголошуються поза всіма функціями та доступні звідусіль.

Але насправді їх більше.



Локальна область видимості (Local) - це частина коду, яка відповідає тілу деякої функції або лямбда-виразу (розглядатимемо в уроках далі).
Ідентифікатори, визначені всередині функції, відносяться до її локальної області видимості та існують тільки в рамках цієї функції або її локальної області.

CyberBionic systematics
CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Ця область створюється не при оголошенні функції, а при її виклиці (тобто, скільки було викликів функції, стільки і різних, нових, локальних областей). Після роботи функції локальна область видимості знищується, а ідентифікатори «забуваються».
Вкладена (нелокальна) область видимості (Enclosing) (була додана в Python 3) — ця область видимості виникає під час роботи вкладених функцій і містить ідентифікатори, які ми визначаємо у вкладеній функції. Ідентифікатори цієї області «є видними» з коду внутрішніх та «охоплюючих» функцій (всередині яких є інші внутрішні функції).
Глобальна область видимості (Global) – це рівень програми (скрипта, модуля) Python, який містить усі ідентифікатори, які ми визначили на рівні програми (тобто, поза функціями, якщо вони є). Ця область створюється під час запуску нашої програми. Ідентифікатори, визначені в цій галузі, будуть доступні в будь-якому місці програмного коду. Під час виконання програми існує єдина глобальна область видимості, яка зберігається, поки наша програма не завершиться, після чого всі ідентифікатори цієї області будуть забуті.
Вбудована область видимості (Built-in) – це рівень мови Python, тобто всі вбудовані об'єкти, функції Python знаходяться в цій області. Ця область активується в момент запуску інтерпретатора Python і всі вбудовані об'єкти автоматично завантажуються до неї. Таким чином ми можемо застосовувати їх (наприклад, вбудовану функцію abs()) без використання операції імпорту якого-небудь модуля.

Використання глобальних змінних має недоліки: глобальна змінна може бути змінена у будь-якій точці програми, що може вплинути на роботу інших частин програми. Тому глобальні змінні мають необмежений потенціал для створення взаємних залежностей, що призводить до ускладнення програми.

Змінні локальної області видимості використовуються, щоб уникнути проблем із побічними ефектами, які можуть статися з глобальними змінними.
Глобальні змінні широко використовуються для передачі даних між секціями коду, які беруть участь у відносинах викликів, такі як паралельні нитки виконання або обробники сигналів.
У Python областю видимості локальної змінної є функція. У деяких мовах будь-який блок коду може мати локальні змінні.
Операція присвоєння функції створює локальну змінну. Якщо потрібно змінити значення змінної з іншої області видимості, скористайтеся операторами global або nonlocal.
Змінні, зазначені в операторі Global, розглядаються як глобальні.
Змінні, зазначені в операторі nonlocal, розглядаються як змінні з найближчої області видимості (зовнішня функція у разі вкладених функцій або глобальна область видимості, якщо функція є глобальною).
Пам'ять під локальні змінні виділяється під час кожного виклику функції. Це уможливлює рекурсію.
Рекурсія - виклик функції з неї 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 використання циклів.
Реалізація рекурсивних викликів функцій у практично застосовуваних мовах, зазвичай, спирається на механізм стеку викликів — адресу повернення і локальні змінні функції, записуються в стек, завдяки чому кожен наступний рекурсивний виклик цієї функції користується своїм набором локальних змінних і за рахунок цього працює коректно. Зворотною стороною цього досить простого за структурою механізму є те, що на кожен рекурсивний виклик потрібна деяка кількість оперативної пам'яті комп'ютера, і за надмірно великої глибини рекурсії може настати переповнення стеку викликів.
Питання бажаності використання рекурсивних функцій у програмуванні неоднозначний: з одного боку, рекурсивна форма, можливо, структурно простіша і наочніша, особливо, коли сам реалізований алгоритм по суті рекурсивний. З іншого боку, зазвичай рекомендується уникати рекурсивних програм, які призводять (або в деяких умовах можуть призводити) до надто великої глибини рекурсії.
У рекурсивних функціях має бути умова виходу з рекурсії, тобто щонайменше одна з гілок коду має повертати результат, а не викликати функцію рекурсивно.

CyberBionic systematics
CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com

Закріплення матеріалу

- Що таке документаційний рядок?
- Чи можна в Python оголошувати функції всередині інших функцій?
- Назвіть кілька стандартних функцій Python.
- Що таке область видимості?
- Які області видимості існують?
- Для чого необхідні оператори global, nonlocal?
- Що таке локальна змінна?
- Що таке глобальна змінна?
- Що таке рекурсія?
- Що потрібно для того, щоб рекурсивна функція виконалася за кінцевий час?

Додаткове завдання

Завдання 1
Напишіть рекурсивну функцію, яка обчислює суму натуральних чисел, які входять до заданого проміжку.

Завдання 2
Створіть функцію quadratic_equation, яка приймає на вхід 3 параметри: a, b, c. Усередині цієї функції створити змінні x1, x2 зі значенням None (спочатку приймаємо, що рівняння не має коренів) та функцію calc_rezult з формальними параметрами зовнішньої функції quadratic_equation. Всередині функції calc_rezult здійснити пошук дискримінанта, згідно з результатом якого зробити розрахунок коренів рівняння. Зовнішня функція quadratic_equation має повернути перелік значень коренів квадратного рівняння. Надати можливість користувачеві ввести з клавіатури формальні параметри для передачі їх у створену функцію quadratic_equation, результати роботи функції відобразити на екрані.

Самостійна діяльність учня

Завдання 1
Прочитайте у документації з мови Python інформацію про перелічені у резюме цього уроку стандартні функції. Перевірте їх на практиці.

Завдання 2
Створіть програму, яка перевіряє, чи є паліндромом введена фраза.

Завдання 3
Нехай на кожну сходинку можна стати з попередньої або переступивши через одну. Визначте, скількома способами можна піднятися на задану сходинку.

Рекомендовані ресурси

Документація з Python
https://docs.python.org/3/library/functions.html
https://docs.python.org/3/reference/simple_stmts.html#the-global-statement
https://docs.python.org/3/reference/simple_stmts.html#the-nonlocal-statement

Статті у Вікіпедії про ключові поняття, розглянуті на цьому уроці
https://ru.wikipedia.org/wiki/Область_видимости
https://uk.wikipedia.org/wiki/Локальна_змінна
https://uk.wikipedia.org/wiki/Глобальна_змінна
https://uk.wikipedia.org/wiki/Рекурсія

CyberBionic systematics
CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine
Tel. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com