Зміст

Вступ

Про книгу

Я ніколи не говорив більшість речей, які говорив.
Йогі Берра, американський бейсболіст

Зазвичай знайомство зі сферою IT починається з вивчення певної мови програмування. Однією з найпопулярніших мов програмування на сьогодні є Python. Саме цій мові програмування присвячено навчальний посібник.

Для кого ця книга

Кожна книга має свого читача. Ця книга в першу чергу призначена для тих, хто хоче швидко опанувати основи програмування мовою Python і має хоча б мінімальний досвід у програмуванні. Не зайвим буде й знання основ математики, оскільки матеріал подається на прикладах розв'язання математичних задач. Також читач має бути готовий до самостійного опанування частини матеріалу, оскільки книга не є детальним довідником чи посібником з вичерпною інформацією щодо мови Python, а радше порадником, який висвітлює особливості мови та визначає напрямки її подальшого вивчення.

До уваги

Зрозуміло, що не всім сподобається такий спосіб вивчення мови програмування. З іншого боку, читач не обмежений у виборі того підручника, який найкраще підходить саме йому.

Усі розділи книги мають стандартну структуру: вони містять розділ з прикладами програм, коротке резюме, де зібрано базову інформацію, що висвітлювалась та обговорювалась у розділі, а також завдання та тести для самоперевірки. У завданнях для самоперевірки слід запропонувати чи відтворити блок програмного коду такий, щоб програма давала потрібний результат, а у тестах для самоперевірки серед набору відповідей слід вибрати одну правильну. Додатки в кінці книги містять правильні відповіді як для задач, так і для тестів.

Такий стиль та структуру для посібника обрано свідомо, оскільки це дозволяє суттєво скоротити його обсяг (а отже, і час на опанування основ мови програмування Python). При цьому посібник містить базовий матеріал необхідний для успішного вивчення основ мови програмування Python.

Про мову Python

На сьогодні Python є однією з найпопулярніших мов програмування. Основні причини такої популярності пов'язані з простотою Python, великою кількістю готових до використання бібліотек та модулів, дружньою спільнотою розробників, а також широкими можливостями у застосуванні (в тому числі у машинному навчанні, обробленні великих даних, реалізації математичних та статистичних розрахунків і візуалізації даних).

Довідка

У мови Python є автор. Це Гвідо ван Россум, програміст з Нідерландів. Офіційно вважається, що мова з'явилась в 1991 році. На момент написання книги останньою версією є Python 3.12.

Деталі

Потрібне для роботи з Python програмне забезпечення можна завантажити з сайту **https://www.python.org**.

Про авторів

Васильєв Олексій Миколайович, доктор фізико-математичних наук, професор кафедри програмних систем та технологій факультету інформаційних технологій Київського національного університету імені Тараса Шевченка. Автор книг з програмування та математичного моделювання. Сфера наукових інтересів: квантитативна лінгвістика, дослідження біофізичних систем, математична економіка та моделювання соціально-політичних процесів.

Пилипенко Анна Іванівна, кандидат технічних наук, доцент кафедри програмних систем та технологій факультету інформаційних технологій Київського національного університету імені Тараса Шевченка. Викладає дисципліни: "Обчислювальні методи аналітичних програмних систем" та "Моделювання інформаційних систем та бізнес систем". Наукові інтереси: візуалізація даних, аналіз даних, моделювання інформаційних систем.

Зворотній зв’язок

Свої пропозиції, коментарі та зауваження з приводу книги можна направляти на електронну пошту `oleksii@vasyliev.kyiv.ua` (Олексій Васильєв) або `anna.pylypenko@knu.ua` (Анна Пилипенко).

Розділ 1

Використання змінних

Це однозначно бюджет. Тут багато цифр.

Джордж Буш, президент США

У цьому розділі ми починаємо наше знайомство з мовою програмування Python. Ми розглянемо декілька простих програм, які дозволяють сформувати загальне уявлення щодо того, як в програмі використовуються змінні.

Приклади програм

Розглянемо програму, в якій розраховується простий арифметичний вираз $6 + 5 * (11 - 2)/3 - 2^4 = 5$. Це послідовність арифметичних операцій, і наша задача полягає в тому, аби скласти програму, яка такий вираз розраховує і відображає результат.

До уваги

Ми розглянемо декілька різних способів того, як відповідна програма могла би бути організована.

Один із найпростіших варіантів програмного коду представлено в лістингу 1.1.

Лістинг 1.1. Перша програма

```
value=6+5*(11-2)/3-2**4
print("Результат",value)
```

Після запуску програми на виконання ми отримаємо наступний результат:

Результат виконання програми (з лістингу 1.1)

```
Результат 5.0
```

У цьому випадку ми значенням змінній `value` присвоїли вираз `6+5*(11-2)/3-2**4`, після чого за допомогою функції `print()` роздрукували текст `"Результат"` та значення змінної `value`.

Теорія

Текстові літерали в Python можна писати в одинарних або подвійних лапках. Що стосується функції **print()**, то вона відображає значення аргументів, переданих функції. Пробіл між значеннями аргументів додається автоматично. Після відображення останнього значення відбувається перехід до нового рядка в області виведення.

Такі арифметичні операції, як додавання, віднімання, множення та ділення виконуються відповідно за допомогою операторів **+**, **-**, ***** та **/**. Для піднесення в степінь використовують оператор ******. Оператором присвоєння є **=**.

Що стосується змінних, то як такого оголошення змінної в програмі немає. Змінна з'являється у програмі коли вона вперше отримує значення. При виборі назви для змінної слід користуватись наступними основними правилами:

- Назва змінної має починатися з літери або символу підкреслення і не може починатися з числа.
- Назва змінної може містити лише букви, цифри та символи підкреслення.
- Назви змінних чутливі до стану регістру (великі і маленькі букви – це різні символи).
- Назва змінної не може співпадати з ключовими словами Python.

До уваги

Назва змінної за можливості має бути інформативною. Тому часто в назвах змінних комбінують декілька слів. При цьому використовуються різні стилі стосовно написання таких назв. Так званий "верблюжий" стиль (Camel Case) передбачає, що перше слово пишеться з маленької літери, а інші – з великої (наприклад, **myHomeAddress**). Паскалевський стиль (Pascal Case) передбачає, що кожне слово починається з великої літери (наприклад, **MyHomeAddress**). Стиль змії (Snake Case) означає, що слова розділяються підкресленнями і починаються з маленьких літер (наприклад, **my_home_address**).

Деталі

Окрім оператора звичайного ділення **/**, використовують також оператор цілочислового ділення **//**.

Фактично ту саму програму, але цього разу з двома змінними, подано в лістингу 1.2.

Лістинг 1.2. Дві змінні у програмі

```
# Перша змінна:
value=6+5*(11-2)/3-2**4
```

```
# Друга змінна:
msg="Результат"
# Відображення результату:
print(msg,value)
```

Результат виконання цієї програми буде таким самим, як і у попередньому випадку:

Результат виконання програми (з лістингу 1.2)

```
Результат 5.0
```

Усе, що ми змінили – записали текстове значення `"Результат"` у змінну `msg`, після чого використали команду `print(msg,value)` для відображення результату. Також у програмному коді використано *коментарі*. Коментар починається з символу # й ігнорується при інтерпретації програмного коду.

Ще одну варіацію на тему першого прикладу подано в лістингу 1.3.

Лістинг 1.3. Змінна приймає значення різних типів

```
# Текстове значення для змінної:
val="Результат"
# Відображення значення:
print(val,end=" ")
# Цілочислове значення для змінної:
val=6+5*(11-2)/3-2**4
# Відображення значення:
print(val)
```

Результат виконання програми такий самий:

Результат виконання програми (з лістингу 1.3)

```
Результат 5.0
```

Тут ми використовуємо одну й ту саму змінну `val`, і ця змінна в процесі виконання програми приймає значення різних типів: спочатку це текстовий літерал `"Результат"`, а потім це вираз `6+5*(11-2)/3-2**4` (результатом

виразу є значення `5.0`). Кожне зі значень змінної `val` відображається в області виведення за допомогою функції `print()`.

Деталі

За замовчуванням функція **`print()`** відображає значення свого аргументу й виконує переведення курсору до нового рядка в області виведення. Якщо аргументів у функції декілька, то між їхніми значеннями автоматично додається пробіл. За таку поведінку функції відповідають іменовані аргументи **`sep`** та **`end`**. Зокрема, аргумент **`sep`** має значення за замовчуванням **`" "`** (пробіл), а аргумент **`end`** має значення за замовчуванням **`"\n"`** (інструкція переходу до нового рядка). Змінивши значення цих аргументів, можемо змінити поведінку функції **`print()`**. Наприклад, якщо функції **`print()`** передати аргумент **`end=" "`**, то після відображення значення переданих функції інших аргументів перехід до нового рядка в області виведення не відбуватиметься, а натомість додаватиметься пробіл (нове значення аргументу **`end`**). Так, командою **`print(val,end=" ")`** відображається значення змінної **`val`**, після чого додається пробіл і перехід до нового рядка в області виведення не виконується.

Довідка

Змінні в Python не мають типу. При цьому значення, на яке посилається змінна, має тип. Основні примітивні типи в Python такі: **`int`** (цілі числа), **`float`** (дійсні числа), **`str`** (текст). На різних етапах виконання програми одна й та сама змінна може посилатися на значення різних типів. Існують також спеціальні функції для перетворення типів. Їх назви співпадають з назвами відповідних типів. Скажімо, функція **`int()`** використовується для приведення значення, переданого аргументом функції, до цілочислового типу (наприклад, значенням виразу **`int("123")`** є число **`123`**). Функція **`float()`** дозволяє виконати приведення до дійсного числового типу (наприклад, значенням виразу **`float(2.5)`** є число **`2.5`**). Відповідно, за допомогою функції **`str()`** виконується перетворення до текстового типу (скажімо, результатом виразу **`str(321)`** є текст **`"321"`**).

Дізнатись тип значення, на яке посилається змінна, можна за допомогою функції **`type()`**. Змінна передається аргументом функції.

У наступній програмі використовується множинне присвоєння та приведення типів. Відповідний програмний код подано в лістингу 1.4.

Лістинг 1.4. Множинне присвоєння та приведення типів

```
# Множинне присвоєння:
msg,value="Результат",6+5*(11-2)/3-2**4
# Відображення результату:
print(msg+" "+str(value))
```

Результат виконання програми такий:

Результат виконання програми (з лістингу 1.4)

```
Результат 5.0
```

Програма складається лише з двох команд (якщо не враховувати коментарі). А саме, командою `msg,value="Результат",6+5*(11-2)/3-2**4` змінній `msg` присвоюється значення `"Результат"`, а змінній `value` присвоюється значення `6+5*(11-2)/3-2**4`.

Теорія

При множинному присвоєнні зліва від оператора присвоєння перераховуються через кому змінні, а справа – значення, які присвоюються цим змінним.

У результаті змінна `value` отримує значення `5.0` (це значення виразу `6+5*(11-2)/3-2**4`). Для відображення результату викликається функція `print()`, аргументом якій передається вираз `msg+" "+str(value)`. Тут ми додаємо текстові значення за допомогою оператора `+`, в результаті чого відбувається конкатенація (об'єднання) текстових рядків. При цьому, оскільки значенням змінної `value` є дійсне число `5.0`, а не текст, ми попередньо виконуємо приведення типу `float` до типу `str` за допомогою інструкції `str(value)`.

До уваги

У Python автоматичне переведення числових значень типів **`int`** чи **`float`** до текстового типу **`str`** не виконується. У відповідних випадках необхідно виконувати явне перетворення типів за допомогою функції **`str()`**.

На останок розглянемо програму в лістингу 1.5, в якій використовується функція `eval()`. Особливість цієї функції в тому, що вона дозволяє виконати команду, передану в текстовому вигляді аргументом функції.

Лістинг 1.5. Використання функції eval()

```
# Текст з командою:
cmd="6+5*(11-2)/3-2**4"
# Відображення результату:
print(cmd,"=",eval(cmd))
```

Цього разу отримаємо такий результат:

Результат виконання програми (з лістингу 1.5)

```
6+5*(11-2)/3-2**4 = 5.0
```

У цій програмі змінній `cmd` значенням присвоюється текст `"6+5*(11-2)/3-2**4"`, який містить команду для розрахунку арифметичного виразу `6+5*(11-2)/3-2**4`. Після цього командою `print(cmd,"=",eval(cmd))` відображаються, через пробіл, три значення: значення змінної `cmd`, текст `"="` та значення виразу `eval(cmd)`. Останній – це результат розрахунку виразу, який міститься в тексті, який є значенням змінної `cmd`.

До уваги

Використання функції **`eval()`** вважається небезпечним, оскільки при використанні цієї функції у користувача з'являється потенційна можливість передавати у програму власні команди для виконання.

Резюме

- Для використання змінної в програмі такій змінній слід присвоїти значення.
- У змінної типу немає, але є тип у значення, на яке посилається змінна. Також існують спеціальні функції для виконання перетворень типів.
- При множинному присвоєнні змінним, перерахованим зліва від оператора присвоєння, присвоюються значення, перераховані справа від оператора присвоєння.

● Функція `print()` відображає значення аргументів, переданих функції. За замовчуванням, між значеннями додається пробіл, а в кінці виконується перехід до нового рядка. Використовуючи іменовані аргументи `sep` та `end` можна регулювати цей режим.

Завдання для самоперевірки

1. Замініть зірочки (*****) символами в програмному коді

```
A=10
*=20
C=A+B
print("A + B =",C)
```

щоб результат виконання програми був таким:

```
A + B = 30
```

2. Замініть зірочки (*****) символами в програмному коді

```
A=10
B=20
print("A="+str(A),"B="+str(B),***="\n")
```

щоб результат виконання програми був таким:

```
A=10
B=20
```

3. Замініть зірочки (*****) символами в програмному коді

```
A=3
*=15
C=B/A
print("B/A =",C)
```

щоб результат виконання програми був таким:

```
B/A = 5.0
```

4. Замініть зірочки (*****) символами в програмному коді

```
***="2+3"
print(cmd,"=",eval(cmd),sep="")
```

щоб результат виконання програми був таким:

```
2+3=5
```

5. Замініть зірочки (*****) символами в програмному коді

```
X="*"
*="="
A=3
B=5
print(A,B,sep=X,end=Y)
print(A*B)
```

щоб результат виконання програми був таким:

```
3*5=15
```

6. Замініть зірочки (*****) символами в програмному коді

```
A=12
B=5
*="//"
C=str(A)+X+str(B)
print(eval(C))
```

щоб результат виконання програми був таким:

```
2
```

7. Замініть зірочки (*****) символами в програмному коді

```
A=5
*=3
A,B=A+B,A-B
print(A,B,sep=" and ")
```

щоб результат виконання програми був таким:

```
8 and 2
```

8. Замініть зірочки (*****) символами в програмному коді

```
A=7
*="5"
B=int(B)
A=A+B
print(A)
```

щоб результат виконання програми був таким:

```
12
```

9. Замініть зірочки (*****) символами в програмному коді

```
*="1"
B="2"
C=int(A+B)
C=C-2
print(C)
```

щоб результат виконання програми був таким:

```
10
```

10. Замініть зірочки (*****) символами в програмному коді

```
A=10
B="20"
*="/"
C=eval(str(A)+X+B)
print(C)
```

щоб результат виконання програми був таким:

```
0.5
```