Кожна мова програмування має свій стиль і Python не є винятком. Розробники мови є прихильниками певної філософії програмування, яку називають **«The Zen of Python»** («Дзен Пайтона»). Її текст можна отримати у інтерпретаторі Python за допомогою команди:

import this

У більшості мов програмування використовуються спеціальні символи (наприклад, фігурні дужки {}) або ключові слова (наприклад, begin і end) для того, щоб розбити код на частини. У цих мовах хорошим тоном є використання відступів при написанні коду, щоб зробити програму зручною для читання як для себе, так і для інших.

На відміну від багатьох інших мов, Python обов’язково вимагає, щоб блоки коду забезпечувалися відступами.

|  |  |
| --- | --- |
|  | *У Python для побудови структури програми використовуються відступи від лівого краю, які створюються за допомогою пропусків (пробілів).* |

Типова структура програми може мати такий вигляд (крапками позначені пропуски):

n = int(input())

out = []

for i in range(n):

....k = 0

....while k < i + 1:

........out.append(i+1)

........k += 1

....if len(out) > n:

........break

out = out[0:n]

for i in out:

....print(i, end = " ")

Окрім того, для Python визначені рекомендації по стилю написання коду [PEP8](https://goo.gl/xGifHN" \t "_blank) .

При виконанні, програма обробляє дані різного [типу](https://pythonguide.rozh2sch.org.ua/#%D1%82%D0%B8%D0%BF%D1%83) (числові, текстові дані тощо). Тип даних задає як множину можливих значень даних, так і визначає операції, які можуть бути над ними виконані.

Кожне з даних характеризується **розміром виділеної пам’яті для зберігання**, **ім’ям (ідентифікатором)**, **типом** і **значенням**.

Імена дозволяють ідентифікувати дані, тобто відрізняти їх між собою. Тип обирають для кожної величини, яка використовується для подання реальних об’єктів.

Мови програмування мають в своєму арсеналі деякі **прості (вбудовані) типи даних**. Вони використовуються як базові блоки для програм та складних (спеціалізованих) типів даних.

У Python вбудовані такі прості типи даних:

1. **Булевий** (має значення True і False).
2. **Цілі числа** (наприклад, 81, 1000).
3. **Числа з рухомою крапкою** (наприклад, 3.14159, 2.5e8, 4000.0).
4. **Рядки** (послідовності текстових символів, наприклад Hello, Python!).

|  |  |
| --- | --- |
|  | *Для розділення цілої і дробової частини у дійсних числах в Python використовується символ крапки.* |

Кожен тип має специфічні правила використання і вони по-різному обробляються комп’ютером.

3.1. Об’єкти та змінні

В Python усе - булеві значення, цілі числа, числа з рухомою крапкою, рядки, складні структури даних, функції - реалізовано як [об’єкти](https://pythonguide.rozh2sch.org.ua/#objects).

Що таке об’єкт?

Об’єкт можна уявити як скриньку, у якій міститься фрагмент даних. Об’єкт має тип (наприклад, тип булевих значень або тип цілих чисел), який визначає, що можна зробити з цими даними.

У реальному світі скринька з написом «Книги» повідомляє нам інформацію, що в ній містяться книги (фрагменти даних), які ми можемо звідти дістати або покласти нові, але виключно книги.

Точно так само і в Python - якщо об’єкт має тип цілих чисел, ви знаєте, що зможете скласти його з іншим об’єктом, який має такий самий тип цілих чисел.

Тип також визначає, чи можна змінити значення, яке зберігається в скриньці (змінюване значення), або воно є константою (незмінне значення).

Об’єкт, значення якого неможливо змінити, можна порівняти із закритою скринькою з віконцем: ви можете побачити значення, але не можете змінити його. В рамках тієї ж аналогії, об’єкт, значення якого можна змінити, схожий на відкриту скриньку: ви не тільки можете побачити значення, яке там зберігається, а й змінити його, однак не можете змінити тип об’єкта (скриньки).

|  |  |
| --- | --- |
|  | *Про об’єкти у Python та деякі особливості інтерпретатора Python читайте на*[*цій сторінці*](http://cs.rozh2sch.org.ua/objectsinpython.html)*.* |

Мови програмування також дозволяють визначати [змінні](https://pythonguide.rozh2sch.org.ua/#%D0%B7%D0%BC%D1%96%D0%BD%D0%BD%D1%96). Ви можете визначити їх для використання у своїй програмі.

В Python для присвоювання змінній певного значення використовується символ =.

|  |  |
| --- | --- |
|  | *У математиці символ = означає****«дорівнює»****. У багатьох мовах програмування, включаючи Python, цей символ використовується для позначення****«присвоювання»****.* |

У наступному фрагменті програми ціле число 12 присвоюється змінній з ім’ям a, потім на екран виводиться значення, пов’язане в поточний момент з цією змінною:

>>> a = 12

>>> a

12

Дуже розповсюджене твердження, що змінна - це контейнер для зберігання значень. Така аналогія справедлива для багатьох мов програмування (наприклад, C).

З іншого боку, змінні в Python схожі не на контейнери, а на **ярлики** чи **стікери-наклейки**, які прикріплюються до об’єктів із простору імен інтерпретатора Python.

До одного об’єкта може прикріплюватися будь-яка кількість стікерів (змінних), і при зміні цього об’єкта, значення, на які посилаються усі ці змінні, також змінюються.

Розглянемо простий фрагмент програми на Python, щоб зрозуміти, що це означає:

>>> a = [5, 6, 7]

>>> b = a

>>> c = b

1. Змінна a містить список із трьох числових значень.
2. Присвоюємо змінній b значення a.
3. Присвоюємо змінній c значення b.

А тепер змінимо **перше** значення (з 5 на 100) у списку, який зберігається у змінній b:

>>> a = [5, 6, 7]

>>> b = a

>>> c = b

>>> b[0] = 100

Виведемо значення усіх змінних на екран:

>>> a = [5, 6, 7]

>>> b = a

>>> c = b

>>> b[0] = 100

>>> print(a, b, c)

[100, 6, 7] [100, 6, 7] [100, 6, 7]

Якщо вести мову про змінні як контейнери, то як зміна вмісту одного контейнера є причиною до одночасної зміни вмісту двох інших?

Але якщо розуміти, що змінні - це всього-навсього стікери-наклейки (ярлики), які прикріплені до об’єктів, то зміна вмісту об’єкта, до якого відносяться усі три стікери, просто відображається в усіх трьох змінних.

|  |  |
| --- | --- |
|  | *В Python змінні - це просто імена. Присвоювання не копіює значення, воно прикріплює ім’я до об’єкта, який містить дані. Ім’я - це****посилання на якийсь об’єкт****, а не сам об’єкт. Ім’я можна розглядати як стікер-наклейку.* |