
програмування

ТЕМА 1. ЛІНІЙНІ ПРОГРАМИ

Історія

1842 - Ада Августа Лавлейс створила першу програму для аналітичної машини Чарльза Беббіджа



Історія.2

1946 - принципи архітектури сучасних комп'ютерів були обґрунтовані Дж. фон Нейманом




Історія.3

1947 – Дж.М. Хоппер задокументував першу комп'ютерну помилку (bug)

9/9

0800 Action started { 1.2700 9.032 477 025
 1000 stopped - action ✓ { 9.037 846 995 - correct
 1500 032 HP-AC 2.130476415 (62) 7.615725059 (12)
 032 PKO 2 2.130476415
 032 2.130676415
 Relays 6-2 in 032 failed special speed test
 in relay - 11,000 test.
 Relays changed

1100 Started Cosine Tape (Sine check)
 1525 Started Multi-Adder Test.

1545  Relay #70 Panel F
 (moth) in relay.

First actual case of bug being found.
 163000 action started.
 1700 closed down.

Історія.4

1954 – Джон Бекус створив першу мову програмування високого рівня Fortran



Сьогодні

Більш, ніж 2 мільярди комп'ютерів

Більш, ніж половина усього населення Землі користується смартфонами

Більш, ніж 3.7 мільярди користувачів Інтернет

Більше 20 мільйонів програмістів

Програми та програмування

Програма – запис алгоритму у формі, придатній для виконання комп'ютером.

Програмування – процес побудови комп'ютерних програм.

- У більш широкому сенсі під програмуванням розуміють весь спектр діяльності, пов'язаний зі створенням і підтримкою в робочому стані програмного забезпечення (software engineering)

Інформація

Інформація - це поняття, що передбачає наявність

- матеріального носія інформації,
- джерела і передавача інформації,
- приймача інформації,
- каналу зв'язку між джерелом і приймачем інформації.

Алгоритми та виконавці

Алгоритм - точний припис, який визначає зміст і послідовність кроків, що переводять задану сукупність початкових даних у шуканий результат за скінченний час.

Виконавцем ми будемо називати пристрій, здатний виконувати дії із заданого набору дій.

- Команду на виконання окремої дії звичайно називають оператором або інструкцією.

Приклади виконавців

пральна машина,

смартфон,

мультиварка,

комп'ютер

Приклади інструкцій:

- виконати прання бавовняної білизни,
- встановити з'єднання із заданим номером,
- приготувати плов

Компілятори та інтерпретатори

Компілятор – програма, що здійснює трансляцію програми, складеної у високорівневій мові програмування, в еквівалентну програму низькорівневою мовою, близькою до машинного коду.

Інтерпретатор – програма, що здійснює покомандний аналіз програми, її обробку та виконання.

Основні етапи побудови програми

Введення (набір) тексту

Перевірка синтаксичної правильності

[Компіляція]

Виконання

Мова Python

Створений на початку 1990-х років Гвідо ван Россумом.

- Назва походить від популярної гумористичної передачі Monty Python's Flying Circus

Основні версії Python

- Python 1.0 - січень 1994
- Python 2.0 - 16 жовтня 2000
- Python 3.0 - 3 грудня 2008
 - Python 3.4 - 16 березня 2014
 - Python 3.10.7 - 09 вересня 2022



Чому Python?

проста мова

потужна мова

широко розповсюджений

має інструменти для наукових розрахунків

універсальна мова

Де використовують

Компанія Google використовує Python у своїй пошуковій системі

Intel, Cisco, Hewlett-Packard, Seagate, Qualcomm і IBM, використовують Python для тестування апаратного забезпечення

Служба колективного використання відеоматеріалів YouTube в значній мірі реалізована на Python

NSA (АНБ США) використовує Python для шифрування і аналізу розвідданих

Компанії JPMorgan Chase, UBS, Getco і Citadel застосовують Python для прогнозування фінансового ринку

Популярна програма BitTorrent для обміну файлами в пірінгових мережах написана мовою Python

Популярний веб-фреймворк App Engine від компанії Google використовує Python як прикладну мову програмування

NASA, Los Alamos, JPL і Fermilab використовують Python для наукових обчислень.

Звідки завантажити

<https://www.python.org/downloads/> для Windows, Linux/Unix, Mac OS

- у багатьох інсталяціях Linux та Mac OS інтерпретатор Python вже включений і його не треба завантажувати. Щоб перевірити, чи встановлено Python, треба набрати команду `python -V`

<https://play.google.com/store/apps/details?id=org.qpython.qpy3&hl=en> для Android

<https://apps.apple.com/us/app/pythonista-3/id1085978097?ls=1>

або

<https://apps.apple.com/us/app/python3ide/id1357215444>

для IOS

Як запустити

Для Windows

- Запуск з командного рядка
 - Запуск виконується так: спочатку у пункті меню «Виконати» набираємо **cmd**. Відкривається вікно з системною підказкою. Після цього набираємо **python**. Отримуємо інформацію про встановлену версію інтрепретатора та підказку інтерпретатора `>>>`.
- Запуск середовища розробки IDLE з інтерфейсу Windows
 - «Пуск» → «Програми» → «Python x.y» → «IDLE(Python GUI)»,
 - де x.y = номер версії та підверсії, наприклад, 3.9

Як запустити.2

Для UNIX/Linux/Mac OS у вікні емулятора терміналу набираємо **python3**. Отримуємо інформацію про встановлену версію інтерпретатора та підказку інтерпретатора >>>

- У деяких сучасних версіях UNIX/Linux/Mac OS замість **python3** треба набрати **python**

Для Android/IOS – звичайний запуск програми натисненням на її піктограму, після чого вибрати «Console» для запуску інтерпретатора або «Editor» для запуску редактора.

Як запустити.3

Завершення роботи з інтерпретатором

- Для Windows
 - Якщо інтерпретатор запущено з командного рядка, то натиснути **<Ctrl+Z>** та **<Enter>**
 - Для середовища IDLE просто використовуємо інтерфейс для виходу.
- Для UNIX/Linux/Mac OS натиснути **<Ctrl+D>** або набрати **exit()**.
- Для Android/iOS – звичайне завершення програми або набрати **exit()**.

Занурюємося у Python...

Константи

Цілі числа будемо записувати в десятковій позиційній системі числення.

Дійсні - у вигляді десяткового неперіодичного дробу. Будемо відділяти дробову частину від цілої крапкою.

Приклади запису чисел:

- 0
- 1
- +31
- -176
- 3.14159

Рядки-константи

Рядки-константи беруть у апострофи

'це рядок'

- або у подвійні лапки

"це теж рядок"

- або у потрійні апострофи

'''це також

рядок'''

- або у потрійні подвійні лапки

''''навіть це також

рядок''''

Змінні

Будь-яка **змінна** має ім'я та значення.

- Ім'я – це ідентифікатор, а значення – константа.
- Значення змінної може бути визначено або не визначено.

Ідентифікатор - це слово, складене з літер і цифр, на першому місці якого обов'язково знаходиться літера.

Приклади ідентифікаторів:

- x
- pi
- a1
- max3
- _E2
- sigma

Основні команди Python

Присвоєння

Введення

Виведення

Тотожна команда

Арифметичний вираз. Множина операцій

Позначимо множину операцій

$$\Omega = \{+, -, *, /, //, \%, **\}$$

- + - додавання
- - віднімання
- * - множення
- / - ділення
- // - ділення націло
- % - остача від ділення
- ** - піднесення до степеня

Арифметичний вираз. Означення

Арифметичним виразом назвемо вираз e , який визначається індуктивно:

- 1. Якщо e - числова константа, то e - арифметичний вираз;
- 2. Якщо e - змінна, то e – арифметичний вираз;
- 3. Якщо e_1, e_2 - арифметичні вирази, $\omega \in \Omega$ - арифметична операція, то $e = e_1 \omega e_2$ - арифметичний вираз;
- 4. Якщо e_1 - арифметичний вираз, то $e = (e_1)$ - арифметичний вираз.

Арифметичний вираз. Приклади

Приклади арифметичних виразів:

$$1$$

$$1+2$$

$$x$$

$$(x+1)$$

$$(x+1)+z$$

$$1+2*x$$

$$2*(x+y)$$

Арифметичний вираз. Обчислення

Пріоритет операцій:

Операції
**
*, /, //, %
+, -

- Спочатку обчислюється результат операцій вищого пріоритету.
- Операції однакового пріоритету обчислюються зліва направо.
- Порядок обчислення може бути змінений дужками.

Наприклад $x/y*z$ буде обчислюватись як $(x/y)*z$.

Команда присвоєння

Синтаксис:

$x = e$

- де x – змінна, e – вираз.

Правило присвоєння:

Python обчислює значення виразу e у правій частині присвоєння та робить його значенням змінної x .

При цьому попереднє значення змінної x стає недосяжним.

Приклади присвоєнь

$$\mathbf{r = s}$$

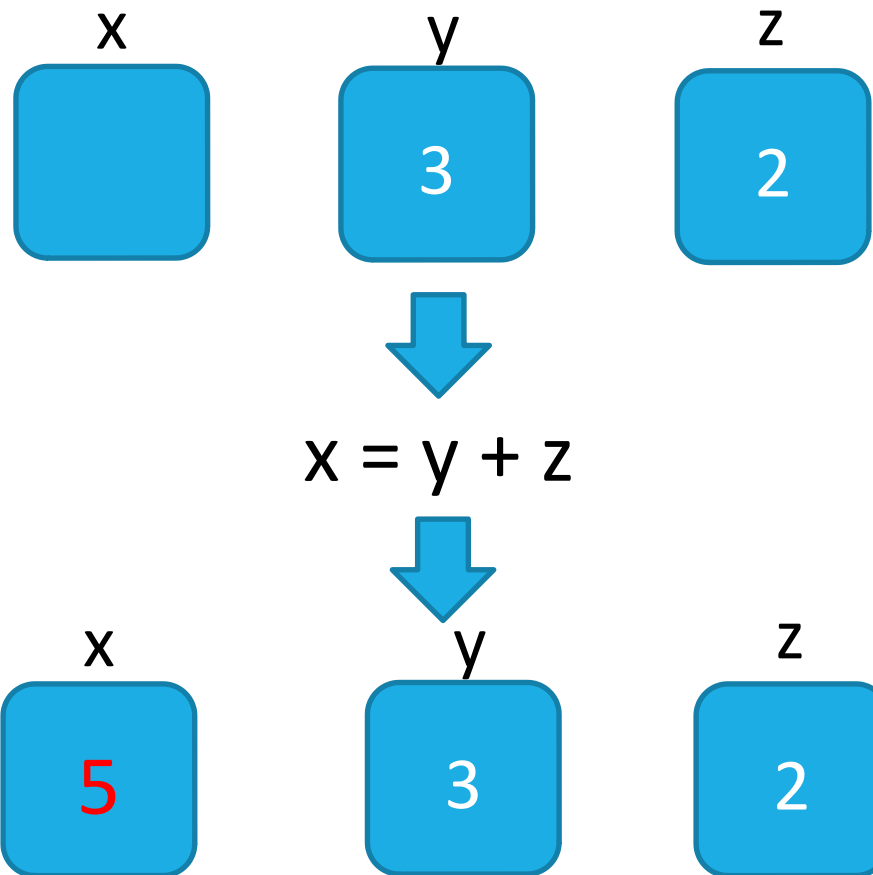
$$\mathbf{s = 1}$$

$$\mathbf{v = i * t}$$

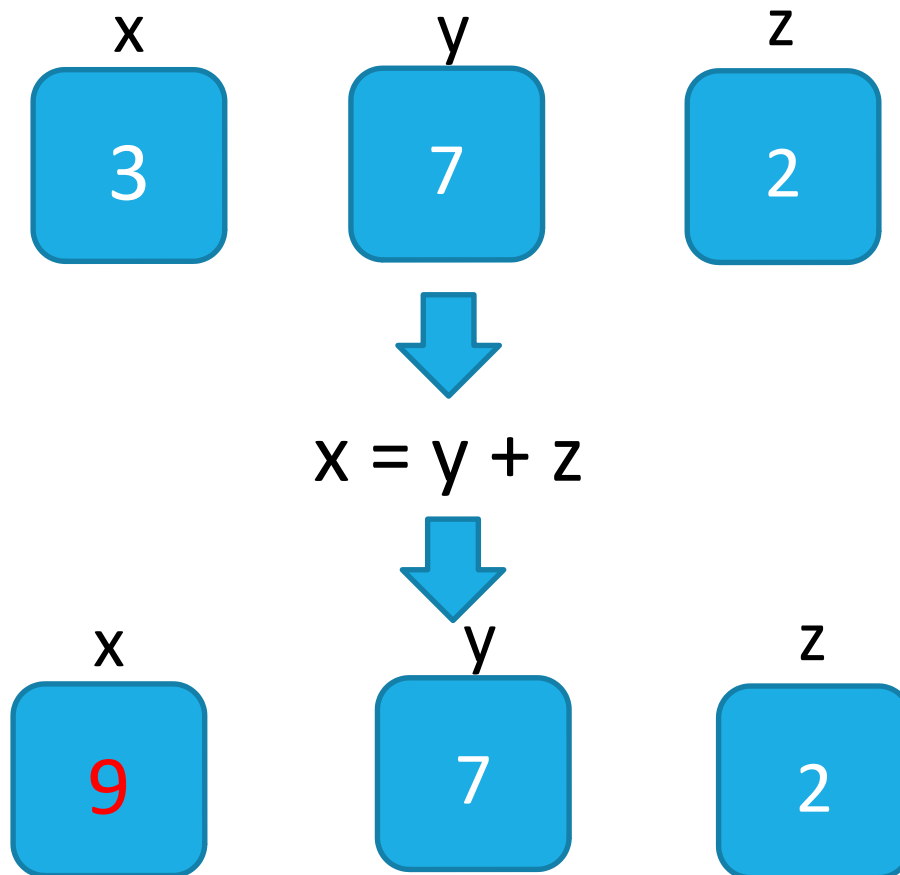
$$\mathbf{x = x + 1}$$

$$\mathbf{a2 = (b - c) / (c + d)}$$

Виконання присвоєнь



Виконання присвоєнь.2



Команда введення

Синтаксис:

x = input(S)

- де x – змінна, S – рядок підказки.

Для введення цілого числа, треба писати

x = int(input(S))

Для введення дійсного числа, треба писати

x = float(input(S))

Правило введення

Правило введення:

Python очікує введення значення змінної з клавіатури.

Після введення робить його значенням змінної x .

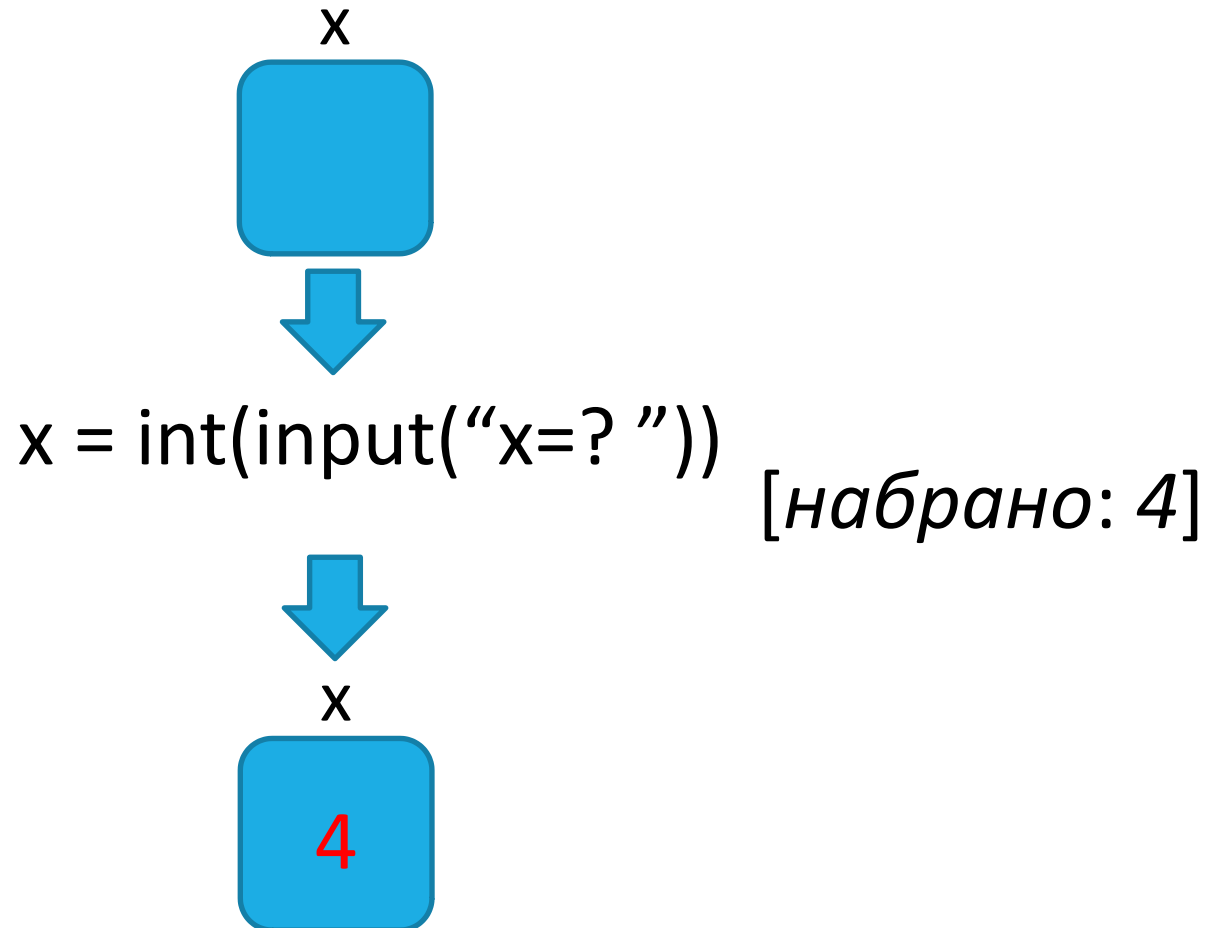
При цьому попереднє значення змінної x стає недосяжним.

Приклади введення:

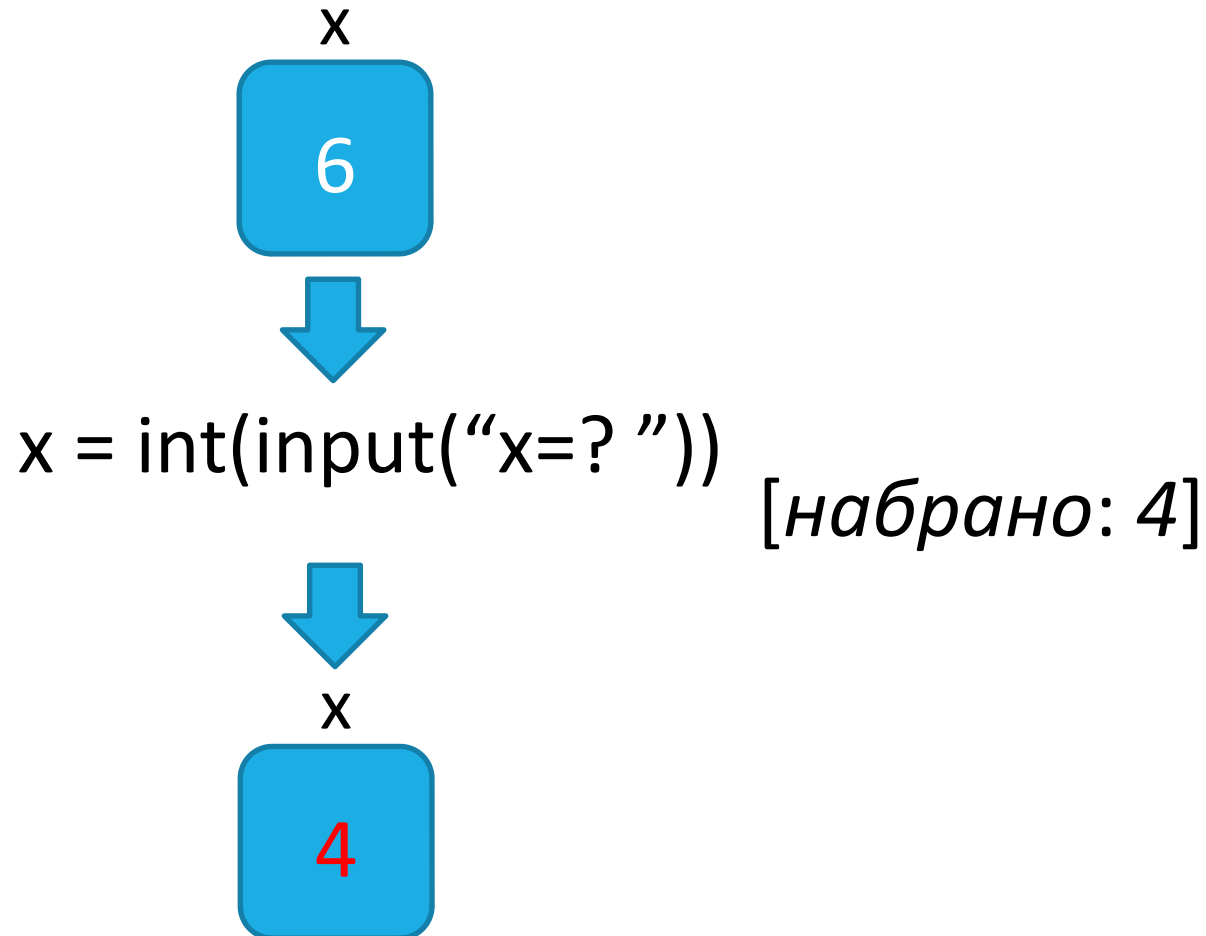
```
m = int(input('Введіть m: '))
```

```
y = float(input('y=? '))
```

Виконання введення



Виконання введення.2



Команда виведення

Синтаксис:

print(e_1 , ..., e_n)

- де e_i – вирази.

Правило виведення:

Python виводить на екран значення виразів e_i

Приклади виведення:

print('m=', m, 'y=', y)

Тотожна команда

Синтаксис:

pass

Правило виконання тотожної команди:

Python нічого не робить.

Спеціальні команди присвоєння

Доволі часто зустрічаються ситуації коли треба виконувати присвоєння такого вигляду:

$$\mathbf{x = x \omega e}$$

- де ω – деяка операція

У подібних випадках можна застосовувати скорочену форму команди присвоєння

$$\mathbf{x \omega = e}$$

Спеціальні команди присвоєння. 2

А саме:

$x += e$

$x -= e$

$x *= e$

$x /= e$

$x **= e$

$x //= e$

$x %= e$

Спеціальні команди присвоєння. 3

Наприклад

$$\mathbf{a += b}$$

замість

$$\mathbf{a = a + b}$$

Ланцюг команд. Означення

Визначимо **ланцюг команд** наступним чином:

- 1 Якщо P – команда присвоєння, введення, виведення або тотожна команда, то P – ланцюг.
- 2 Якщо P, Q – ланцюги, то
 - P
 - Q- ланцюг.

Ланцюг команд. Правило ланцюга

Правило ланцюга.

Python виконує команди з ланцюга послідовно.

Ланцюг команд. Приклади ланцюгів

$$\mathbf{v} = \mathbf{i} * \mathbf{t}$$

$$\mathbf{r} = \mathbf{s}$$

$$\mathbf{s} = \mathbf{1}$$

$$\mathbf{x} = \mathbf{x} + \mathbf{1}$$

$$\mathbf{x} = \mathbf{x} + \mathbf{1}$$

Рівносильність інструкцій

Дві інструкції P , Q будемо називати **рівносильними**

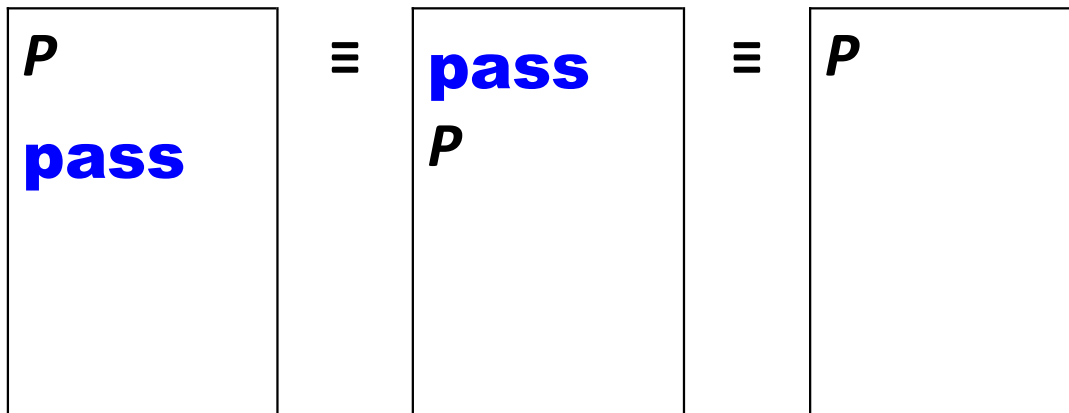
$$P \equiv Q$$

якщо вони дають однаковий результат.

- Результатом інструкції є зміна значення деякої змінної, або/та стану клавіатури (введення), або стану екрану (виведення). Тобто, інструкції можна розглядати як функції на станах виконавця Python

Властивість ланцюга

Властивість ланцюга:



Лінійні програми

Лінійною називається програма, яка є ланцюгом команд введення, виведення, присвоєння або тотожної команди.

Правила запису програм

У програмах у Python можна виділити фізичні та логічні рядки.

Фізичний рядок – це один рядок у інтерпретаторі або у файлі з текстом програми.

Логічний рядок – це один оператор програми з точки зору інтерпретатора.

Правила запису програм.2

Якщо треба продовжити логічний рядок програми на наступний фізичний рядок, у кінці першого рядка треба поставити “\”

Наприклад:

$$x = (a - b - c) * (c - b + a) * \backslash \\ (c - 2 * b) / (a * a + b * b + c * c)$$

Правила запису програм.3

Декілька фізичних рядків об'єднуються у 1 логічний рядок також коли відкриваюча дужка стоїть у першому фізичному рядку, а відповідна їй закриваюча, - у одному з наступних

Наприклад:

$$\mathbf{x = (a - b - c * b + a * b$$
$$\mathbf{+ b * b + c * c)}$$

Відступи

У Python відступи є важливими

Відступ у логічному рядку – це кількість пропусків перед першим символом, який не є пропуском.

Оператори, які утворюють ланцюг, повинні мати однаковий відступ

Коментарі

Коментарі починаються з # або беруться у ' ' ' (" " ") з обох боків

v = i * t # обчислення швидкості

або

**""" Це коментар,
який включає багато рядків
"""**

Приклад лінійної програми

Обчислення значення поліному

$$y = x^{**6} - 4 * x^{**4} + 3 * x - 7$$

Резюме

Ми розглянули:

1. Коротку історію розвитку програмування
2. Визначення інформації
3. Поняття алгоритму та виконавця
4. Програми та програмування, компілятори та інтерпретатори
5. Де завантажити та як запустити Python
6. Основні команди Python (присвоєння, введення, виведення, тотожню команду)
7. Ланцюги та лінійні програми
8. Рівносильність інструкцій
9. Правила запису програм у Python

Де прочитати

1. Обвінцев О.В. Інформатика та програмування. Курс на основі Python. Матеріали лекцій. – К., Основа, 2017
2. A Byte of Python (Russian) Версія 2.01 Swaroop С Н (Translated by Vladimir Smolyar),
<http://wombat.org.ua/AByteOfPython/AByteofPythonRussian-2.01.pdf>
3. Бублик В.В., Личман В.В., Обвінцев О.В.. Інформатика та програмування. Електронний конспект лекцій, 2003 р., <http://www.matfiz.univ.kiev.ua/books>
4. Марк Лутц, Изучаем Python, 4-е издание, 2010, Символ-Плюс
5. Самоучитель Python. <http://pythonworld.ru/samouchitel-python>
6. С. Шапошникова. Python. Введение в программирование <https://younglinux.info/python.php>
7. Python 3.7.2 documentation