**ОСНОВНІ ТЕОРЕТИЧНІ ВІДОМОСТІ**

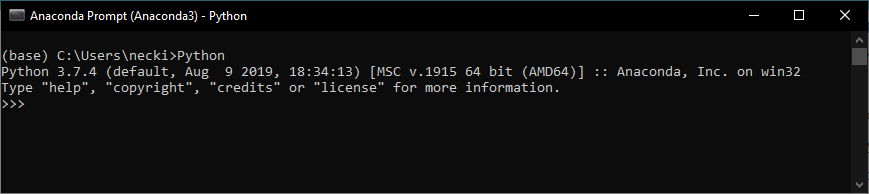
**Початок роботи з Python**

**Встановлення**

Python можна скачати з python.org. Однак якщо він ще не встановлений, то замість нього рекомендується встановити дистрибутивний пакет Anaconda, який вже включає в себе більшість бібліотек, необхідних для роботи в області науки про дані.

**Консоль Python**

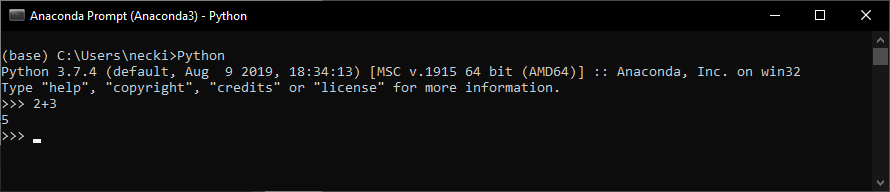
Щоб почати працювати з Python, потрібно відкрити командний рядок на комп'ютері. Щоб відкрити консоль Python, необхідно ввести python, для Windows, або python3 для Mac OS / Linux, і натиснути enter. Якщо ви встановили Anaconda, треба відкрити вікно Anaconda Prompt, та ввести Python у консолі що з’явилася.



Після запуску Python командний рядок змінилася на >>>. Це означає, що зараз ми можемо використовувати тільки команди на мові Python. Вводити >>> - не потрібно, Python буде робити це самостійно. Для виходу з консолі Python, в будь-який момент - необхідно ввести exit () або використовувати поєднання клавіш Ctrl + Z для Windows і Ctrl + D для Mac / Linux.

**Калькулятор в консолі**.

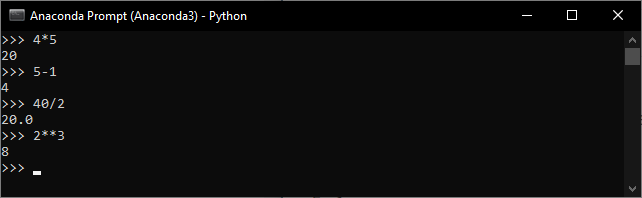
Спробуємо набрати простий математичний вираз, 2 + 3, і натиснути enter.



Як можна переконатися Python знає математику! Також можливе виконання і інших арифметичних дій, як:

4 \* 5; 5 – 1; 40/2;

Щоб обчислити ступінь числа, наприклад, 2 у кубі, ми вводимо: 2\*\*3.



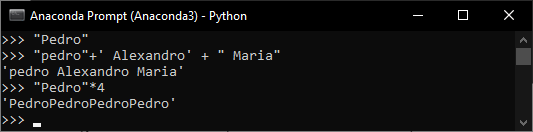
**Рядки**

Для прикладу роботи з рядками спробуємо ввести своє ім'я. Рядки необхідно вводити в лапках:

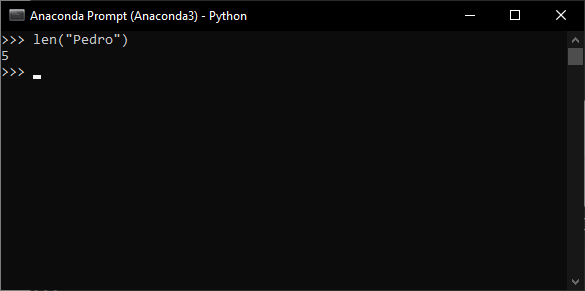


Рядок повинен завжди починатися і закінчуватися однаковими символами. Їми можуть бути одинарні (') або подвійні (") лапки.

Над рядками також можуть проводиться арифметичні операції:



Для обчислення кількості символів у рядку використовується метод len();



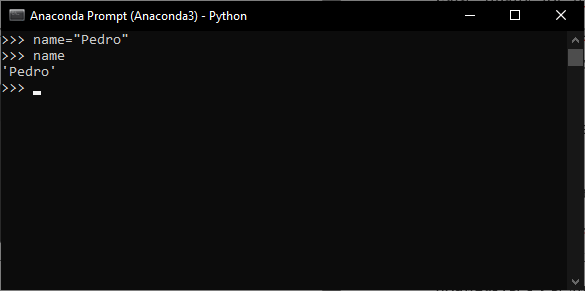
**Змінні і типи даних**

**Змінні**

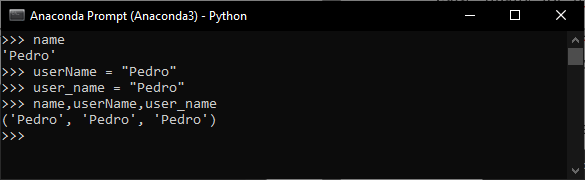
Змінна зберігає певні дані. Назва змінної в Python має починатися з алфавітного символу або зі знака підкреслення і може містити алфавітно-цифрові символи і знак підкреслення. Крім того, назва змінної не повинна збігатися з назвою ключових слів мови Python. Ключових слів не так багато, їх легко запам'ятати: and, as, assert, break, class, continue, def, del, elif, else, except, False, finally, for, from, global, if, import, in, is, lambda, None, nonlocal, not, or, pass, raise, return, True, try, while, with, yield.

Наприклад, створимо змінну:

Тут визначена змінна name, яка зберігає рядок "Pedro".



У Пайтон застосовується два типи найменування змінних: camel case і underscore notation. Camel case має на увазі, що кожне нове підсловом в найменуванні змінної починається з великої літери. Underscore notation має на увазі, що підслова в найменуванні змінної поділяються знаком підкреслення. наприклад:



І також треба враховувати що назви змінних чутливі до регістру, тому змінні name і Name представлятимуть різні об'єкти.

**Типи даних**

Змінна зберігає дані одного з типів даних. В Python існує безліч різних типів даних, які поділяються на категорії: числа, послідовності, словники, набори:

**boolean** - логічне значення True або False.

**int** - представляє ціле число, наприклад, 1, 4, 8, 50.

**float** - представляє число з плаваючою точкою, наприклад, 1.2 або 34.76

**complex** - комплексні числа

**str** - рядки, наприклад "hello". В Python 3.x рядки представляють набір символів в кодуванні Unicode

**bytes** - послідовність чисел в діапазоні 0-255

**byte array** - масив байтів, аналогічний bytes з тим відмінністю, що може змінюватися

**list** - список

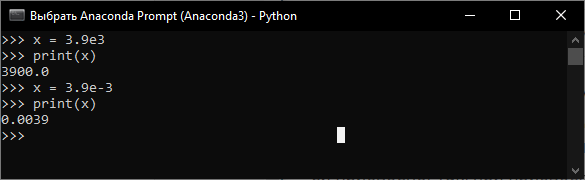
**tuple** - кортеж

**set** - невпорядкована колекція унікальних об'єктів

**frozen set** - те ж саме, що і set, тільки не може змінюватися (immutable)

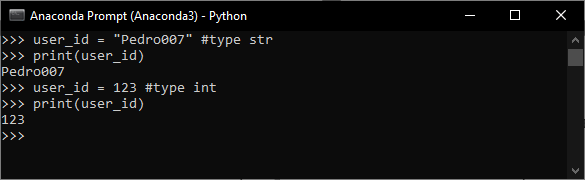
**dict** - словник, де кожен елемент має ключ і значення

Python є мовою з динамічною типізацією. Він визначає тип даних змінної виходячи із значення, яке їй присвоєно. Так, при присвоєнні рядки в подвійних або одинарних лапках змінна має тип str. При присвоєнні цілого числа Python автоматично визначає тип змінної як int. Щоб визначити змінну як об'єкт float, їй присвоюється дробове число, в якому роздільником цілої і дробової частини є точка. Число з плаваючою крапкою можна визначати в експоненційної запису:

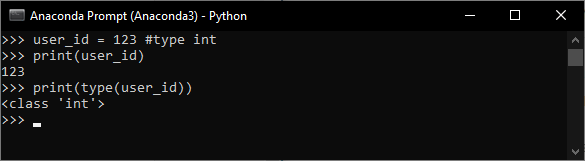


Число float може мати тільки 18 значущих симолів. Так, в даному випадку використовуються тільки два символи - 3.9. І якщо число занадто велике або занадто мало, то ми можемо записувати число у подібній нотації, використовуючи експоненту. Число після експоненти вказує ступінь числа 10, на яке треба помножити основне число - 3.9.

При цьому в процесі роботи програми ми можемо змінити тип змінної, присвоївши їй значення іншого типу:



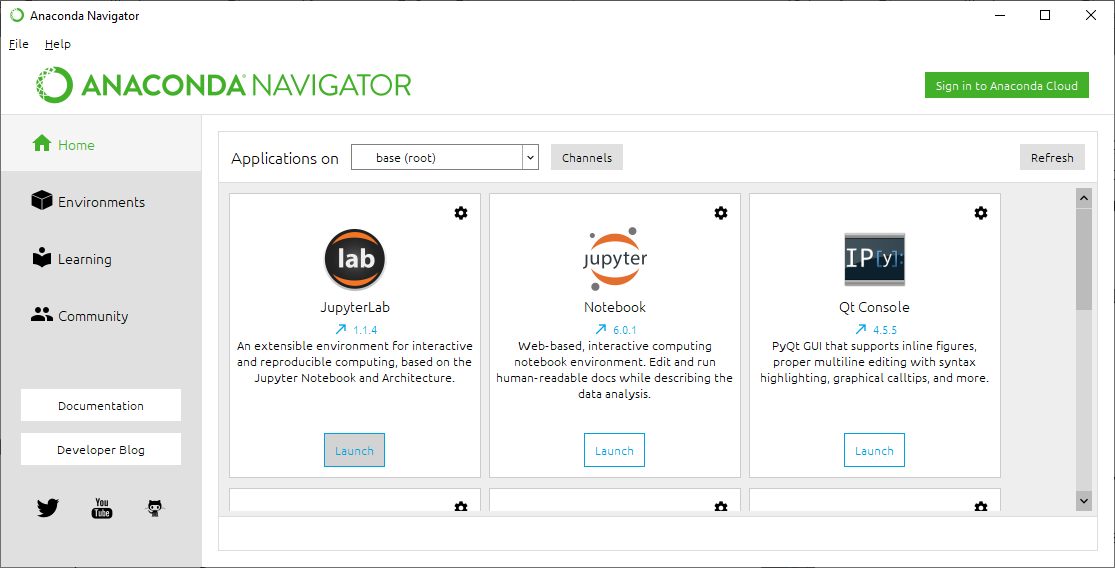
За допомогою функції type () можна динамічно дізнатися поточний тип змінної:



**Jupiter lab**

JupyterLab - це інтерактивне середовище розробки для роботи з блокнотами, кодом і даними. Проект Jupyter існує для розробки програмного забезпечення з відкритим вихідним кодом, відкритих стандартів і сервісів для інтерактивних та відтворюваних обчислень.

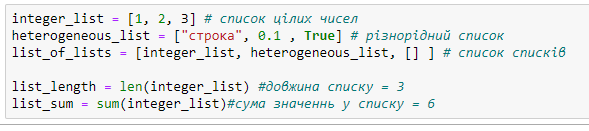
Використовувати JupyterLab ми будемо для виконання лабораторних, а саме написання сохраніеніе компіляції і запуску коду. Відкрити JupyterLab можна з пакету Anaconda.



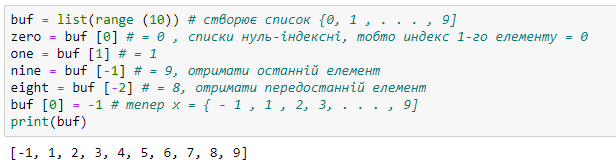
Більш детально ознайомитися з JupyterLab можна на сайті <https://proglib.io/p/jupyter>

**Списки**

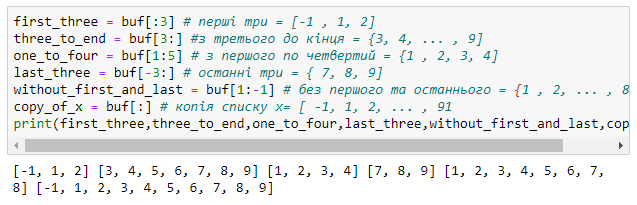
Однією з найважливіших структур даних в Python є список. Це просто впорядкована сукупність (або колекція), схожа на масив в інших мовах програмування, але з додатковими функціональними можливостями.



Встановлювати значення і отримувати доступ до n-го елемента списку можна за допомогою квадратних дужок:



Крім цього, квадратні дужки застосовуються для «нарізки» списків:

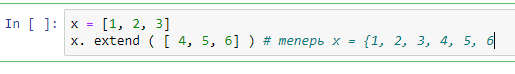


В Python є оператор ln, який перевіряє належність елемента списку:

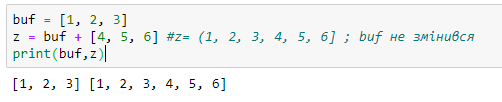


Перевірка полягає в почерговому перегляді всіх елементів, тому користуватися ним стоїть тільки тоді, коли точно відомо, що список невеликий або неважливо, скільки часу піде на перевірку.

Списки легко зчіплювати один з одним:



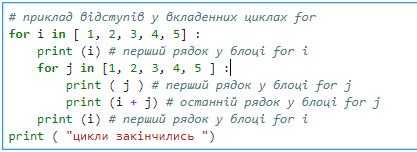
Якщо потрібно залишити список х без змін, то можна скористатися складанням списків:



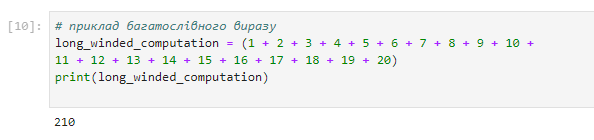
**Структура програми Python**

**Пробільні символи**

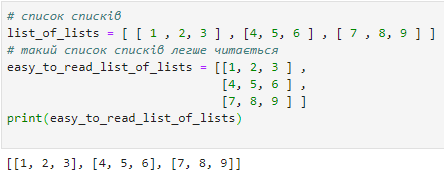
У багатьох мовах програмування для розмежування блоків коду використовуються фігурні дужки. В Python використовуються відступи:



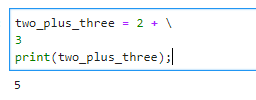
Це робить код легким для читання, але в той же час змушує стежити за форматуванням. Пропуск всередині круглих і квадратних дужок ігнорується, що полегшує написання багатослівних виразів:



і дозволяє легко читати код:



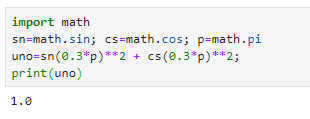
Для продовження оператора на наступному рядку використовується зворотна коса риса, втім, такий запис буде застосовуватися рідко:



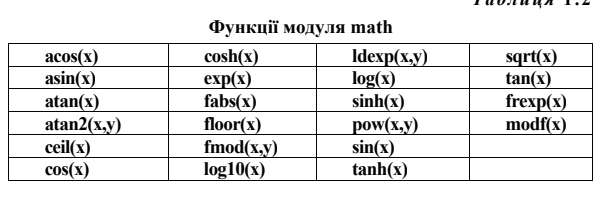
**Модулі (Імпорт бібліотек)**

Деякі бібліотеки середовища програмування на основі Python не завантажуються за замовчуванням. Для того щоб ці інструменти можна було використовувати, необхідно імпортувати модулі, які їх містять.

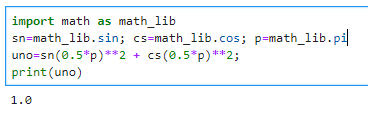
Один з підходів полягає в тому, щоб просто імпортувати сам модуль:



Тут math - це назва модуля, що містить функції і константи для 'роботи з регулярними виразами. Імпортувавши таким способом весь модуль, можна звертатися до функцій, випереджаючи їх префіксом math.

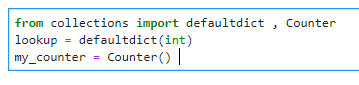


Якщо в коді змінна з ім'ям math вже є, то можна скористатися псевдонімом модуля:



Ім'я користувача використовують також в тих випадках, коли імпортований модуль має громіздке ім'я або коли в коді відбувається часте звертання до модуля.

Якщо з модуля потрібно отримати кілька конкретних значень, то їх можна імпортувати в явному вигляді і використовувати без обмежень:



**Керуючі конструкції**

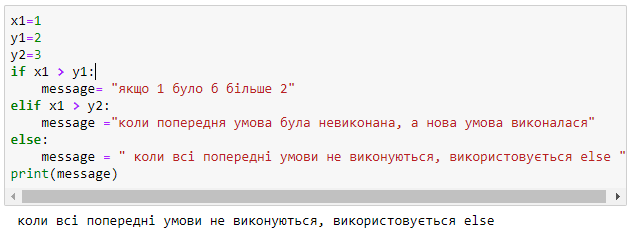
**Оператори розгалуження (if, else, elif)**

Як і в більшості інших мов програмування, дії можна виконувати за умовою, застосовуючи оператори розгалуження, такі як:

if – якщо + умова,

elif – інакше якщо + умова,

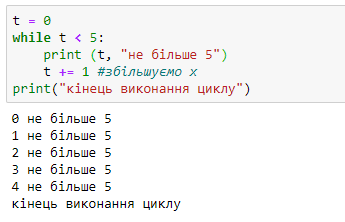
else – інакше.



**Цикли (while, for)**

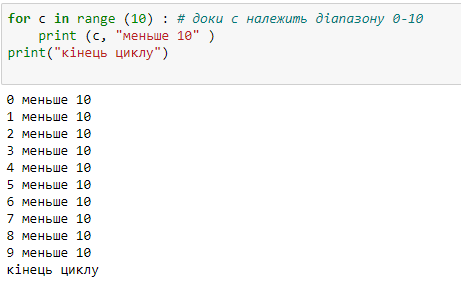
В Python є цикл while, який працює як і в інших язиках програмування. **while** <умова>: - тіло циклу

Тобто всі операції записані в тілі циклу будуть повторно виконуватися до тих пір, доки умова циклу не перестане бути істинною.

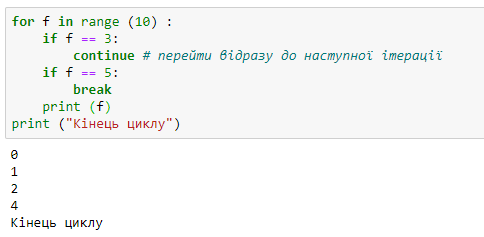


Однак частіше буде використовуватися цикл for спільно з оператором in: **for** <змінна> **in** <діапазон>: блок

Блок коду після заголовка виконується Доті, поки змінна належить діапазону (причому цим діапазоном може бути список, числова послідовність, рядок, інша послідовність якихось проіндексованих значень):



Якщо потрібно більш складна логіка управління циклом, то можна скористатися операторами continue та break:



**Істинність**

Булеві змінні в Python працюють так само, як і в більшості інших мов програмування лише з одним винятком - вони пишуться з великої літери:



В Python може використовуватися будь-яке значення там, де очікується логічний тип Boolean. Всі наступні елементи мають логічне значення False:

* False; .
* None;
* set() (множина):
* [] (пустий список);
* {} (пустий словник);