Знайомство з мовою Python

У всьому є своя мораль,
треба лише вміти її знайти!

Л. Керролл. "Аліса в Країні Див"[*]

У цій книзі йтиметься про те, як писати програми на мові програмування, яка називається Python (правильно читається як пайтон, але зазвичай назву мови читають як пітон, що теж цілком припустимо). Отже, розв'язувати будемо два завдання, одне з яких пріоритетне, а друге, хоч і допоміжне, проте досить важливе. Наше основне завдання — звичайно ж, вивчення синтаксису мови програмування Python. Паралельно ми будемо освоювати ази програмування, явно чи неявно беручи до уваги, що відповідні алгоритми передбачається реалізовувати мовою Python.

Навіть якщо у читача є досвід програмування іншими мовами, не варто ставитися поверхнево чи зверхньо до процесу побудови алгоритму програми. Правила хорошого тону в програмуванні передбачають, що написання програми починається задовго до набирання програмного коду. Непогано взяти аркуш паперу й накреслити загальну схему виконання програми. А для цього процедуру розв'язання великої та складної задачі варто розбити на послідовність простих дій. З одного боку, цей процес універсальний. З іншого — ті задачі, які ми назвали вище "простими", розв'язуються за допомогою базових команд або функцій мови програмування, якою збираються складати програму. Тому, обмірковуючи алгоритм, цілком абстрагуватися від конкретних можливостей мови програмування навряд чи вийде. Ураховуючи ж гнучкість й ефективність мови програмування Python, слід визнати, що алгоритми навіть "класичних" задач, реалізуючись на Python, стають простішими й зрозумілішими. Іншими словами, навіть якщо читач має досвід

[*] Тут і далі епіграфи подаються у перекладі автора.

складання алгоритмів, знайомство з мовою програмування Python дозволить йому побачити багато знайомих речей зовсім по-іншому.

Узагалі, мов програмування доволі багато. Більше того, час від часу з'являються нові. Тому природно виникає запитання: чому саме Python? Наша відповідь буде складатися з декількох пунктів.

- Мова програмування Python — мова високого рівня, досить "молода", проте дуже популярна, яка вже зараз широко використовується на практиці, і сфера застосування Python постійно розширюється.

Щодо мов програмування нерідко застосовують такі вирази, як мова ***високого рівня***, мова ***середнього рівня*** або мова ***низького рівня***. Ця класифікація досить умовна й базується на рівні абстракції мови. Адже мова, якою розмовляють люди, дещо відрізняється від тієї "мови", яку "розуміють" комп'ютери. Команда, написана простою людською мовою, буде зовсім неприйнятною для комп'ютера. Команда, готова до виконання комп'ютером (***машинний код***), буде незрозумілою для більшості простих смертних. Тому вибирати доводиться між Сциллою та Харибдою. Про мови, орієнтовані на програміста, говорять, що ці мови високого рівня. Про мови, орієнтовані на комп'ютер, говорять, що ці мови низького рівня. Проміжна група мов називається мовами середнього рівня. Хоча ще раз підкреслимо, що поділ цей досить умовний.

- Синтаксис мови Python мінімалістичний і гнучкий. Цією мовою можна складати прості й ефективні програми.
- Стандартна бібліотека для цієї мови містить багато корисних функцій, що значно полегшує процес створення програмних кодів.
- Мова Python підтримує декілька парадигм програмування, включаючи ***структурне***, ***об'єктно-орієнтоване*** й ***функціональне*** програмування. І це далеко не весь список.
- Мова Python — цілком вдалий вибір, як для першої мови в навчанні програмуванню.

Існують й інші причини, щоб вивчити мову програмування Python, можливо, навіть вагоміші за перераховані вище. Про деякі ми ще будемо говорити (у контексті особливостей мови програмування Python). У всякому разі, тут будемо виходити з того, що читач для себе ухвалив

рішення про вивчення мови Python або, принаймні, цікавиться цією мовою програмування.

Парадигма програмування — це найзагальніша концепція, яка визначає фундаментальні характеристики й базові методи реалізації програмних кодів. Наприклад, парадигма ***об'єктно-орієнтованого*** програмування (скорочено ООП) передбачає, що програму реалізують через набір взаємодіючих об'єктів, які, у свою чергу, зазвичай створюються на основі класів. У рамках ***структурного*** програмування програма є комбінацією даних і процедур (функцій) для їх обробки. Мова може підтримувати одразу декілька парадигм. Так, мови Java і C# — повністю об'єктно-орієнтовані, тому для написання найменшої програми цими мовами доведеться описати, як мінімум, один клас. У мові C підтримується парадигма структурного програмування, тому класів й об'єктів у мові C немає. Зате вони є в мові C++. Остання підтримує як парадигму об'єктно-орієнтованого програмування, так і парадигму структурного програмування. Як наслідок, під час роботи з мовою C++ класи й об'єкти можна використовувати, а можна й не використовувати залежно від потреб програміста й специфіки задачі, яку розв'язують. Це ж зауваження стосується й мови Python: із одного боку, під час написання програми мовою Python у нас є можливість удатися до потужного арсеналу об'єктно-орієнтованого програмування, а з іншого боку, часто бувають прийнятними й методи структурного програмування.

Існують й інші, більш витончені концепції програмування. Скажімо, парадигма функціонального програмування припускає, що результат функції в програмі визначається виключно значеннями аргументів, переданих функції, і не залежить від стану зовнішніх (стосовно функції) змінних. Відповідні функції прийнято називати ***чистими функціями***, і вони мають ряд корисних властивостей, що дозволяють істотно оптимізувати й прискорити обчислювальний процес. Ця концепція, як і ряд інших, знаходить реалізацію в мові Python.

Далі обговоримо деякі важливі моменти й "підводне каміння", яке може зустрітися на подекуди важкому, та все ж цікавому й захопливому шляху опанування новими вершинами у програмуванні.

Коротка історія та особливості мови Python

> Серйозне ставлення до будь-чого
> в цьому світі є фатальною
> помилкою.
>
> *Л. Керролл. "Аліса в Країні Див"*

У мови Python є автор ***Гвідо ван Россум*** (Guido van Rossum). І хоча в розробці й популяризації мови на теперішній момент устигло взяти участь багато талановитих розробників, саме Гвідо ван Россум отримав заслужену славу творця цієї перспективної й популярної мови програмування. Взагалі ж робота над мовою почалася у 80-х роках минулого століття. Вважають, що перша версія мови з'явилася в 1991 році. Щодо назви мови програмування Python, то, формально, це назва рептилії. Відповідно, часто як логотип використовують милу (або не дуже) змійку типу "пітон". І хоча практично будь-який навчальний чи довідковий посібник із мови Python містить розповідь про те, що насправді Python це не "пітон", а назва гумористичної передачі "Літаючий цирк Монті Пайтона", для історії це вже не важливо.

Доречно згадати слова капітана Врунгеля з однойменної повісті Андрія Некрасова: "Як ви яхту назвете, так вона й попливе". У мови програмування Python досить агресивна назва, і, треба визнати, ця назва себе виправдовує. За аргумент до такого твердження може правити як гнучкість й ефективність самої мови, так і та швидкість, із якою вона завоювала собі "місце під сонцем" серед найпопулярніших мов програмування.

Мова Python бурхливо розвивається. Цьому сприяє не тільки досить вдала концепція мови, а також згуртоване співтовариство, сформоване

з розробників і популяризаторів мови. Важливий і той факт, що необхідне програмне забезпечення, включаючи середовища розробки, в основному безкоштовне. Усе це дає підстави розглядати Python як одну з найперспективніших мов програмування.

На сьогодні Python використовується при реалізації найрізноманітніших проектів, серед яких:

- розробка сценаріїв для роботи з Web та Internet-програмами;
- мережеве програмування;
- засоби підтримки технологій HTML і XML;
- програми для роботи з електронною поштою й підтримки Internet-протоколів;
- програми для обслуговування найрізноманітніших баз даних;
- програми для наукових розрахунків;
- програми з графічним інтерфейсом;
- створення ігор і комп'ютерної графіки та багато іншого.

Зрозуміло, охопити всі ці теми в одній книзі досить складно. Та ми й не ставимо це собі за мету. А втім, навіть на відносно невеликій кількості нескладних прикладів цілком можливо продемонструвати елегантність і виключну ефективність мови Python. Цим, власне, ми й займемося в основній частині книги — тобто трохи згодом. Зараз обговоримо особливості та деякі «технічні» моменти, які важливі для розуміння основ програмування мовою Python.

Python належить до мов програмування, ***що інтерпретуються***, і це має певні наслідки. Формально те, що мова програмування належить до інтерпретованих, означає, що програмний код виконується за допомогою спеціальної програми-***інтерпретатора***. Інтерпретатор виконує програмний код порядково (з попереднім аналізом виконуваного коду). Недолік такого підходу полягає в тому, що, по-перше, помилки виявляються фактично на етапі виконання програми і, по-друге, швидкість виконання програми відносно невисока. Тому нерідко застосовується більш складна схема: вихідний програмний код компілюється в проміжний код, а вже цей проміжний код виконується безпосередньо інтерпретатором. У цьому разі швидкість виконання програми збільшується, але разом із нею збільшується і потреба у системних ресурсах. Приблизно за такою схемою виконується програмний код, написаний мовою Python.

Нагадаємо, що окрім мов, які інтерпретуються, існують мови, програми на яких ***компілюються*** (мається на увазі компіляція в машинний код). У цьому випадку вихідний програмний код компілюється у виконавчий (машинний) код, який виконується (зазвичай) під керуванням операційної системи. Компільовані у виконавчий код програми характеризуються відносно високою швидкістю виконання.

Якщо ми хочемо написати програму на мові Python, то для цього, як мінімум, знадобиться набрати відповідний програмний код. Тут можливі варіанти, але, в принципі, код набирається в будь-якому текстовому редакторі, а відповідний файл зберігається з розширенням `py` чи `pyw` (для програм із графічним інтерфейсом). Під час першого запуску (після внесення змін у програмний код) створюється проміжний код, який записується у файл із розширенням `pyc`. Якщо після цього в програму зміни не вносилися, то під час виконання програми буде виконуватися відповідний `pyc`-файл. Після внесення змін у програму під час чергового запуску вона перекомпілюється в `pyc`-файл. Це загальна схема. Нас, насправді, вона цікавить виключно в плані загального розвитку, хоча на практиці іноді буває важливо враховувати особливості виконання програми, написаної мовою Python.

Як зазначалося вище, програмний код можна набирати й у текстовому редакторі. Та ось для виконання такого програмного коду знадобиться спеціальна програма, яка називається ***інтерпретатором***. Іншими словами, для роботи з Python на комп'ютер необхідно встановити програму-інтерпретатор. Ми окремо зупинимося на цьому питанні. Зараз же тільки зауважимо, що зазвичай використовується не просто інтерпретатор, а ***інтегроване середовище розробки***, яке, крім іншого, включає в себе як інтерпретатор, так і редактор програмних кодів.

Інтерпретатор виконує програму команда за командою. Тому, в принципі, якщо програма складається з декількох команд, її можна організувати у вигляді файлу з програмним кодом, а потім «відправити» цей файл на виконання. Ще один варіант — «передавати» інтерпретатору для виконання по одній команді. Обидва режими можливі й підтримуються інтерпретатором мови Python. Ми будемо складати і запам'ятовувати програмні коди у файлах — тобто використаємо «традиційний» підхід.

Про режим, за якого в командному вікні інтерпретатора команди вводяться і виконуються одна за одною, говорять, як про ***режим командного рядка,*** або ***режим калькулятора***.

Оскільки мова Python розвивається інтенсивно, й від версії до версії в синтаксис і структуру мови вносяться зміни, важливо враховувати, для якої версії мови Python складають (і передбачають використовувати) програмний код. Особливо це важливо, враховуючи ту обставину, що під час внесення змін ***принцип зворотної сумісності*** спрацьовує далеко не завжди: якщо програмний код коректно виконується в давніших версіях мови, то зовсім не факт, що він буде виконуватися під час роботи з новішими версіями. Однак панікувати з цього приводу не варто. Зазвичай проблема несумісності кодів для різних версій пов'язана з особливостями синтаксису деяких функцій або конструкцій мови і досить легко усувається.

На момент написання книги актуальною є версія Python 3.6. Саме вона використовувалася для тестування прикладів у книзі. У разі виходу нових версій або стандартів мови, для забезпечення сумісності програмних кодів є сенс проглянути той розділ довідкової системи, в якому описано нововведення. Зробити це можна, наприклад, на офіційному сайті підтримки мови Python `www.python.org/doc/` у розділі під назвою ***What's New In Python*** (у перекладі означає ***Що нового в Python***).

Але все це технічні деталі, які хоч і важливі, та все ж не першорядні. А першорядними для нас будуть синтаксис мови Python і його основні інструкції керування. І, власне, тут на нас чекає чимало приємних сюрпризів.

Особливо багато сюрпризів буде для читачів, які знайомі з такими мовами програмування, як Java чи C++, наприклад. Але це, до речі, зовсім не означає, що новачків у програмуванні мова Python залишить байдужими. Просто ті, хто вивчав основи ООП і програмує згаданими мовами, значно розширять свій кругозір щодо методів і прийомів програмування, а також, у певному сенсі, їм доведеться змінити своє уявлення про мови програмування.

Синтаксис мови Python більш ніж цікавий. По-перше, він простий, зрозумілий і наочний. Його навіть можна назвати по-спартанськи лаконічним.

Разом з цим програмні коди, написані на Python, зазвичай легко читаються й аналізуються, а обсяг програмного коду набагато менший, порівняно з аналогічними програмами, написаними іншими мовами програмування. Як ілюстрацію в лістингу В.1 наведено код програми мовою C++, у результаті виконання якого в консольне вікно виводиться повідомлення `Hello, world!`.

Лістинг В.1. Програма мовою C++

```
#include <iostream>
using namespace std;
int main(){
    cout<<"Hello, world!"<<endl;
    return 0;
}
```

Аналогічний програмний код, але вже мовою Java, представлено в лістингу В.2.

Лістинг В.2. Програма мовою Java

```
class MyClass{
    public static void main(String[] args){
        System.out.println("Hello, world!");
    }
}
```

Нарешті, у лістингу В.3 показано, як виглядатиме програма для виведення в консольне вікно текстового повідомлення, якщо для її написання скористатися мовою програмування Python.

Лістинг В.3. Програма мовою Python

```
print("Hello, world!")
```

Неважко помітити, що це всього одна команда, де вбудованій функції `print()` як аргумент передається текст, який необхідно надрукувати в консольному вікні. Зрозуміло, далеко не завжди у нас буде виходити писати такі «економні» коди, але приклад усе ж багато в чому показовий.

Про всяк випадок, коротко прокоментуємо наведені вище коди мовами C++ і Java — просто, щоб у читача, не знайомого з цими мовами, не виникло комплексу меншовартості. Почнімо з програми лістингу В.1, написаної мовою C++:

- Інструкція `#include <iostream>` підключає заголовковий файл бібліотеки вводу/виводу.
- Команда `using namespace std` означає використання стандартного простору імен.
- Функція з назвою `main()` називається головною функцією програми. Виконання програми в C++ — це виконання головної функції програми.
- Ідентифікатор `int` ліворуч від функції `main()` означає, що функція повертає цілочисловий результат.
- Пара фігурних дужок (`{` і `}`) виділяє тіло головної функції.
- Команда `cout<<"Hello, world!"<<endl` виводить у консоль текстове повідомлення `Hello, world!`, і відбувається перехід до нового рядка (інструкція `endl`). Оператор виводу `<<` виводить текст, зазначений праворуч від нього, на пристрій, визначений ідентифікатором `cout` (за замовчуванням — консоль).
- Інструкція `return 0` завершує виконання головної функції (тобто програми), а результатом функція повертає `0` (означає закінчення роботи програми «у штатному режимі» — тобто без помилок).

Програма в лістингу В.2, нагадаємо, написана мовою Java, і у відповідному програмному коді призначення інструкцій таке:

- Створюється клас із назвою `MyClass`: перед назвою класу зазначається ключове слово `class`, а тіло класу береться у фігурні дужки (зовнішня пара дужок `{` і `}`).
- У тілі класу описується головний метод із назвою `main()`, тіло методу береться в блок із фігурних дужок (внутрішня пара дужок `{` і `}`).
- Перед назвою головного методу зазначено такі ідентифікатори: `public` (відкритий метод — тобто доступний поза класом), `static` (статичний метод — для виклику методу немає необхідності створювати об'єкт класу), `void` (метод не повертає результат).
- Як параметр (аргумент) методу `main()` указано змінну `args`, яка є текстовим масивом (текст — це об'єкт класу `String`, а наявність порожніх квадратних дужок `[]` свідчить про те, що це текстовий масив — упорядкований набір текстових значень).
- Текст у консольне вікно виводиться за допомогою методу `println()`: як аргумент методу передається текст, що виводиться в консоль, а сам метод викликається з об'єкта потоку виводу `out`, який, у свою чергу, є статичним полем класу `System`.

Зрозуміло, на фоні всього цього різноманіття програма (а точніше, одна єдина команда) мовою Python виглядає більш ніж ефектно. Але ще раз підкреслюємо, що, навіть якщо читач зрозумів не все з викладеного вище (щодо кодів C++ і Java), це не страшно. Нам, у нашій подальшій роботі, усе це не знадобиться. Ми просто хотіли проілюструвати масштаби, так би мовити, розбіжностей для різних мов.

Під час роботи з мовою Python ми не зустрінемо багатьох звичних для інших мов конструкцій. Наприклад, на відміну від мов C++ і Java, в яких командні блоки виділяються фігурними дужками `{` і `}`, і на відміну від мови Pascal, у якій блоки виділяються за допомогою інструкцій `begin` і `end`, у мові Python блок команд виділяється відступом (рекомендований відступ — чотири пробіли). У мові Python немає необхідності закінчувати кожну команду крапкою з комою. Існують й інші особливості мови Python. Ми будемо знайомитися з ними поступово і, перефразовуючи М.Є. Салтикова-Щедріна, російського письменника-сатирика, намагатимемося при цьому не застосовувати силу.

Мається на увазі цитата «Просвіту впроваджувати помірковано, по можливості уникаючи кровопролиття» із сатиричного роману «Історія одного міста» М.Є. Салтикова-Щедріна.

Тут просто важливо зрозуміти, що мова Python досить своєрідна, самобутня й у багатьох відношеннях не схожа на інші популярні сьогодні мови програмування.