

# Python Starter

Змінні та типи даних

# Python Starter

## План заняття

1. Змінні та константи
2. Арифметичні операції
3. Скорочена форма запису арифметичних операцій під час роботи зі змінними
4. Операції порівняння
5. Логічні операції
6. Форматування та виведення рядків на екран
7. Основні методи при роботі з рядками

# Python Starter

Після уроку обов'язково



Повторіть цей урок у відео форматі на [ITVDN.com](http://itvdn.com)

Доступ можна отримати через керівництво вашого навчального центру



Перевірте, як Ви засвоїли цей матеріал на [TestProvider.com](http://testprovider.com)

# Python Starter

Тема

Змінні та типи даних

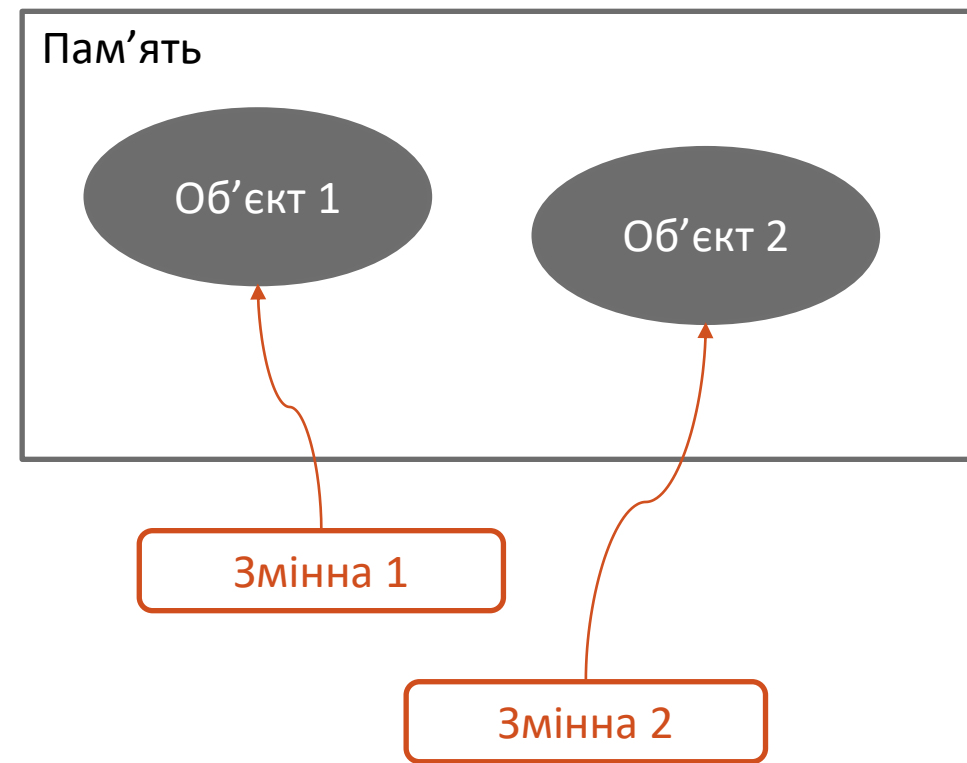
# Python Starter

## Змінні та константи

**Ідентифікатор** – це ім'я об'єкту. Він може складатися з великих і маленьких літер, цифр, знаків підкреслення, не повинен починатися з цифри і не може співпадати із зарезервованими ключовими словами мови. Реєстр має значення.

**Змінна** у Python – посилання на який-небудь об'єкт. Відповідно до PEP 8, імена змінних повинні записуватися маленькими літерами через знаки підкреслення: `my_variable`.

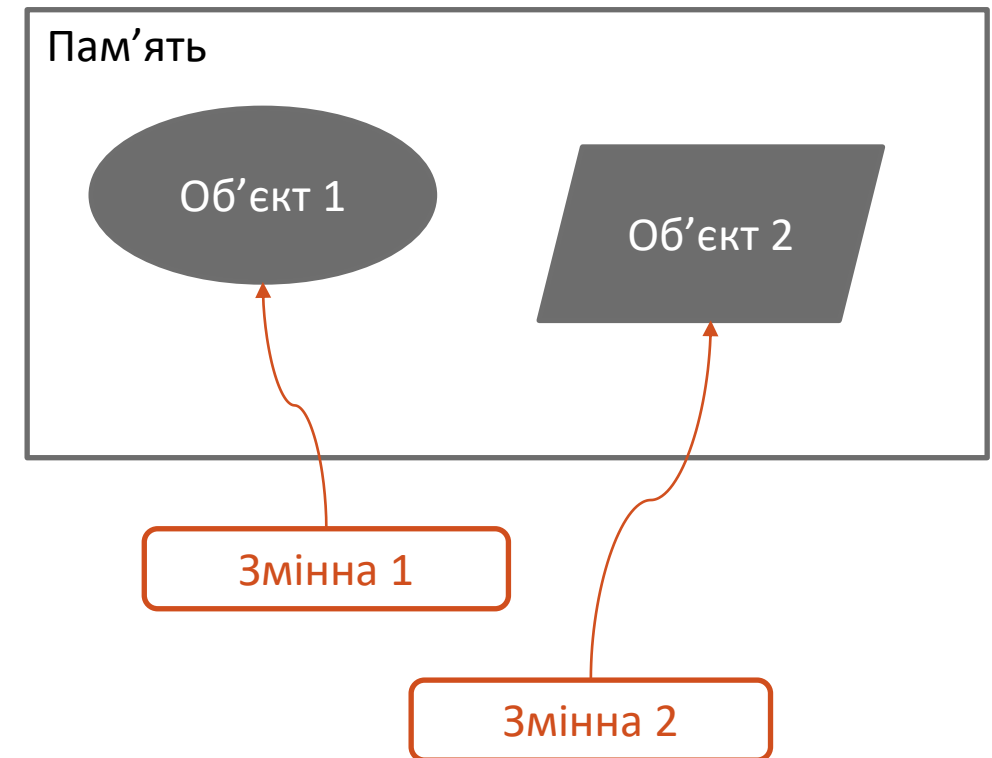
**Константа** – значення, яке не призначено для зміни. У Python немає окремої синтаксичної конструкції для оголошення констант. Прийнято називати їх ідентифікаторами, написаними великими літерами: `MY_CONSTANT`.



# Python Starter

## Поняття типів даних

- *Тип даних* (тип) — множина значень та операцій на цих значеннях.
- Тип визначає можливі значення та їх сенс, операції, і навіть способи зберігання значень типу. Вивчається теорією типів. Невід'ємною частиною більшості мов програмування є системи типів, що використовують типи для забезпечення того чи іншого ступеня типобезпеки.
- Операція призначення типу інформаційним сутностям називається *типізацією*.



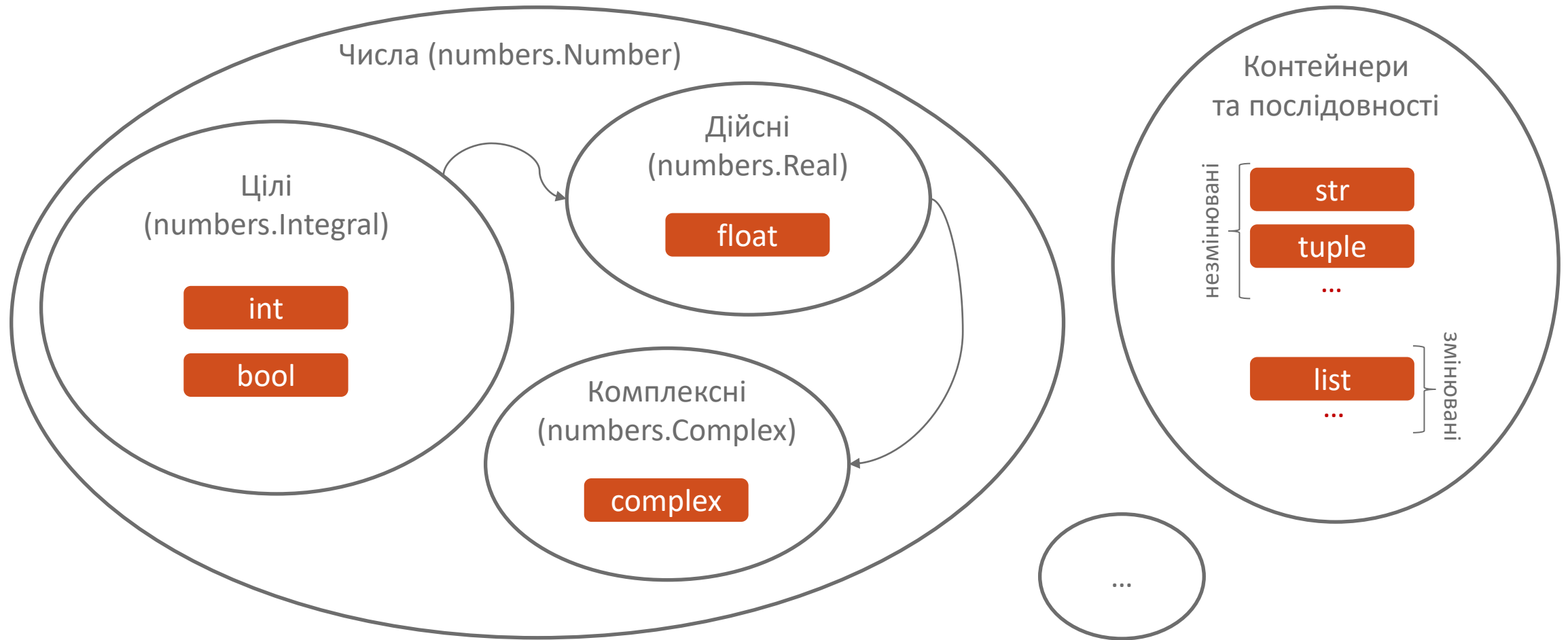
# Python Starter

## Типізація

Типізація	Статична	Динамічна
Сильна (строга)	C#, Java, Haskell, Scala	<b>Python</b> , Ruby
Слабка	C	JavaScript, PHP

# Python Starter

## Основні стандартні типи даних





# Python Starter

Мова програмування Python

Операції з числами

# Python Starter

## Операції з числами

$x + y$	сума
$x - y$	різниця
$x * y$	добуток
$x / y$	частка
$x // y$	операція цілочисельного ділення
$x \% y$	остача від ділення
$-x$	число, протилежне до $x$
$+x$	$x$

# Python Starter

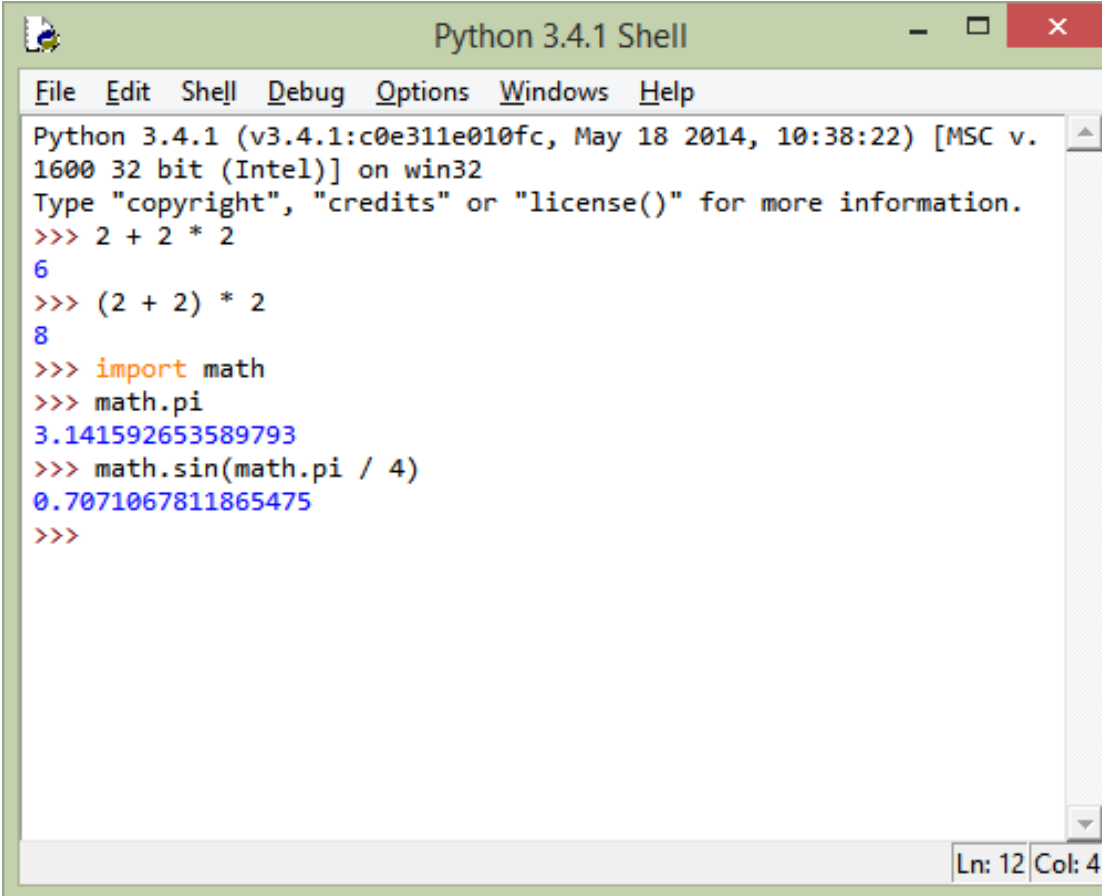
## Операції з числами

<b>abs(x)</b>	модуль числа x
<b>int(x)</b>	перетворити x на ціле число
<b>float(x)</b>	перетворити x на дійсне число
<b>complex(re, im)</b>	створити комплексне число $re + im*i$
<b>c.conjugate()</b>	число, спряжене комплексному числу c
<b>x ** y</b> <b>pow(x, y)</b>	x в степені y
<b>round(x)</b> <b>round(x, n)</b>	округлити дійсне число x (до n цифр після коми, якщо n вказано)

# Python Starter

## Операції з числами

- Арифметичні операції мають пріоритети (такі ж, як і в математиці). Щоб змінити порядок обчислень, слід використовувати круглі дужки.
- Можна імпортувати модуль `math`, написавши `import math`, та використовувати його функції та константи.



```
Python 3.4.1 Shell
File Edit Shell Debug Options Windows Help
Python 3.4.1 (v3.4.1:c0e311e010fc, May 18 2014, 10:38:22) [MSC v.
1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 2 + 2 * 2
6
>>> (2 + 2) * 2
8
>>> import math
>>> math.pi
3.141592653589793
>>> math.sin(math.pi / 4)
0.7071067811865475
>>>
```

Ln: 12 Col: 4

# Python Starter

## Скорочена форма запису арифметичних операцій під час роботи зі змінними

Для прискорення процесу розроблення під час роботи зі змінними використовують скорочену форму запису :

- `my_variable = my_variable + 4`      еквівалентно :      `my_variable += 4`
- `my_variable = my_variable - 4`      еквівалентно :      `my_variable -= 4`
- `my_variable = my_variable * 4`      еквівалентно :      `my_variable *= 4`
- `my_variable = my_variable / 4`      еквівалентно :      `my_variable /= 4`

# Python Starter

## Деякі функції та константи модуля math

<code>math.trunc(x)</code>	відкинути дробову частину дійсного числа <code>x</code> , повертає ціле число
<code>math.floor(x)</code>	найбільше ціле число (ціле в математичному сенсі, а не як тип даних), яке не перевищує дане дійсне число
<code>math.ceil(x)</code>	найменше ціле число, більше або рівне даному дійсному
<code>math.pi</code> , <code>math.e</code>	константи $\pi$ , $e$
<code>math.sin</code> , <code>math.cos</code> тощо	математичні функції; повний їх список можна переглянути в документації ( <a href="https://docs.python.org/3/library/math.html">https://docs.python.org/3/library/math.html</a> ) або набравши в консолі інтерпретатора: <pre>import math dir(math)</pre>

# Python Starter

Введення в Python

Логічні операції та порівняння

# Python Starter

## Логічні операції

- *Логічна операція* — операція над виразами логічного (булевого) типу, що відповідає деякій операції над висловлюваннями в алгебрі логіки. Як і висловлювання, логічні вирази можуть набувати одне з двох істиннісних значень — "істинно" або "хибно".
- Логічні операції служать для одержання складних логічних виразів із більш простих. У свою чергу, логічні вирази зазвичай використовуються як умови для керування послідовністю виконання програми.



# Python Starter

## Таблиці істинності

x	y	x and y
False	False	False
False	True	False
True	False	False
True	True	True

x	y	x or y
False	False	False
False	True	True
True	False	True
True	True	True

x	not x
False	True
True	False

# Python Starter

## Логічні операції у Python

<b>x or y</b>	якщо x – хибне, то y, інакше x
<b>x and y</b>	якщо x – хибне, то x, інакше y
<b>not x</b>	якщо x – хибне, то True, інакше False

# Python Starter

## Операції порівняння

<code>x &lt; y</code>	x строго менше y
<code>x &gt; y</code>	x строго більше y
<code>x &lt;= y</code>	x менше або дорівнює y
<code>x &gt;= y</code>	x більше або дорівнює y
<code>x == y</code>	x дорівнює y
<code>x != y</code>	x не дорівнює y
<code>x is y</code>	x та y – це один и той самий об'єкт
<code>x is not y</code>	x та y не є одним і тим самим об'єктом в пам'яті

З операціями `<`, `>`, `<=`, `>=` можна використовувати подвійні порівняння, наприклад `-2 <= x < 3`.

# Python Starter

Введення в Python

Робота з рядками

# Python Starter

## Рядки

- *Рядки* – текстові дані типу `str`.
- Рядкові літерали (значення, що задаються в коді) обрамляються з обох боків одинарними або подвійними лапками.
- Декілька рядкових літералів поспіль розпізнаються як один рядок.
- Рядки, що виділяються однією парою лапок, повинні розташовуватися на одному рядку коду. Якщо потрібно створити рядок з кількох рядків тексту, можна:
  - використовувати спеціальну послідовність символів `\n` ;
  - розмістити кілька рядкових літералів на різних рядках коду, не додаючи між ними жодних операцій;
  - використовувати спеціальний вид рядкових літералів, який обрамляється трьома парами одинарних чи подвійних лапок. Будь-яка частина вихідного тексту програми, що знаходиться між ними, включаючи переведення рядків та відступи, вважається рядком.

# Python Starter

## Деякі операції з рядками

<code>s1 + s2</code>	конкатенація (об'єднання) рядків
<code>s % x</code> <code>s % (x<sub>1</sub>, x<sub>2</sub>, ..., x<sub>n</sub>)</code>	форматування рядку в стилі C s – форматний рядок, x <sub>1</sub> ...x <sub>n</sub> – значення <a href="https://docs.python.org/3/library/stdtypes.html#printf-style-string-formatting">https://docs.python.org/3/library/stdtypes.html#printf-style-string-formatting</a>
<code>s.format(args)</code>	форматування рядку в стилі C# <a href="https://docs.python.org/3/library/stdtypes.html#str.format">https://docs.python.org/3/library/stdtypes.html#str.format</a>
<code>s[i]</code> (повернутися у сьомому уроці)	символ, що стоїть у рядку s на позиції i (нумерація починається з нуля)

# Python Starter

## Виведення

Для виведення значень на екран служить функція print:

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

де

- objects – це об'єкти, які потрібно вивести
- sep – дільник
- end – рядок, який потрібно вивести після всіх об'єктів
- file – файл, до якого необхідно вивести дані
- flush — чи потрібно відразу після виведення скинути вміст буфера у файл. Якщо ви виводите інформацію на одному рядку через тривалі проміжки часу, і вона не з'являється на екрані, доки ви не виведете символ нового рядку, додайте параметр flush=True

Жоден із цих параметрів не є обов'язковим.

# Python Starter

## Введення

- Для введення даних із клавіатури можна використовувати функцію `input`:

`input(prompt)`

`input()`

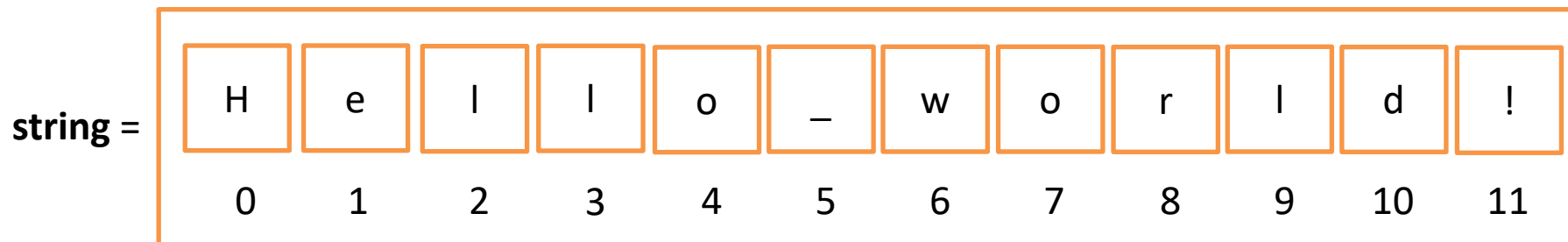
- Якщо параметр `prompt` заданий, він виводиться як пояснючий текст, запрошення до введення.
- Функція призупиняє виконання програми, доки користувач не введе рядок тексту, зчитує його та повертає.
- Зверніть увагу, що вона повертає саме рядок, тобто значення типу `str`, тому якщо необхідно ввести число, потрібно скористатися однією з функцій, які були розглянуті вище, щоб сконструювати число з його текстового представлення.



# Python Starter

## Поняття рядку

- **Рядки** – незмінні послідовності кодів символів (у Python 3 – у кодуванні Unicode, у Python 2 – в ASCII). Представлені класом **str**. У Python 2 також є клас **unicode**, який представляє Unicode-рядки подібно до **str** в Python 3.
- Рядкові літерали виділяються одинарними або подвійними лапками. Можна використовувати потроєні лапки для створення багаторядкових рядків. Якщо перед рядковим літералом стоїть префікс **r**, більшість escape-послідовностей ігноруються. У Python 2 префікс **u** задає Unicode-літерал.
- Підтримують всі спільні для послідовностей операції, і навіть реалізують безліч власних методів.
- Функція **ord(char)** повертає код символу **char**, а функція **chr(code)** повертає символ із кодом **code**.



У Python рядок є об'єктом, що складається з декількох елементів-символів.

# Python Starter

## Робота з рядками

Функція або метод	Призначення
<code>string = 'str'; string = "str"; string = '''str'''; string = """str"""</code>	Літерали рядків
<code>string = "s\np\ta\nbbb"</code>	Екрановані послідовності
<code>string = r"C:\temp\new"</code>	Неформатовані рядки (пригнічують екранування)
<code>string = b"byte"</code>	Рядок байтів
<code>string.find(str, [start],[end])</code>	Пошук підрядка у рядку. Повертає номер першого входження або -1
<code>string.rfind(str, [start],[end])</code>	Пошук підрядка у рядку. Повертає номер останнього входження або -1
<code>string.index(str, [start],[end])</code>	Пошук підрядка у рядку. Повертає номер першого входження або викликає ValueError
<code>string.rindex(str, [start],[end])</code>	Пошук підрядка у рядку. Повертає номер останнього входження або викликає ValueError
<code>string.replace(шаблон, заміна[, maxcount])</code>	Заміна шаблону на заміну. maxcount обмежує кількість замін

# Python Starter

## Робота з рядками

Функція або метод	Призначення
<code>string.split(символ)</code>	Розбиття рядка по роздільнику
<code>string.isdigit()</code>	Чи складається рядок із цифр
<code>string.isalpha()</code>	Чи складається рядок із літер
<code>string.isalnum()</code>	Чи складається рядок із цифр або літер
<code>string.islower()</code>	Чи складається рядок із символів у нижньому регістрі
<code>string.isupper()</code>	Чи складається рядок із символів у верхньому регістрі
<code>string.isspace()</code>	Чи складається рядок із невідображуваних символів (пробіл, символ переведення сторінки ('\f'), "новий рядок" ('\n'), "переведення каретки" ('\r'), "горизонтальна табуляція" ('\t' ) та "вертикальна табуляція" ('\v'))
<code>string.startswith(str)</code>	Чи починається рядок <b>string</b> із шаблону str
<code>string.endswith(str)</code>	Чи закінчується рядок <b>string</b> шаблоном str
<code>ord(символ)</code>	Символ в його код ASCII
<code>chr(число)</code>	Код ASCII в символ

# Python Starter

## Робота з рядками

Функція або метод	Призначення
<b>string.capitalize()</b>	Переводить перший символ рядка у верхній регістр, а решту у нижній
<b>string.center(width, [fill])</b>	Повертає відцентрований рядок, по краях якого стоїть символ fill (пробіл за замовчуванням)
<b>string.expandtabs([tabsize])</b>	Повертає копію рядка, де всі символи табуляції замінюються одним або декількома пробілами, залежно від поточного стовпця. Якщо TabSize не вказано, розмір табуляції дорівнює 8 пробілам
<b>string.lstrip([chars])</b>	Видалення пробільних символів на початку рядка
<b>string.rstrip([chars])</b>	Видалення пробільних символів в кінці рядка
<b>string.strip([chars])</b>	Видалення пробільних символів на початку і в кінці рядка
<b>string.partition(шаблон)</b>	Повертає кортеж, який містить частину перед першим шаблоном, сам шаблон і частину після шаблону. Якщо шаблон не знайдено, повертається кортеж, що містить сам рядок, а потім два порожні рядки
<b>string.rpartition(sep)</b>	Повертає кортеж, що містить частину перед останнім шаблоном, сам шаблон і частину після шаблону. Якщо шаблон не знайдено, повертається кортеж, що містить два порожні рядки, а потім самий рядок

# Python Starter

## Робота з рядками

Функція або метод	Призначення
<code>string.swapcase()</code>	Переводить символи нижнього регістру до верхнього, а верхнього – до нижнього
<code>string.title()</code>	Першу літеру кожного слова переводить у верхній регістр, а решту в нижній
<code>string.zfill(width)</code>	Робить довжину рядка не меншу за width, за необхідності заповнюючи перші символи нулями
<code>string.ljust(width, fillchar=" ")</code>	Робить довжину рядка не меншу за width, за необхідності заповнюючи останні символи символом fillchar
<code>string.rjust(width, fillchar=" ")</code>	Робить довжину рядка не меншу за width, за необхідності заповнюючи перші символи символом fillchar
<code>string.isdecimal()</code>	Повертає <i>True</i> , якщо рядок не порожній і складається лише з десяткових цифр.
<code>string.isidentifier()</code>	Повертає <i>True</i> , якщо рядок є <u>допустимим ідентифікатором мови Python</u>
<code>string.isnumeric()</code>	Повертає <i>True</i> , якщо рядок не порожній і складається тільки з десяткових цифр та символів, які так само відносяться до цифр.
<code>string.isprintable()</code>	Повертає <i>True</i> якщо всі символи в рядку є друкованими або якщо рядок є порожнім.

# Python Starter

## Робота з рядками

Функція або метод	Призначення
<code>string.isspace()</code>	Повертає True, якщо рядок не порожній і складається лише з пробільних символів, інакше повертається False.
<code>string.istitle()</code>	Повертає True, якщо рядок не порожній і є рядком заголовків
<code>string.partition(sep)</code>	Розбиває рядок за вказаним роздільником і повертає кортеж із трьох елементів: рядок до роздільника, сам роздільник та рядок після роздільника.
<code>string.splitlines([keepends])</code>	Розбиває рядок на підрядки по фіксованому набору роздільників і повертає їх у списку. На відміну від методу <code>split()</code> , в якому можна вказати довільний роздільник, у методі <code>splitlines</code> набір роздільників є фіксованим: <code>\n</code> - перенесення рядка; <code>\r</code> — повернення каретки; <code>\r\n</code> — повернення каретки та перенесення рядка; <code>\v</code> or <code>\x0b</code> - вертикальний відступ; <code>\f</code> or <code>\x0c</code> — розрив сторінки; <code>\x1c</code> - роздільник файлів; <code>\x1d</code> - роздільник груп; <code>\x1e</code> - роздільник рядків; <code>\x85</code> — наступний рядок; <code>\u2028</code> — роздільник рядків; <code>\u2029</code> — роздільник абзаців.
<code>string.translate()</code>	Перетворює символи вихідного рядку відповідно до правил перетворення в об'єкті <code>table</code> . Найпростіше цей метод використовувати у зв'язці з методом <code>str.maketrans()</code> , що дозволяє швидко задати таблицю перетворень..
<code>string.zfill(width)</code>	Повертає новий рядок, в якому вихідний рядок <code>str</code> доповнений нулями зліва так, щоб довжина нового рядка стала б рівною <code>width</code> .

# Python Starter

## «Сирі" рядки

"Сирі" рядки – пригнічують екранування.

Якщо перед відкриваючою лапкою стоїть символ 'r' (у будь-якому регістрі), то механізм екранування відключається.

```
S = r'C:\newt.txt'
```

Але, попри призначення, "сирий" рядок не може закінчуватися символом зворотного слешу. Шляхи вирішення:

```
S = r'\n\n\'[:-1]
```

```
S = r'\n\n' + '\\'
```

```
S = '\\n\\n'
```

# Python Starter

## Байти (bytes)

- **Байтові рядки в Python** — дуже схожі на звичайні рядки, але з деякими відмінностями.
- **Байт** — мінімальна одиниця зберігання та оброблення цифрової інформації. Послідовність байт являє собою будь-яку інформацію (текст, картинку, мелодію...).
- Функція `bytes` приймає список чисел від 0 до 255 і повертає байти, що отримуються за рахунок застосування функції `chr`.
- Тип `bytes` у Python може бути аналогічний масиву типу `uint8` у C, який по суті являє собою 8-бітні двійкові числа, розташовані послідовно. Наприклад, під час читання з файлу двійковим способом повертається тип `bytes` або рядок з префіксом `b` також є типом `bytes`. Такі як:
  - `a = b'abcd'`
  - `print(type(a)) # <class 'bytes'>`
- Байти аналізуються як числа
- В основному він ділиться на цифрові типи, такі як `UINT8`, `UINT16`, `UINT32`, `UINT64` і т. д., які відповідають кожному 1, 2, 4 і 8 байтам, з'єднаним разом і інтерпретованим як число, яке ділиться більш ніж на один байт.
- Для синтаксичного аналізу рекомендується використовувати вбудовану бібліотеку структур, метод є більш загальним.
- `struct.unpack(fmt, byte)`, де `fmt` - це форматований рядок, який розділений на дві частини, початок якого визначає розмір і кінець, а останній контролює тип числа через символи, які зазвичай використовуються наступним чином. Докладніше про використання `FMThelp(struct)`.



# Python Starter

## Байти (bytearray)

- **Bytearray** в Python - масив байт. Від типу `bytes` відрізняється лише тим, що є змінюваним. Тип даних `bytearray` є різновидом типу `bytes` і підтримує ті ж самі методи та операції. На відміну від типу `bytes`, тип `bytearray` допускає можливість безпосередньої зміни об'єкту і містить додаткові методи, що дозволяють виконувати ці зміни.
- Створити об'єкт типу `bytearray` можна наступними способами:
- За допомогою функції `bytearray([<Рядок>, <Кодування>[, <Оброблення помилок>]])`. Якщо параметри не вказані, повертається порожній об'єкт. Щоб перетворити рядок на об'єкт типу `bytearray`, необхідно передати щонайменше два перші параметри. Якщо рядок вказано лише у першому параметрі, то настає виняток `TypeError`.
- `struct.unpack(fmt, byte)`, де `fmt` - це форматований рядок, який розділений на дві частини, початок якого визначає розмір і кінець, а останній контролює тип числа через символи, які зазвичай використовуються наступним чином. Докладніше про використання `FMThelp(struct)`.

# Python Starter

## Різниця між типом байтів, кодом ASCII та типом str

Тип `bytes` дуже легко сплутати з кодом ASCII та типом `str`.

**Байти** – це просто масив 8-бітних чисел у якості одиничного елементу, а ASCII - це метод декодування для аналізу такого масиву чисел, аналогічно є utf-8 і т. д.

`str` – це тип об'єкту, а не концепція рядку в C, і його не можна напямую перетворити в число.

### Анализ типу байтів

Аналіз типу байтів можна поділити на два типи: один – аналізувати як числовий тип, а інший – як текст.

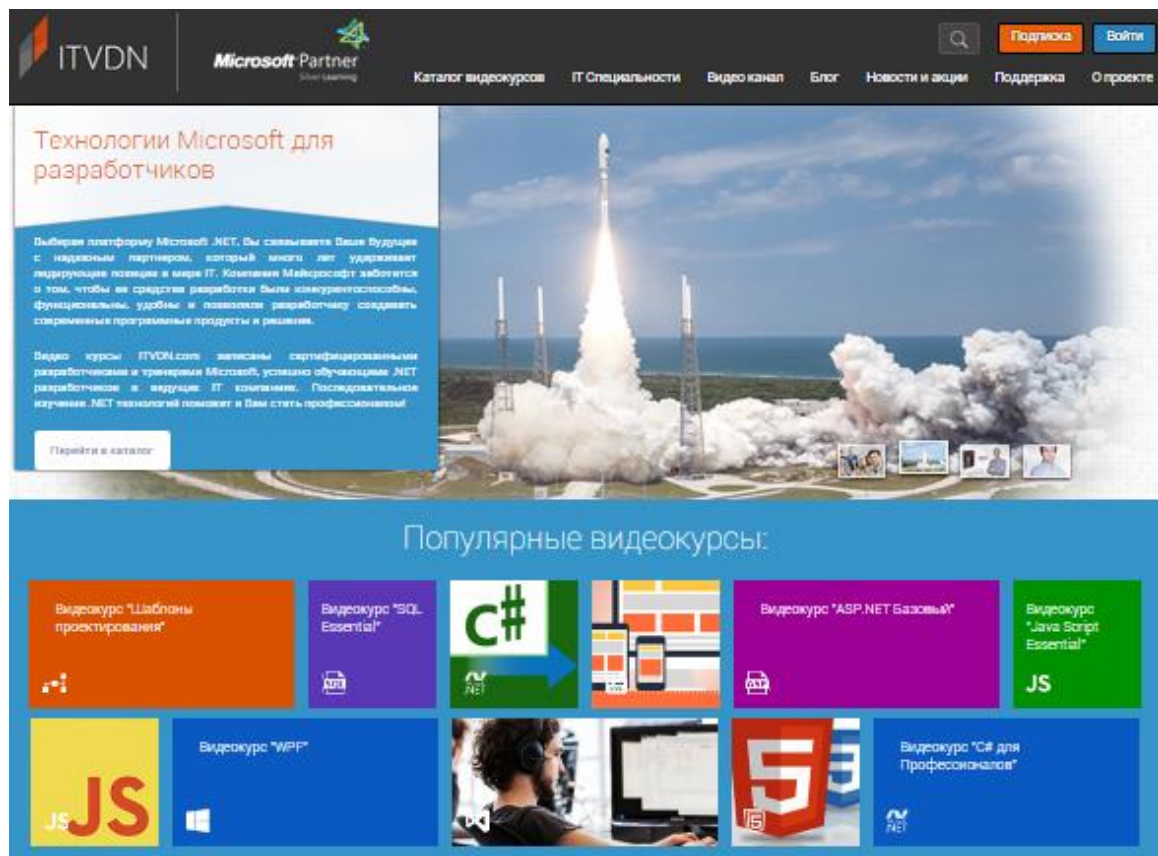
Байти аналізуються як числа

В основному він ділиться на цифрові типи, такі як `UINT8`, `UINT16`, `UINT32`, `UINT64` і т. д., які відповідають кожному 1, 2, 4 і 8 байтам, з'єднаним разом і інтерпретованим як число, яке ділиться більш ніж на один байт. Оброблення з прямим порядком байтів

Для синтаксичного аналізу рекомендується використовувати вбудовану бібліотеку структур, метод є більш загальним.

# Дивіться наші уроки у відео форматі

## ITVDN.com



Перегляньте цей урок у відео форматі на освітньому порталі [ITVDN.com](http://itvdn.com) для закріплення пройденого матеріалу.

Усі курси записані сертифікованими тренерами, які працюють у навчальному центрі CyberBionic Systematics



# Перевірка знань

TestProvider.com

TestProvider

Мы помогаем людям оценить себя

Регистрация Войти

Поиск сертификата

Главная Услуги и цены Центр Тестирования Поддержка О нас

Мы в социальных сетях

## Тестирование

Языки программирования и информационные технологии

**Microsoft**

C# ASP.NET MVC JavaScript Patterns OF Design SQL Architecture Guide WCF HTML&CSS XML SEO WPF HTML5&CSS3 JQuery XNA SharePoint GUI for Android Windows Azure Platform Microsoft Patterns&Practices TFS SCRUM ReSharper TDD WWF LINQ Entity Framework Windows Forms Refactoring Microsoft Expression Blend 4 Windows Phone 8 Windows 8 AppStore Visual Studio Tips&Tricks MSF MEF SilverLight AJAX MEF Service Oriented Architecture

Добро пожаловать на TestProvider.com!

Сайт перенесен на новую облачную платформу с использованием системы единой авторизации Single Sign On. Если вы хотите восстановить статистику по предыдущим экзаменам обратитесь в [службу поддержки](#). Для восстановления информации с предыдущей версии сайта, просба написать в службу поддержки Ваш старый и новый логины.

ITVDN PROMETRIC TEST CENTER CyberBionic Microsoft Partner Windows Azure Cloud Partner EBA

TestProvider – це online сервіс перевірки знань з інформаційних технологій. За його допомогою Ви можете оцінити Ваш рівень та виявити слабкі місця. Він буде корисним як у процесі вивчення технології, так і для загальної оцінки знань IT спеціаліста.

Після кожного уроку проходите тестування для перевірки знань на [TestProvider.com](http://TestProvider.com)

Успішне проходження фінального тестування дозволить Вам отримати відповідний Сертифікат.



# Python Starter

Q&A

# Python Starter

Дякую за увагу!

# Інформаційний відеосервіс для розробників програмного забезпечення

