



Become QA Auto

Типи даних

Бутенко Сергій



```
le.py > ...
x = "PiMyLifeUp" #First assign the 'x' variable a string
type(x) #Output the data type of our 'x' variable
x = 43 #Now assign it a integer value
type(x) #Output its type again

days = 5
print("There are " + str(days) + " days remaining")

a, b, c = "Apple", "Banana", "Cherry"
b = c = "Lime"

print("Hello World")
x = 453
int(x)
int(y)

print("Life Up")
```



План лекції



Тип даних як характеристика змінних



Основні типи даних



Катастрофічні помилки, пов'язані з типами даних



Тип даних як характеристика змінних



Тип даних - це характеристика змінної, яка визначає:

- яких значень може набувати змінна;
- операції, які можна робити над цими даними;
- який обсяг пам'яті виділяється для збереження цієї змінної.

◎ Основні типи даних та їх допустимі значення



Цілі – числа без десяткової крапки.

Приклад: -5, -21, 100500, 0, 259

Дійсні - числа, які мають десяткову крапку в своєму записі. Приклад: 9.8, 3.14, -99.99, 5.0, 0.0

Символьний тип - будь-які символи або текст.

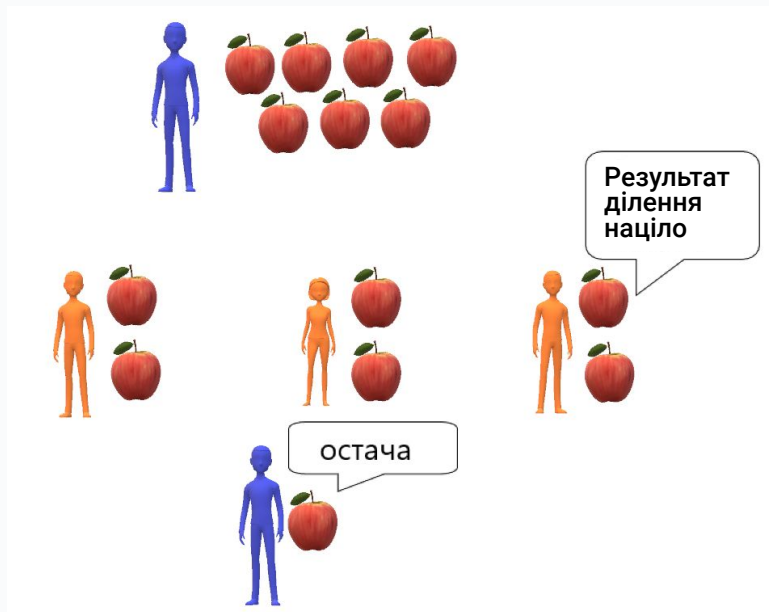
Приклад: "?", "Prometheus", "s", "8"

Логічний тип. Він має лише два значення - True (Істина) і False (Хибність).

Наприклад, цих значень можуть набувати операції порівняння: при $x=5$ вираз $x>0$ набуває значення True, а при $x=-3$ вираз $x>0$ набуває значення False.



Допустимі операції над цілими числами



додавання +

$$1 + 3 = 4$$

віднімання -

$$1 - 7 = -6$$

множення *

$$5 * 3 = 15$$

ділення /

$$6 / 2 = 3$$

остача від ділення націло %

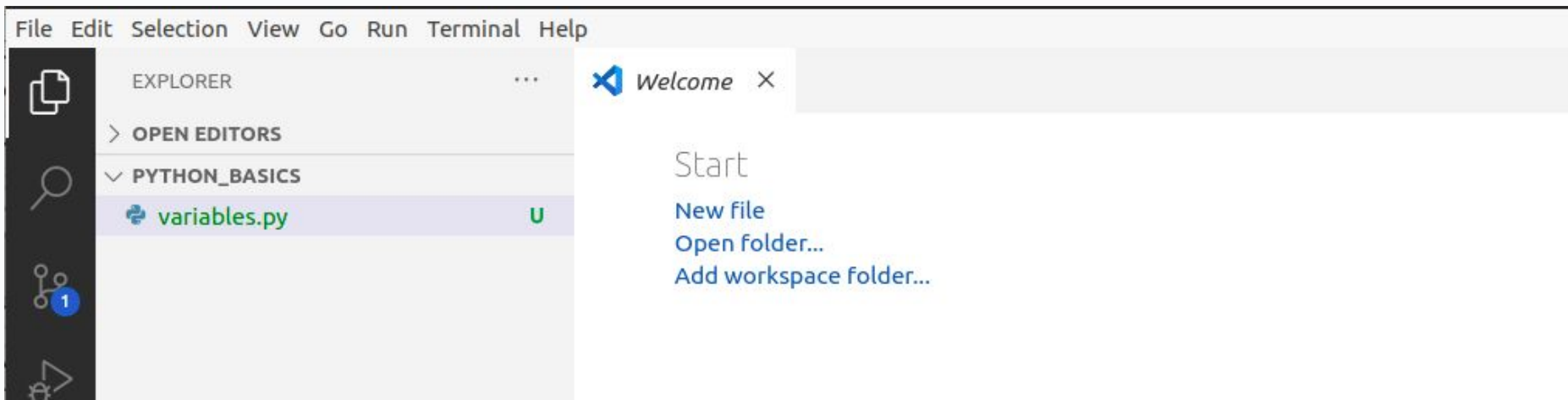
$$7 \% 3 = 1$$

цілочисельне ділення //

$$7 // 3 = 2$$

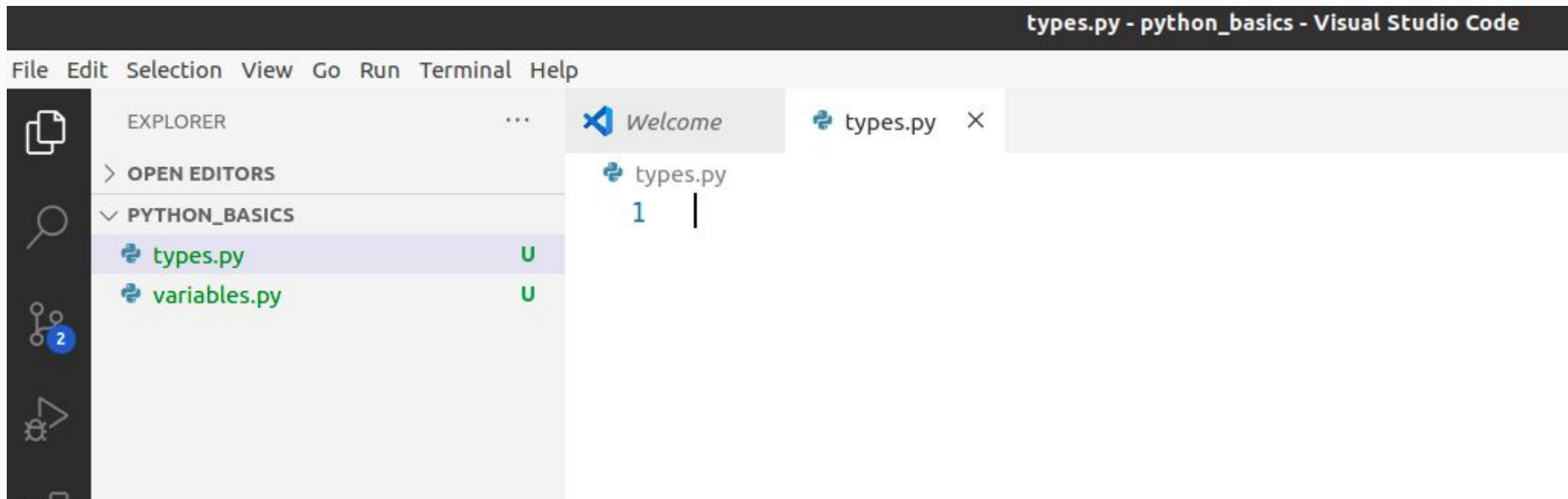


Практика





Практика





Допустимі операції над цілими числами



types.py - python_basics

File Edit Selection View Go Run Terminal Help

EXPLORER

> OPEN EDITORS

PYTHON_BASICS

- types.py 1, U
- variables.py U

Welcome

types.py

```
1 a = 7
2 b = 3
3
4 print("Сума", a + b)
5 print("Різниця", a - b)
6 print("Добуток", a * b)
7 print("Частка", a / b)
8 print("Остача від ділення", a % b)
```




Допустимі операції над цілими числами



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a project named 'PYTHON_BASICS' with two files: 'types.py' and 'variables.py'. The 'types.py' file is open in the editor, showing the following code:

```
1 a = 7
2 b = 3
3
4 print("Сума", a + b)
5 print("Різниця", a - b)
6 print("Добуток", a * b)
7 print("Частка", a / b)
8 print("Остача від ділення", a % b)
```

The Run and Debug toolbar at the top right of the editor has a green play button icon highlighted with a red rectangle. Below the editor, the Terminal panel shows the output of running the script:

```
python_basics git:(master) x /usr/bin/python3.8 "/home/sbutenko/repos/LnD/Become QA Auto/python_basics/types.py"
Сума 10
Різниця 4
Добуток 21
Частка 2.3333333333333335
Остача від ділення 1
python_basics git:(master) x
```

⦿ Допустимі операції над дійсними числами



додавання +

$$1 + 3.2 = 4.2$$

віднімання -

$$7 - 1,5 = 5,5$$

множення *

$$1.5 * 3 = 4.5$$

ділення /

$$6 / 1.5 = 4.0$$



Допустимі операції над дійсними числами



```
EXPLORER
└─ PYTHON...
   ├── types_2.py
   ├── types_3.py
   ├── types_4.py
   └─ types.py

types_2.py
1  c = 3.14
2  d = 9.8
3
4  print("Сума", c + d)
5  print("Різниця", c - d)
6  print("Добуток", c * d)
7  print("Частка", c / d)
8
```



Допустимі операції над дійсними числами



types_2.py X

types_2.py > ...

```
1 c = 3.14
2 d = 9.8
3
4 print("Сума", c + d)
5 print("Різниця", c - d)
6 print("Добуток", c * d)
7 print("Частка", c / d)
8 |
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

JUPYTER

Windows PowerShell

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell <https://aka.ms/pscore6>

```
PS C:\Users\sergii.butenko\python_basics> & C:/Users/sergii.butenko/AppData/Local/Programs/Python/Python310/python.exe c:/Users/sergii.butenko/python_basics/types_2.py
Сума 12.940000000000001
Різниця -6.66
Добуток 30.772000000000002
Частка 0.32040816326530613
PS C:\Users\sergii.butenko\python_basics>
```



Pyth

Pyth

Pyth

⊙ Допустимі операції над символьними величинами



Конкатенація (склеювання) позначається +.

Результат: до кінця першого значення
«доклеюється» початок другого

Приклади:

«шко» + «ла» = «школа»

«8» + «6» = «86»

Допустимі операції над символьними величинами



The screenshot shows a code editor interface. On the left is the 'EXPLORER' sidebar with a folder named 'PYTHON_BASICS' containing four files: 'types_2.py', 'types_3.py' (selected), 'types_4.py', and 'types.py'. The main editor area shows the contents of 'types_3.py' with the following code:

```
types_3.py > [1] first_part
1  first_part = 'prome'
2  last_part = 'theus'
3
4  print(first_part + last_part)
5
6
```



Допустимі операції над символьними величинами



types_3.py X

types_3.py > [e] first_part

```
1 first_part = 'prome'
2 last_part = 'theus'
3
4 print(first_part + last_part)
5
6
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

JUPYTER

```
PS C:\Users\sergii.butenko\python_basics> & C:/Users/sergii.bute
basics/types_3.py
prometheus
PS C:\Users\sergii.butenko\python_basics> █
```

⦿ Логічний (булевий) тип



Операції порівняння:

більше	>	$5 > 4$
менше	<	$-9 < 0$
більше рівне	\geq	$3 \geq 2$
менше рівне	\leq	$10 \leq 10$
рівне	$==$	$2+2 == 3+1$
не рівне	$!=$	$5 != 2+2$

⦿ Логічний (булевий) тип



The screenshot shows a code editor interface. On the left is the 'EXPLORER' sidebar with a folder named 'PYTHON_BASICS' containing five files: 'types_2.py', 'types_3.py', 'types_4.py' (which is selected and highlighted), and 'types.py'. On the right, the 'types_4.py' file is open, displaying the following Python code:

```
1 print(5 > 4)
2 print(-9 < 0)
3 print(3 >= 2)
4 print(10 <= 10)
5 print(2+2 == 3+1)
6 print(5 != 2+2)
7
```



Логічний (булевий) тип



The screenshot shows a code editor with a file named `types_4.py` open. The file contains the following Python code:

```
1 print(5 > 4)
2 print(-9 < 0)
3 print(3 >= 2)
4 print(10 <= 10)
5 print(2+2 == 3+1)
6 print(5 != 2+2)
7
```

Below the code editor, the terminal window is visible, showing the command to run the script and its output:

```
PS C:\Users\sergii.butenko\python_basics> & C:/Users/sergii
basics/types_4.py
True
True
True
True
True
True
True
PS C:\Users\sergii.butenko\python_basics>
```



Катастрофа “Аріан-5”



- Помилка у програмі під час перетворення дійсного числа, яке зберігалось у 8 байтах пам'яті, у ціле, для збереження якого було виділено 2 байти пам'яті.
- Надто велике значення дійсного числа не вмістилося в 2 байтах, що спричинило переповнення.
- 4 червня 1996 року під час першого запуску на 34 секунді польоту ракета вибухнула

⦿ **Документальні кадри катастрофи**



https://youtu.be/gp_D8r-2hwk

Особливості типізації в Python



Явно-типізовані мови вимагають, щоб тип нових змінних, функцій і їх аргументів задавався явно.

Мови з **неявною** типізацією не передбачають вказання типів змінних, функцій та їх аргументів.

Наприклад:

<code>int sum = 4</code>	- явна типізація
<code>count = 25.5</code>	- неявна типізація

Статична типізація визначається тим, що типи змінних і функцій встановлюються на етапі створення. В **динамічній типізації** усі типи даних з'ясовуються вже під час виконання програми.

Наприклад, динамічна типізація дозволяє таке:

```
count = 74  
count = "Павло"
```

Сильна типізація - не дозволяє змішувати у виразах різні типи та не виконує автоматичні неявні перетворення. Мови зі **слабкою типізацією** виконують багато неявних перетворень автоматично.

Наприклад, сильна типізація забороняє таке:

```
a = "Число" + 2
```

Мова Python - це мова, що підтримує динамічну, сильну і неявну типізацію:

- не вимагає вказання типів;
- типи даних визначаються присвоєним значенням;
- не можна змішувати у виразах різні типи.



◎ Перетворення типів в Python

- `int(str)` – перетворити рядок на ціле число.
- `float(str)` – перетворити рядок на число з плаваючою комою.
- `bool(val)` – перетворити значення на логічне значення, `true` або `false` (якщо `val` – відсутнє, рівне 0 чи 0.0 то результат `false`, інакше - `true`).
- `str(val)` – повертає рядкове представлення значення.

`a = "Число" + str(2)` – дозволяє сильна типізація. Результат: "Число2"

Використовуйте команду `type()`, щоб отримати тип значення змінної

Наприклад,

```
count = 74
```

```
type(count)      - поверне int
```

```
count = "Павло"
```

```
type(count)      - поверне str
```



Підсумки



Тип даних - це характеристика будь якої змінної, яка визначає:

- яких значень може набувати змінна;
- операції, які можна робити над цими даними;
- який обсяг пам'яті виділяється для збереження цієї величини.



Основними типами даних є:

- цілі числа,
- дійсні числа,
- символьний,
- булевий (логічний).