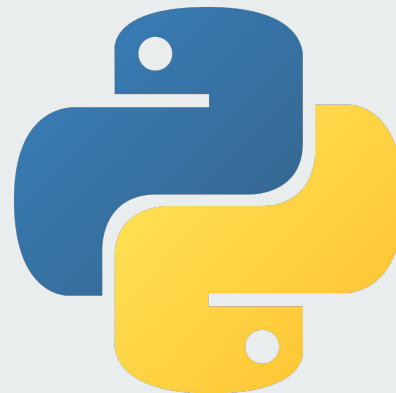




Урок 10

Різниця між стрічками та числами. Введення даних

Типи даних. Перетворення типів даних.
Функції `type()`, `int()`, `float()`, `str()`, `input()`.

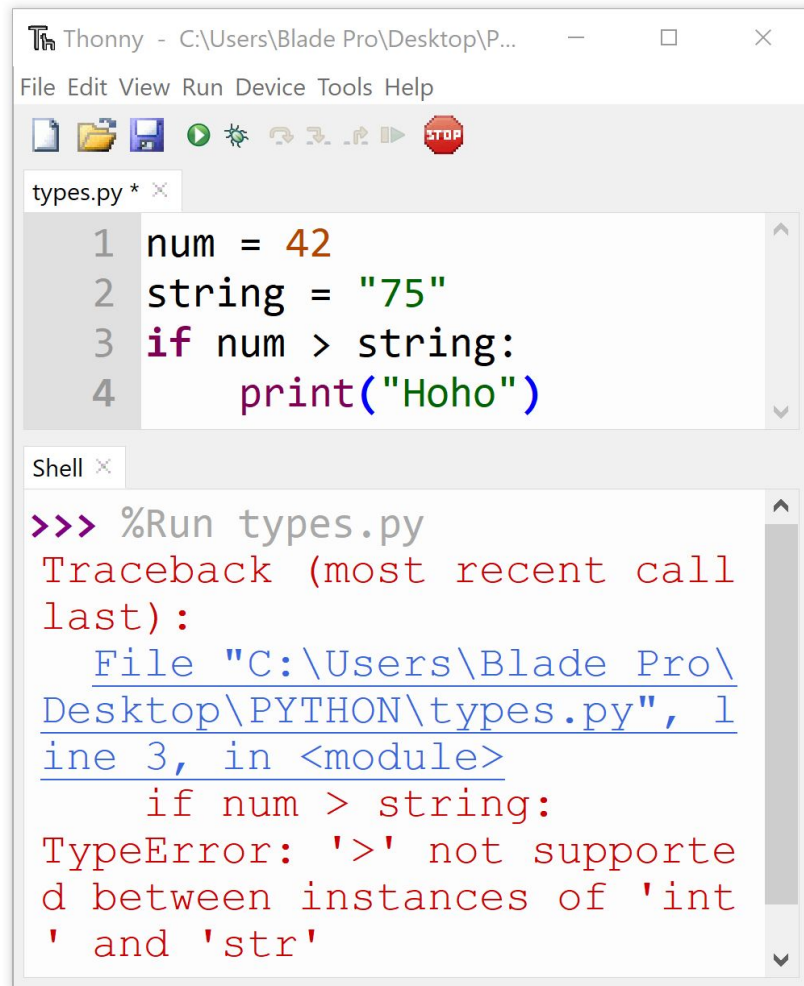


Тип даних

визначає, які операції ми можемо проводити з цією конкретною інформацією.

У Python є такі типи даних:

- ціле число (**integer**)
- число з крапкою (**float**)
- стрічка (**string**)
- boolean - **True/False**



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named `types.py` with the following code:

```
1 num = 42
2 string = "75"
3 if num > string:
4     print("Hoho")
```

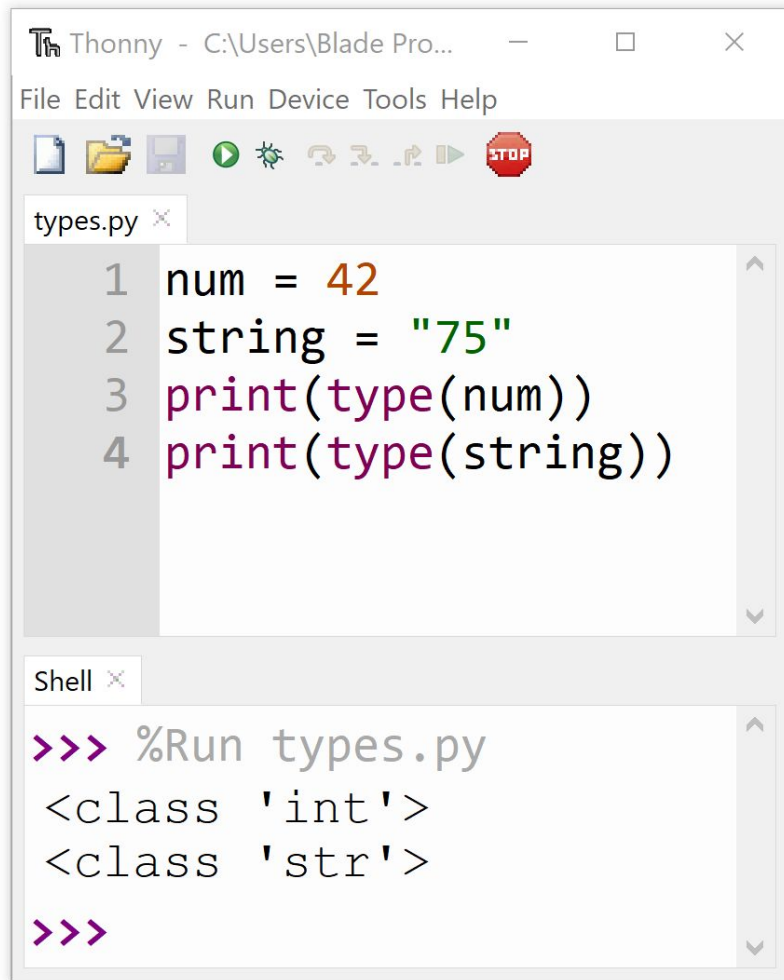
Below the editor is a Shell window showing the execution of the script. The prompt `>>>` is followed by `%Run types.py`. The output shows a `Traceback (most recent call last):` error, indicating a `TypeError: '>' not supported between instances of 'int' and 'str'`. The error message is color-coded, with the comparison operator `>` in red and the variable names `int` and `str` in blue.

Типи даних

Стрічки та числа з однаковим вмістом не дорівнюють одне одному, тому що це різні типи даних.

10 != "10"

Щоб дізнатись тип даних, використовують функцію **type()**.



The screenshot shows the Thonny Python IDE interface. The main editor window displays a file named `types.py` with the following Python code:

```
1 num = 42
2 string = "75"
3 print(type(num))
4 print(type(string))
```

Below the editor is the Shell window, which shows the output of running the script:

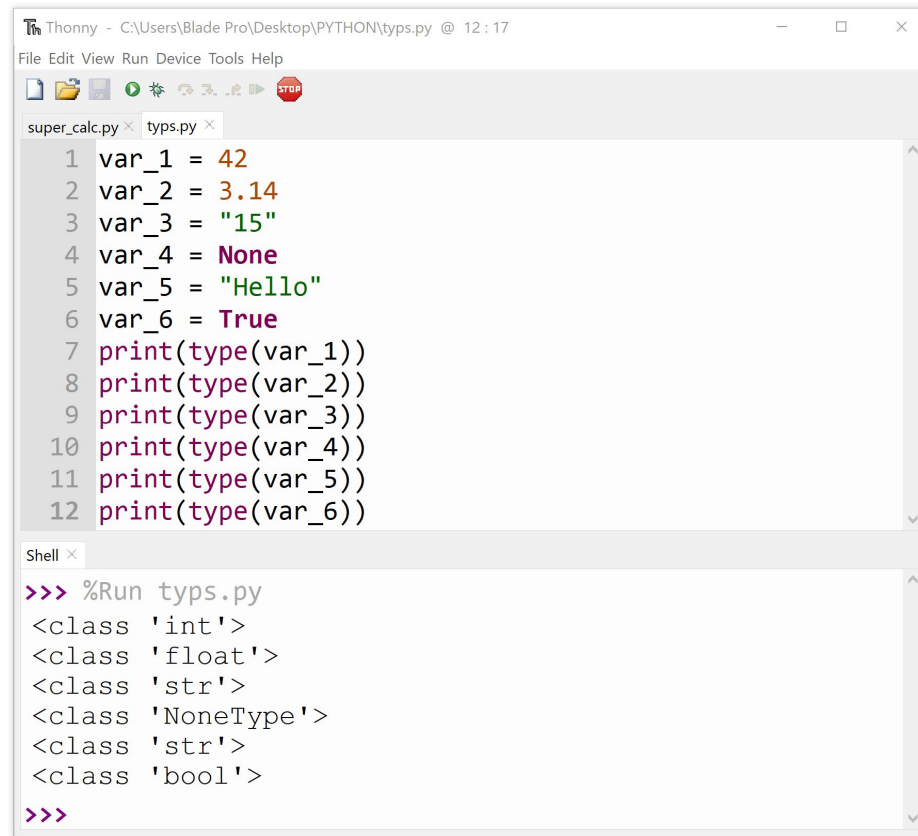
```
>>> %Run types.py
<class 'int'>
<class 'str'>
>>>
```



Практична робота

- 1) Створіть новий файл (**New**)
- 2) Створіть змінні, наведені справа та збережіть файл (**Save**) під назвою **new_types.py**
- 3) Виведіть інформацію про тип даних цих змінних у вікно **Shell** за допомогою функцій **print()** та **type()**.

```
var_1 = 42
var_2 = 3.14
var_3 = "15"
var_4 = None
var_5 = "Hello"
var_6 = True
```



The image shows a screenshot of the Thonny Python IDE. The window title is "Thonny - C:\Users\Blade Pro\Desktop\PYTHON\typs.py @ 12:17". The menu bar includes "File", "Edit", "View", "Run", "Device", "Tools", and "Help". Below the menu bar is a toolbar with icons for file operations and running code. The main editor area has two tabs: "super_calc.py" and "typs.py". The "typs.py" tab is active and contains the following Python code:

```
1 var_1 = 42
2 var_2 = 3.14
3 var_3 = "15"
4 var_4 = None
5 var_5 = "Hello"
6 var_6 = True
7 print(type(var_1))
8 print(type(var_2))
9 print(type(var_3))
10 print(type(var_4))
11 print(type(var_5))
12 print(type(var_6))
```

Below the editor is a "Shell" tab, which is also active. It shows the output of running the script:

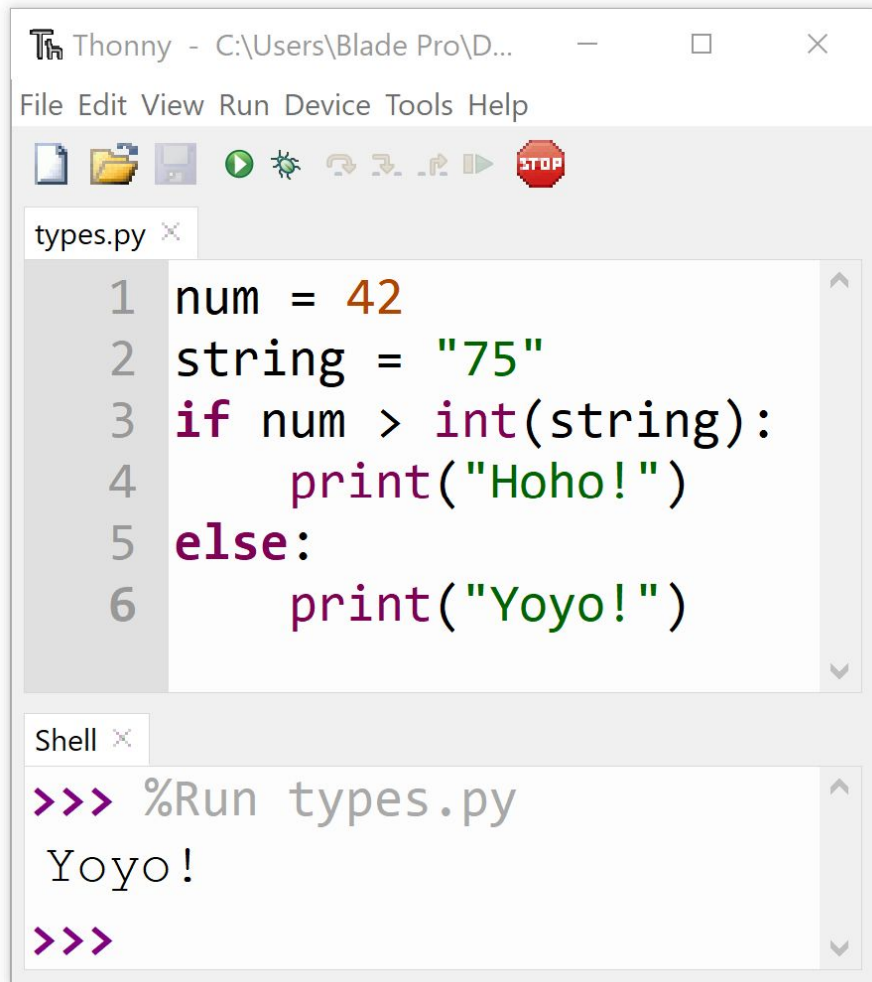
```
>>> %Run typs.py
<class 'int'>
<class 'float'>
<class 'str'>
<class 'NoneType'>
<class 'str'>
<class 'bool'>
>>>
```

Якщо все зроблено правильно - ви побачите такий результат

Перетворення стрічки на число

Щоб перетворити стрічку на ціле число, використовують функцію **int()**.

Щоб перетворити стрічку на число з крапкою (не ціле), використовують функцію **float()**.



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named 'types.py' with the following code:

```
1 num = 42
2 string = "75"
3 if num > int(string):
4     print("Hoho!")
5 else:
6     print("Yoyo!")
```

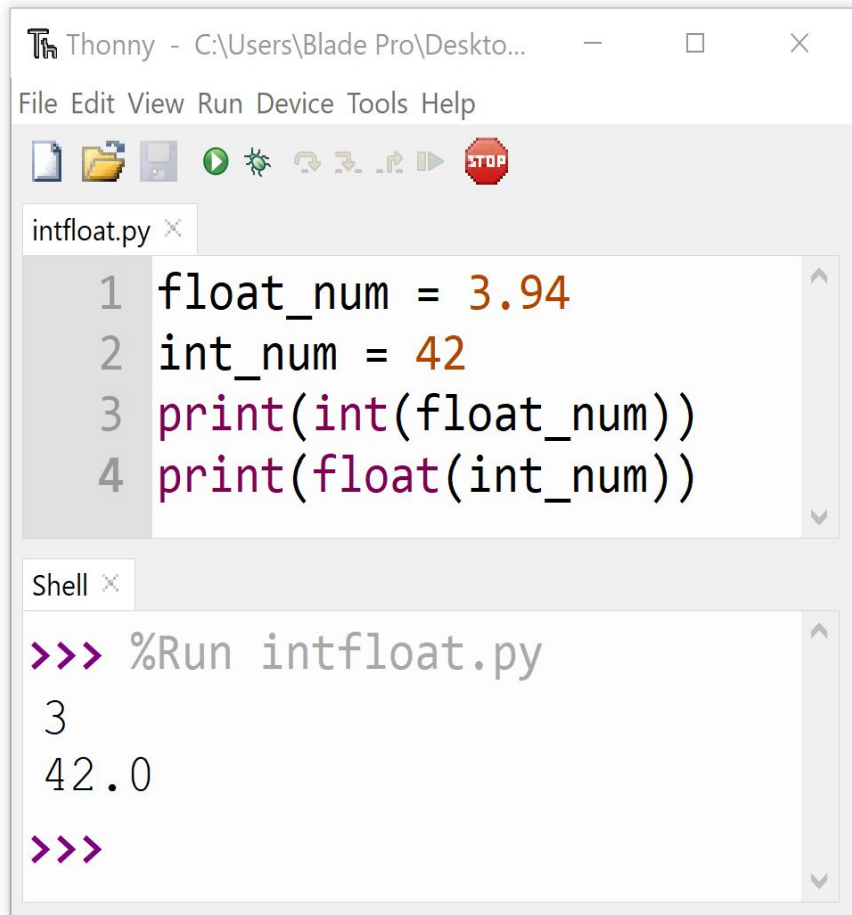
Below the editor, the Shell window shows the execution output:

```
>>> %Run types.py
Yoyo!
>>>
```

Перетворення числа на число

Щоб перетворити число з крапкою (не ціле) на ціле число, використовують функцію **int()**.
Все, що було після крапки, зникне.

Щоб перетворити ціле число на число з крапкою (не ціле), використовують функцію **float()**.



The screenshot shows the Thonny Python IDE interface. The main editor window displays a script named `intfloat.py` with the following code:

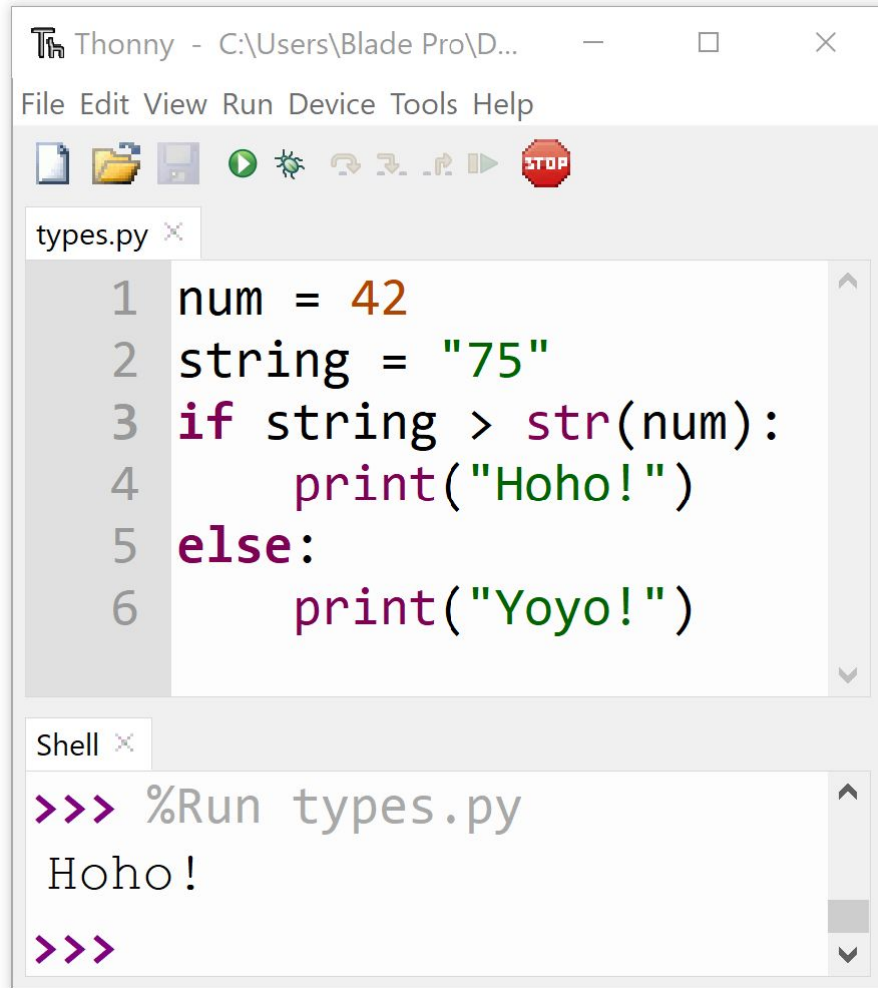
```
1 float_num = 3.94
2 int_num = 42
3 print(int(float_num))
4 print(float(int_num))
```

Below the editor is the Shell window, which shows the command `>>> %Run intfloat.py` and the resulting output:

```
3
42.0
>>>
```

Перетворення числа на стрічку

Щоб перетворити число на стрічку, використовують функцію **str()**.



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named `types.py` with the following code:

```
1 num = 42
2 string = "75"
3 if string > str(num):
4     print("Hoho!")
5 else:
6     print("Yoyo!")
```

Below the editor is a Shell window showing the execution of the script:

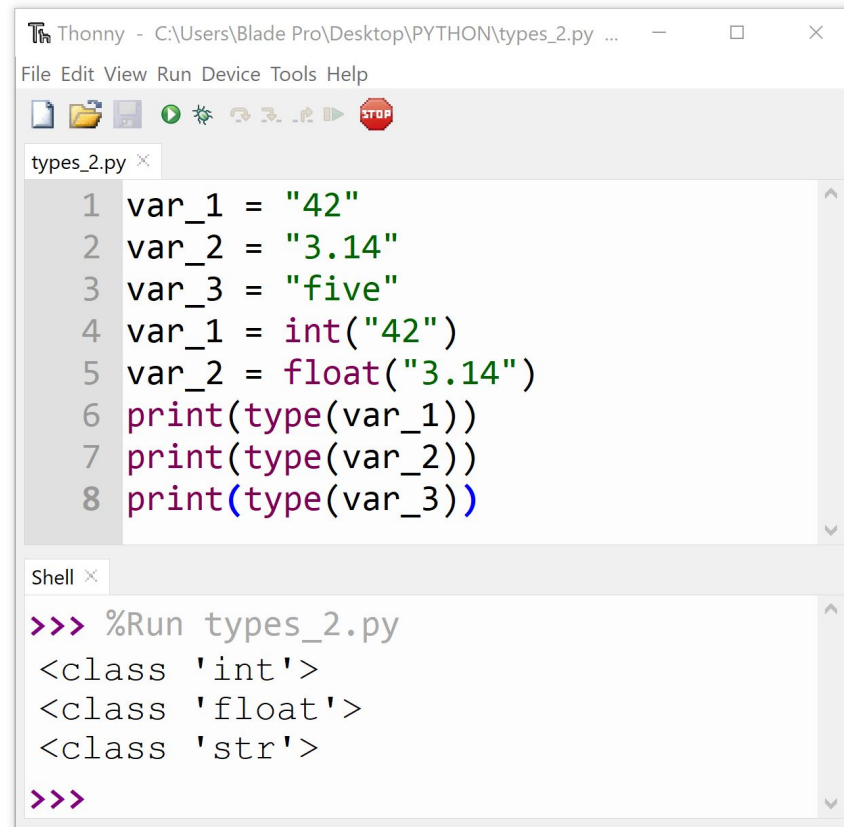
```
>>> %Run types.py
Hoho!
>>>
```




Практична робота

- 1) Створіть новий файл (**New**)
- 2) Створіть змінні, наведені справа та збережіть файл (**Save**) під назвою **types_2.py**
- 3) Перетворіть стрічки у числа з правильним типом даних та виведіть інформацію про тип даних цих змінних у вікно **Shell** за допомогою функцій **print()** та **type()**..

```
var_1 = "42"  
var_2 = "3.14"  
var_3 = "five"
```



The image shows a screenshot of the Thonny Python IDE. The window title is "Thonny - C:\Users\Blade Pro\Desktop\PYTHON\types_2.py ...". The menu bar includes "File", "Edit", "View", "Run", "Device", "Tools", and "Help". Below the menu bar is a toolbar with icons for file operations and execution. The main editor area shows a file named "types_2.py" with the following code:

```
1 var_1 = "42"  
2 var_2 = "3.14"  
3 var_3 = "five"  
4 var_1 = int("42")  
5 var_2 = float("3.14")  
6 print(type(var_1))  
7 print(type(var_2))  
8 print(type(var_3))
```

Below the editor is a "Shell" window showing the output of running the script:

```
>>> %Run types_2.py  
<class 'int'>  
<class 'float'>  
<class 'str'>  
>>>
```

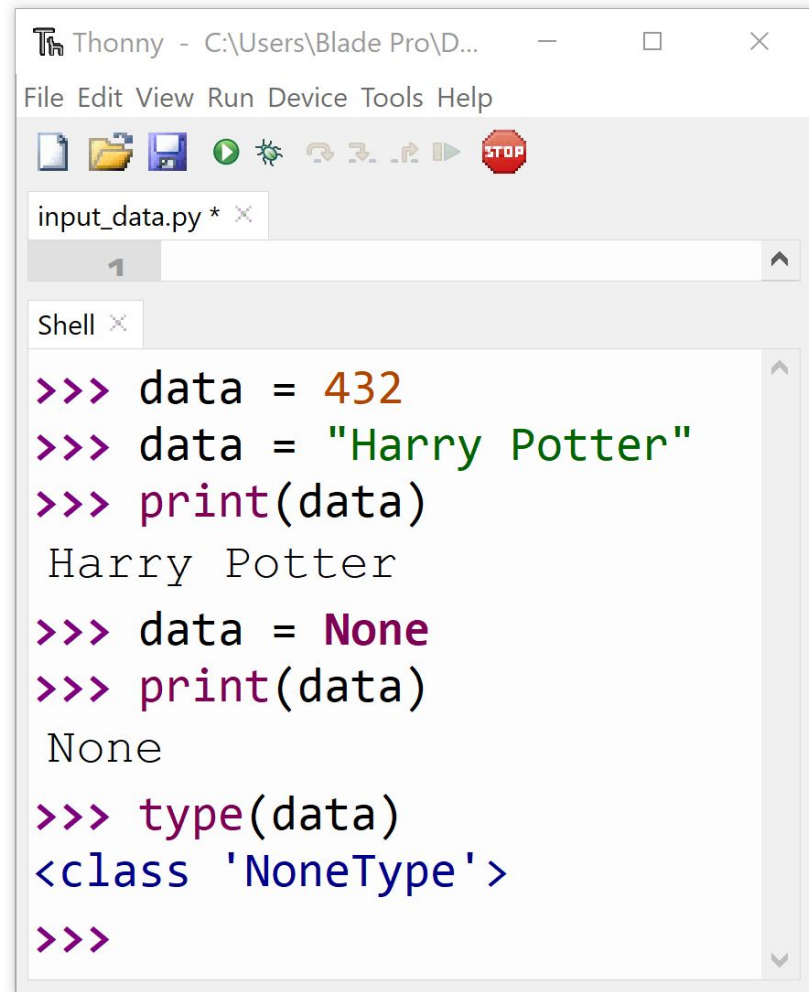
Якщо все зроблено правильно - ви побачите такий результат

Введені користувачем дані

це те, що людина набирає на клавіатурі:

- символ
- натиснення стрілки
- натиснення клавіші ENTER
- тощо

Python сприймає введені дані, як **стрічку**.



```
Thonny - C:\Users\Blade Pro\D...
File Edit View Run Device Tools Help

input_data.py * x
1

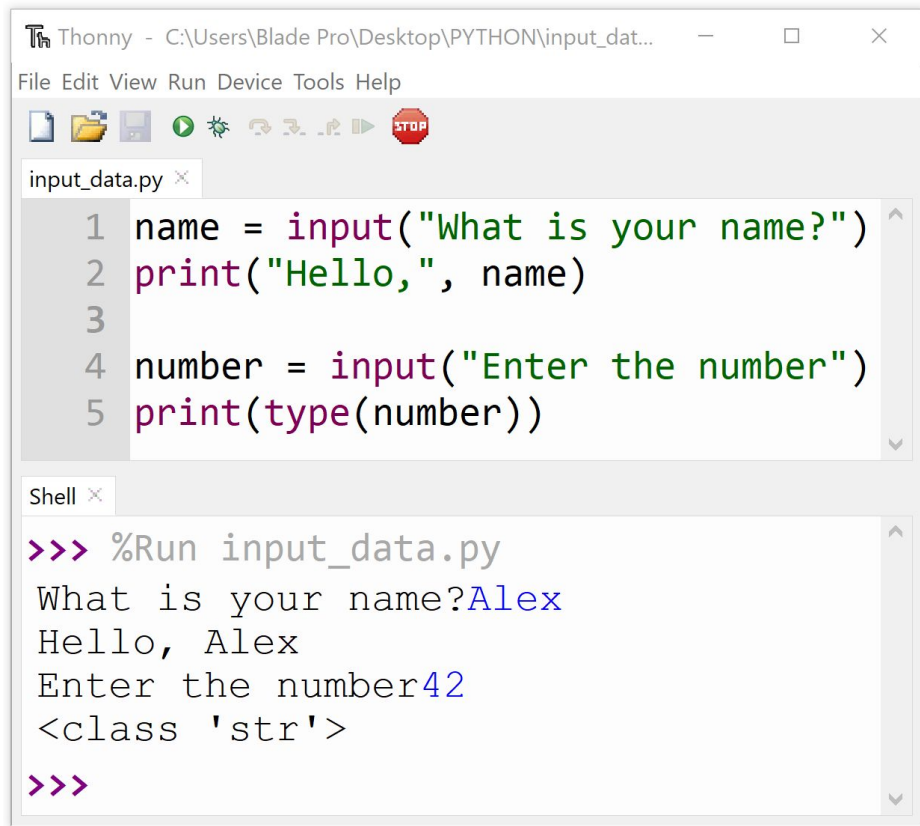
Shell x
>>> data = 432
>>> data = "Harry Potter"
>>> print(data)
Harry Potter
>>> data = None
>>> print(data)
None
>>> type(data)
<class 'NoneType'>
>>>
```

Введення даних

Для отримання даних від користувача використовується функція **input()**.
У дужках можна записати питання, що ми очікуємо від користувача.

Python сприймає введені дані, як стрічку.

Щоб використовувати введені дані для розрахунків, треба перетворити стрічки на числа.



The screenshot shows the Thonny Python IDE interface. The top window displays the code for `input_data.py`:

```
1 name = input("What is your name?")
2 print("Hello,", name)
3
4 number = input("Enter the number")
5 print(type(number))
```

The bottom window, titled "Shell", shows the execution output:

```
>>> %Run input_data.py
What is your name?Alex
Hello, Alex
Enter the number42
<class 'str'>
>>>
```

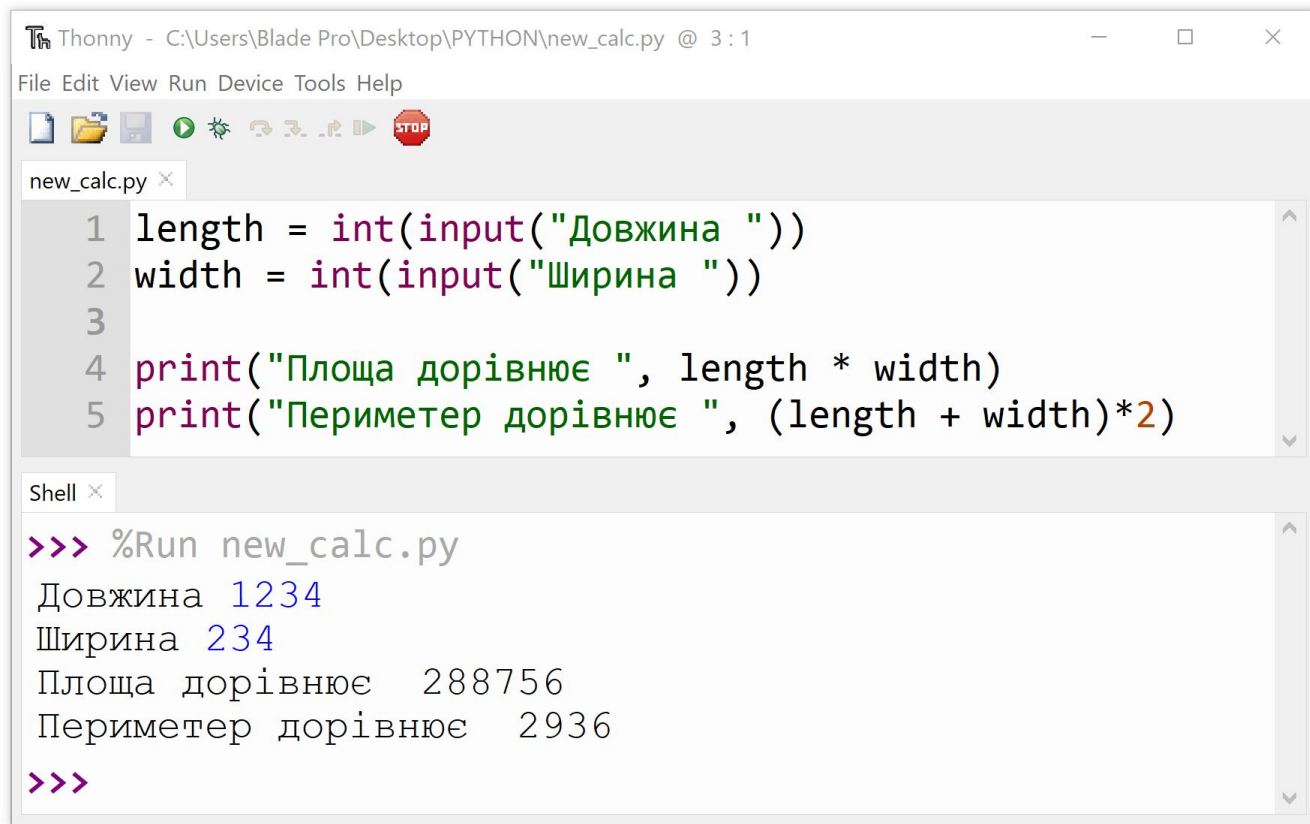


Практична робота

- 1) Створіть новий файл (**New**)
- 2) Створіть нову програму для розрахунку площі або периметру з використанням функції **input()**, щоб отримувати інформацію про довжину та ширину від користувача. Збережіть файл (**Save**) під назвою **new_calc.py**

Додаткове завдання

Створіть нову програму для розрахунку площі та периметру з використанням **умов** та функції **input()**, щоб отримувати інформацію про довжину та ширину від користувача. Збережіть файл (**Save**) під назвою **super_calc.py**



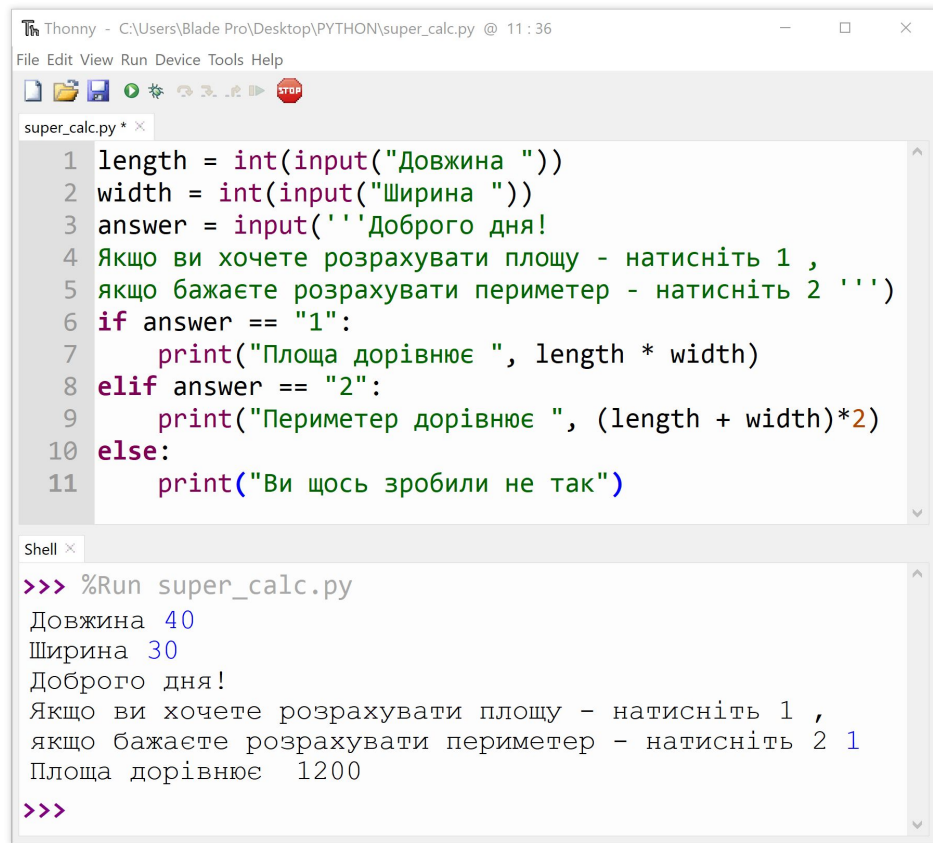
The image shows a screenshot of the Thonny Python IDE. The title bar reads "Thonny - C:\Users\Blade Pro\Desktop\PYTHON\new_calc.py @ 3 : 1". The menu bar includes "File", "Edit", "View", "Run", "Device", "Tools", and "Help". Below the menu is a toolbar with icons for file operations and execution. The main editor window displays a Python script named "new_calc.py" with the following code:

```
1 length = int(input("Довжина "))
2 width = int(input("Ширина "))
3
4 print("Площа дорівнює ", length * width)
5 print("Периметер дорівнює ", (length + width)*2)
```

Below the editor is a "Shell" window showing the execution output:

```
>>> %Run new_calc.py
Довжина 1234
Ширина 234
Площа дорівнює 288756
Периметер дорівнює 2936
>>>
```

Якщо все зроблено правильно - ви побачите такий результат



The image shows a screenshot of the Thonny IDE window. The title bar indicates the file path is C:\Users\Blade Pro\Desktop\PYTHON\super_calc.py and the time is 11:36. The menu bar includes File, Edit, View, Run, Device, Tools, and Help. Below the menu is a toolbar with icons for file operations and running the code. The main editor area shows a Python script named super_calc.py with the following code:

```
1 length = int(input("Довжина "))
2 width = int(input("Ширина "))
3 answer = input('Доброго дня!
4 Якщо ви хочете розрахувати площу - натисніть 1 ,
5 якщо бажаєте розрахувати периметер - натисніть 2 ')
6 if answer == "1":
7     print("Площа дорівнює ", length * width)
8 elif answer == "2":
9     print("Периметер дорівнює ", (length + width)*2)
10 else:
11     print("Ви щось зробили не так")
```

Below the editor is a Shell window showing the execution of the script:

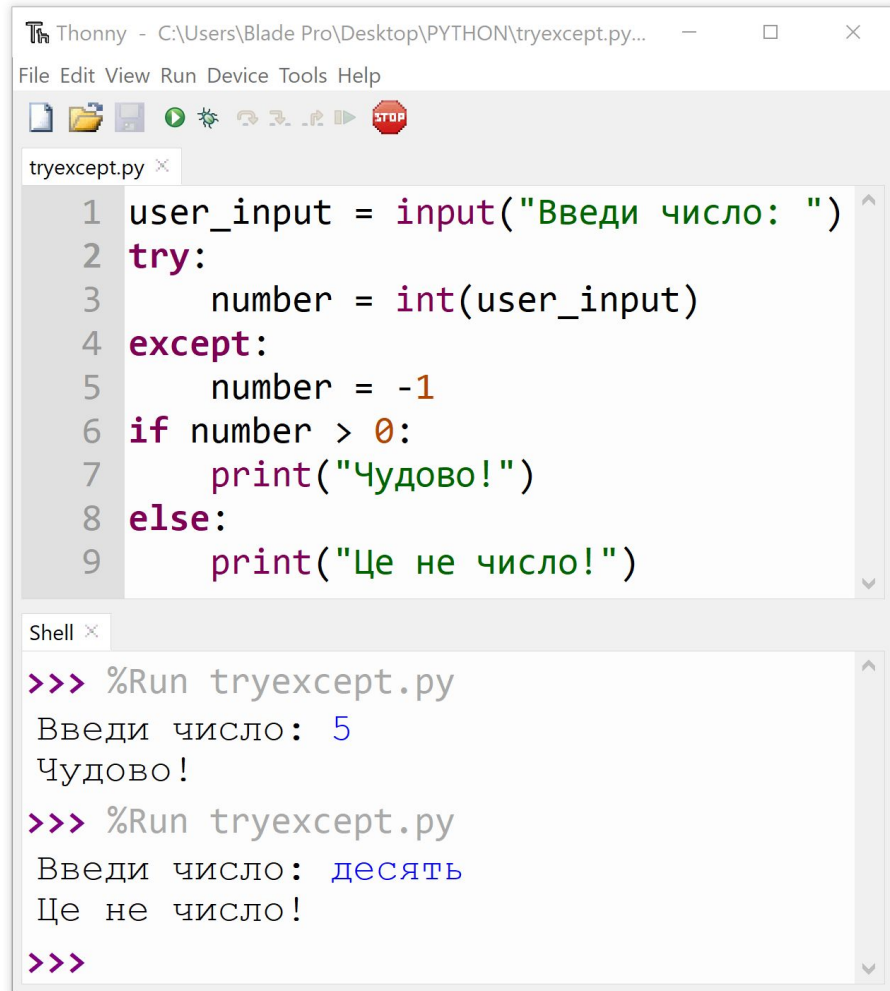
```
>>> %Run super_calc.py
Довжина 40
Ширина 30
Доброго дня!
Якщо ви хочете розрахувати площу - натисніть 1 ,
якщо бажаєте розрахувати периметер - натисніть 2 1
Площа дорівнює 1200
>>>
```

Якщо все зроблено правильно - ви побачите такий результат

Оператори try та except

Якщо є вірогідність, що користувач може ввести некоректні дані, які призводять до помилки, використовують виключення за допомогою операторів **try** та **except**.

Якщо у блоці **try** виникає помилка, то виконується команда з блоку **except**.



```
Thonny - C:\Users\Blade Pro\Desktop\PYTHON\tryexcept.py...
File Edit View Run Device Tools Help

tryexcept.py
1 user_input = input("Введи число: ")
2 try:
3     number = int(user_input)
4 except:
5     number = -1
6 if number > 0:
7     print("Чудово!")
8 else:
9     print("Це не число!")

Shell
>>> %Run tryexcept.py
Введи число: 5
Чудово!
>>> %Run tryexcept.py
Введи число: десять
Це не число!
>>>
```




Практична робота

Створіть програму для розрахунку площі або периметру з використанням **умов**, функції **input()** та перевіркою вводу даних від користувача (**try, except**).

Спочатку програма повинна отримати інформацію про довжину та ширину від користувача.

Потім програма повинна запитати, що саме хоче розрахувати користувач - площу чи периметр - та виконати розрахунок обраної величини.

Збережіть файл (**Save**) під назвою **puper_calc.py**



Підсумки

Навчилися працювати з **різними типами даних**, перетворювати стрічки у числа та навпаки за допомогою спеціальних функцій

Дізнались, як **отримувати та використовувати** у програмі введені дані від користувача