Programming Fundamental

A Hitchhiker Guide to Coding with Python

Lesson 2: Handling of Inputs

Akarate Singta

Department of Mathematics
Faculty of Science and Technology, RMUTT

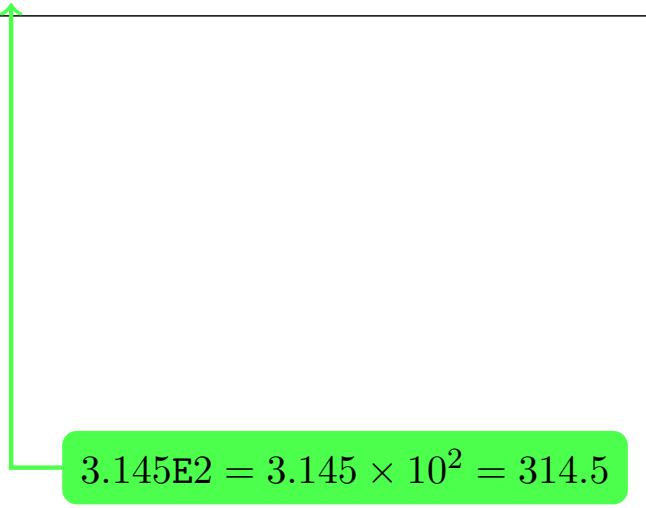
Lesson Outline

Basic Data Types

Class	Examples	Data Type
`int`	`4`, `-290`, `65_000_000`, `...`	Integer Numbers
`float`	`4.0`, `-0.29`, `15.458`, `3.145E2`, `1.22e-3` `...`	Floating Point Numbers

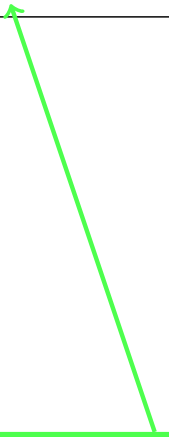
Basic Data Types

Class	Examples	Data Type
`int`	`4`, `-290`, `65_000_000`, `...`	Integer Numbers
`float`	`4.0`, `-0.29`, `15.458`, `3.145E2`, `1.22e-3` `...`	Floating Point Numbers



Basic Data Types

Class	Examples	Data Type
`int`	`4`, `-290`, `65_000_000`, `...`	Integer Numbers
`float`	`4.0`, `-0.29`, `15.458`, `3.145E2`, `1.22e-3` `...`	Floating Point Numbers



$$1.22\mathrm{E}-3 = 1.22 \times 10^{-3} = 0.00112$$

Basic Data Types

Class	Examples	Data Type
`int`	`4, -290, 65_000_000, ...`	Integer Numbers
`float`	`4.0, -0.29, 15.458, 3.145E2, 1.22e-3 ...`	Floating Point Numbers
`complex`	`3+2j, -2+0.5j, -3.14j, -5.1+3.114e2j, ...`	Complex Numbers

Basic Data Types

Class	Examples	Data Type
`int`	`4, -290, 65_000_000, ...`	Integer Numbers
`float`	`4.0, -0.29, 15.458, 3.145E2, 1.22e-3 ...`	Floating Point Numbers
`complex`	`3+2j, -2+0.5j, -3.14j, -5.1+3.114e2j, ...`	Complex Numbers
`str`	`'Hello', "World", '2022', 's', '*!4aP(&^4)', ...`	Strings

Basic Data Types

Class	Examples	Data Type
`int`	`4, -290, 65_000_000, ...`	Integer Numbers
`float`	`4.0, -0.29, 15.458, 3.145E2, 1.22e-3 ...`	Floating Point Numbers
`complex`	`3+2j, -2+0.5j, -3.14j, -5.1+3.114e2j, ...`	Complex Numbers
`str`	`'Hello', "World", '2022', 's', '*!4aP(&^4)', ...`	Strings
`bool`	`True, False`	Boolean

Basic Data Types

The type of a Python object determines what kind of object it is; every object has a type.

```
>>> type(-255)
<class 'int'>
>>> type(3.1415E5)
<class 'float'>
>>> type(3.0-16.5j)
<class 'complex'>
>>> type(True)
<class 'bool'>
```

Data Operations: Numbers

Operation	Operator	Example	Result
Addition	+	`123 + 45.5`	`168.5`
Subtraction	-	`123 - 45.5`	`77.5`
Multiplication	*	`123 * 45.5`	`5571.9`
Exponentiation	**	`123 ** 4`	`228886641`
Division	/	`123 / 45.5`	`2.7032967`
Floor Division	//	`123 / 45`	`2`
Modulus	%	`123 % 45`	`32`

Data Operations: Strings

Operation	Operator	Example	Result
Addition	+	`'Hello' + 'World'`	`'HelloWorld'`
		`'Hello' + 2022`	☹ TypeError
Multiplication	*	`3 * 'Hello'`	`'HelloHelloHello'`
		`'*' * 10`	`'**********'`
		`'World' * 10.45`	☹ TypeError
		`'Hello' * 'World'`	☹ TypeError

Input Function

Python `input()` function takes user keyboard input. It returns the user input in form of a `string` data type.

```
>>> name = input('Enter your name: ')
Enter your name: John Wick
>>> print(name)
John Wick
>>> type(name)
<class 'str'>
```

Example

ask_name.py

```
1 print('This is how input function works')
2 name = input('Enter your name: ')
3 age = input('Enter your age: ')
4 print('Your name is', name, ', and you are', age, 'years old.')
```

```
>_ python ask_name.py
This is how input function works
Enter your name: John Wick
Enter your age: 42
Your name is John Wick , and you are 42 years old.
```

Example

area_rectangle.py

```
1 print('Rectangle Area Calculator')
2 width = input('Enter the width: ')
3 height = input('Enter the height: ')
4 area = width * height
5 print('Area is', area)
```


```
>_ python area_rectangle.py
Rectangle Area Calculator
Enter the width: 4.5
Enter the height: 6.25
Traceback (most recent call last):
  File ".../area_rectangle.py", line 4, in <module>
    area = width * height
TypeError: can't multiply sequence by non-int of type 'str'
```

Type Casting

Type casting in Python is achieved by constructor functions:

- `int()` - constructs an integer number from an integer literal, a float literal (by removing all decimals), or a string literal
- `float()` - constructs a float number from an integer literal, a float literal or a string literal
- `str()` - constructs a string from a wide variety of data types, including strings, integer literals and float literals

Type Casting

```
>>> x = '123456'
>>> type(x)
<class 'str'>
>>> y = int(x)
>>> type(y)
<class 'int'>
>>> print(y + 1)
123457
```

String Literal

Integer Casting

Type Casting

```
>>> x = '123.456'
>>> type(x)
<class 'str'>
>>> y = float(x)
>>> type(y)
<class 'float'>
>>> print(y + 1)
124.456
```

String Literal

Floating Point Casting

Type Casting

```
>>> x = 3.1414
>>> type(x)
<class 'float'>
>>> y = str(x)
>>> type(y)
<class 'str'>
>>> print(y + '55555')
3.141455555
```

Floating Point Literal

String Casting

Type Casting

```
>>> int('3.1415')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base
    10: '3.1415'
>>> int(3.1415)
3
>>> float('3')
3.0
>>> float(3)
3.0
```

Example

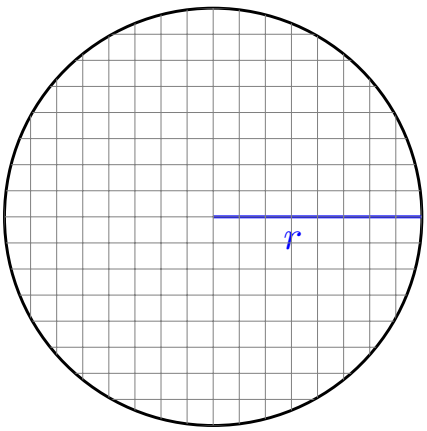
area_rectangle.py

```python
print('Rectangle Area Calculator')
width = float( input('Enter the width: ') )
height = float( input('Enter the height: ') )
area = width * height
print('Area is', area)
```

```
>_ python area_rectangle.py
Rectangle Area Calculator
Enter the width: 4.5
Enter the height: 6.25
Area is 28.125
```

Exercise

Write a Python script to perform the calculation of the *perimeter* and *area* of a circle. It takes an input radius from user.



Perimeter $= 2 \cdot \pi \cdot r$

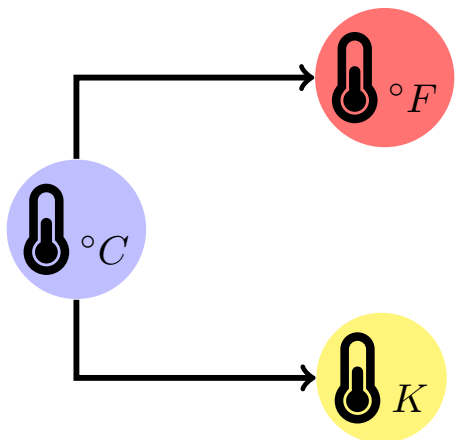
Area $= \pi \cdot r^2$

Exercise

The following formulas

$$\frac{C}{5} = \frac{F - 32}{9} = \frac{K - 273.15}{5}$$

can be used to do conversions of temperatures in degree Celsius (C), Fahrenheit (F) and Kelvin (K). Write a Python script to convert a temperature in degree Celsius to degree Fahrenheit and Kelvin.



Exercise

Equal Monthly Installment (EMI) formula is:

$$\text{EMI} = P \cdot \frac{r \cdot (1+r)^n}{(1+r)^n - 1}$$

where P is a principle (loan); r is a monthly interest rate, and n is a number of payments.
Write a Python script to do calculations of EMI and the amount of monthly payment. It takes inputs P, r and n.