DataTypesAndOperations

April 10, 2020

1 Data types

Data is stored into objects. * A constant is an object * A variable is an object * Actually, everything in Python is an object. We will see more on this Every object has a *type* * The notion of type is very important in computer science. * In Python they are less evident than in other programming languages. But still, everythin has a type in Python.

We are going to discuss some of the Built-in Python types

1.1 Boolean data type

[1]:
```
False
```

[1]: False

[2]:
```
True
```

[2]: True

[3]:
```
print(False,'has type',type(False))
print(True,'has type',type(True))
```

```
False has type <class 'bool'>
True has type <class 'bool'>
```

This is all reasonable but… what can we do with Booleans? * Boolean expressions

[4]:
```
not True
```

[4]: False

[5]:
```
True and False
```

[5]: False

[6]:
```
True or False
```

[6]: True

We can also create Boolean variables

[7]:
```
v=False
print(v,'has type',type(v))
v=True
```

```
False has type <class 'bool'>
```

We can also compare Booleans

[8]:
```
v=False
print('Is v True?',v==True)
v=True
print('Is v True?',v==True)
```

```
Is v True? False
Is v True? True
```

In the coming classes we will see that Boolean expressions are fundamental to write *interesting* programs. They allow us to decide wheter: * A block of instructions should be executed or not * A block of instructions should be executed 0, 1, or more times ...

1.2 Numeric data types

1.2.1 Integers and reals

We are going to use * `int` (integers) * `float` (real values)

[9]:
```
5
n=5
```

[10]:
```
type(5)
```

[10]:
```
int
```

[11]:
```
type(n)
```

[11]:
```
int
```

> Actually, everything has type. Even the `type` of an object has a type. The name of this type is `type` :D

If you execute `type(type(n))` you get `type`

[12]:
```
type(type(n))
```

[12]:
```
type
```

[13]:
```
v=5
print(v,'has type',type(v))
v=-5
print(v,'has type',type(v))
v=5.1
print(v,'has type',type(v))
v=1.2e-3
print(v,'has type',type(v))
v=1.2E-3
print(v,'has type',type(v))
v=1.2e+3
print(v,'has type',type(v))
```

```
5 has type <class 'int'>
-5 has type <class 'int'>
5.1 has type <class 'float'>
0.0012 has type <class 'float'>
0.0012 has type <class 'float'>
1200.0 has type <class 'float'>
```

There are also other numeric types, but we are not going to use them * binary, octal, hexadecimal numbers * complex numbers

1.2.2 Arithmetic comparisons

[14]:
```
### Arithmetic operations
```

[15]:
```
type(1.2e-3)
```

[15]: float

[16]:
```
type(0b001)
```

[16]: int

[17]:
```
type(1+2j)
```

[17]: complex

[18]:
```
5 + 2
```

[18]: 7

[19]:
```
5 + (-2)
```

[19]: 3

[20]:
```
5.1 + 2
```

[20]: 7.1

[21]:
```
1.2e-3
```

[21]: 0.0012

[ ]:
```

```