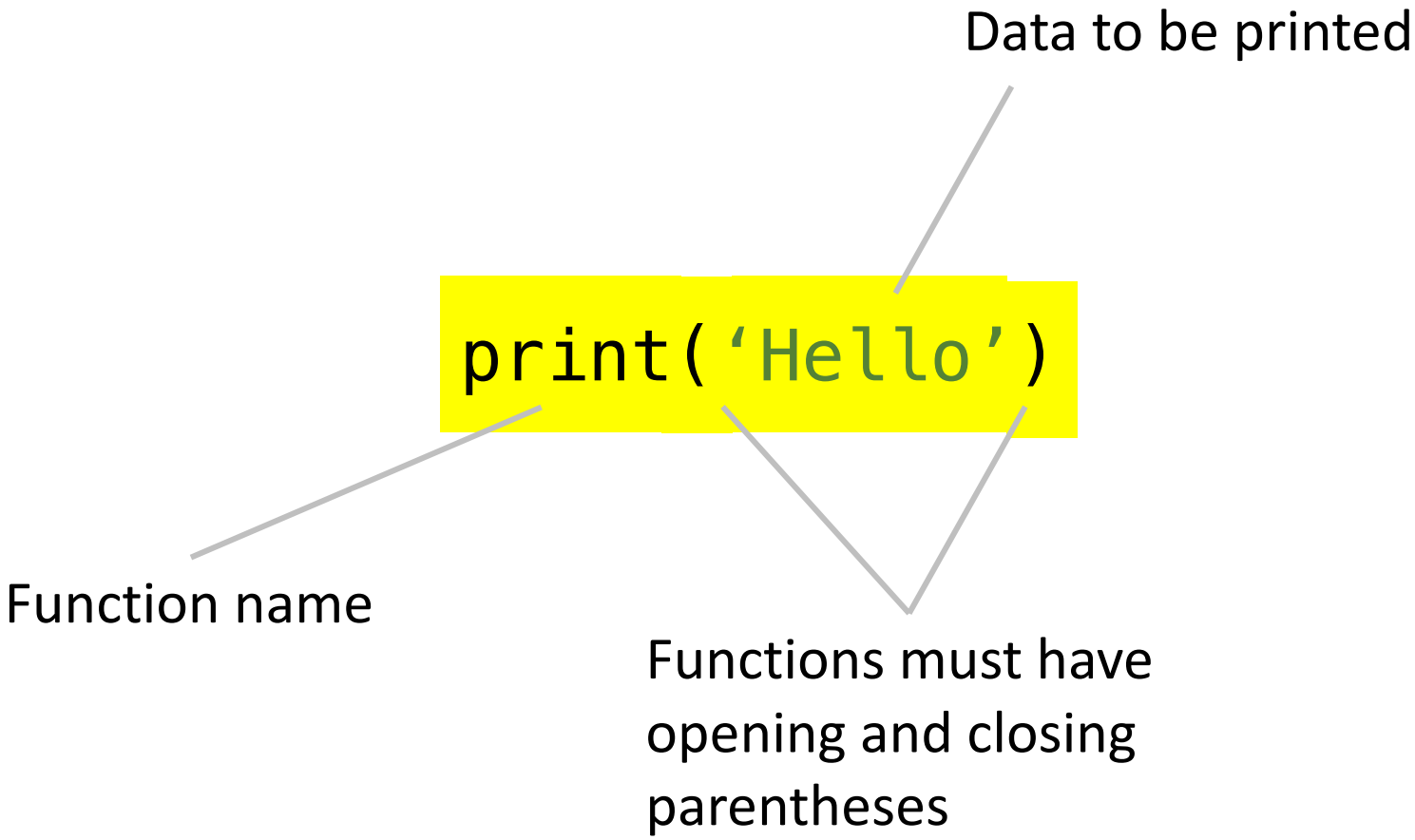
Printing, Expressions

Contents

1. Printing in Python
    - String Data Type
    - Numeric Data Type
    - Printing Arithmetic Expression
2. Arithmetic Operators in Python
3. Operator Precedence
4. String Operators

Printing in Python



String Data Type

Data can be of several types. This is string data.

```
print('Hello')
```

A string data type is made up of symbols e.g. letters, symbols, punctuations and even numbers.

A string must be expressed with a pair of quotation marks.

Experiment

What happens if you omit one or both quotation marks?

```
print('hello)
            ^
SyntaxError: EOL while scanning string literal
```

```
print(hello)

NameError: name 'hello' is not defined
```

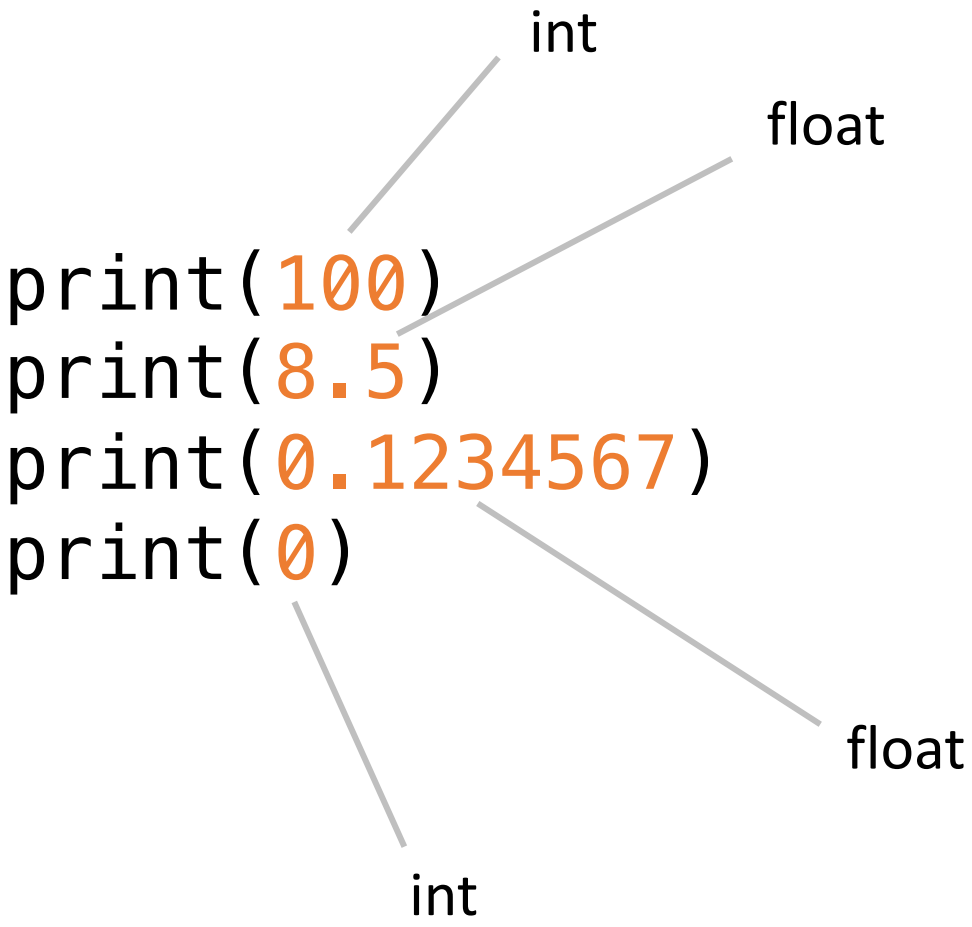
Numeric Data Type

This is a numeric data.

```
print(555)
```

Python has two common types of numeric data, integer and float (i.e. floating-point value)

Numeric Data Type



Printing Arithmetic Expression

You can put an arithmetic expression in a print function call

```
print(5+5)
```

The expression will evaluate to a final number and then printed.

In this case, 10 is printed.

Printing Arithmetic Expression

Arithmetic expressions comprises operators and the operands

```
print(5+5)
```

More accurately, an arithmetic expression is a number or a binary operator applied to two expressions.

- is also a unary operator and can be applied to a single expression.

Practice Exercise

Write a Python statement to print the number of seconds there are in one day. Use arithmetic expressions so that Python does the computation.

```
print(24*60*60)
```

Arithmetic Operators in Python

+	plus	addition
-	minus	subtraction
*	times	multiplication
/	divided by	division
//	divided by	integer division
**	power	exponentiation
%	modulus	remainder
-	negative	negation

Arithmetic Operators in Python

	Example	**Result**
+	5 + 2	7
-	5 - 2	3
*	5 * 2	10
/	5 / 2	2.5
//	5 // 2	2
**	5 ** 2	25
%	5 % 2	1
-	-5	-5

Operator Precedence

```
print(5+5*8)

print(5+(5*8))

print((5+5)*8)
```

Operator Precedence

(…)
x**y
-x
x*y, x/y, x//y, x%y
x+y, x-y

Operator Precedence

```
6 - 3 ** 2 / (1 + 2)
```

Operator Precedence

```
6 - 3 ** 2 / (1 + 2)
6 - 3 ** 2 / 3
```

Operator Precedence

```
6 – 3 ** 2 / (1 + 2)
6 – 3 ** 2 / 3
6 – 9 / 3
```

Operator Precedence

```
6 – 3 ** 2 / (1 + 2)
6 – 3 ** 2 / 3
6 – 9 / 3
6 – 3
```

Operator Precedence

```
6 – 3 ** 2 / (1 + 2)
6 – 3 ** 2 / 3
6 – 9 / 3
6 – 3
3
```

Operator Precedence

```
(6 + (3 – 2 * 2)) ** 2
(6 + (3 – 4)) ** 2
(6 + (–1)) ** 2
(6 – 1) ** 2
5 ** 2
25
```

String Operators

When used on strings, the + symbol functions as a concatenation operator i.e. it combines the two strings.

```
print('Black' + ' Panther')
```

```
print('Amazon river dolphins' + ' are' + ' pink')
```

String Operators

When used on strings, the * symbol multiplies the string by the given number

```
print('ocelot' * 3)
```

String concatenation will not work unless both operands are strings. This expression will throw an error.

```
print('Family of ' + 5 + ' otters')
```

```
TypeError: must be str, not int
```

Use str() function to convert the cast the data to a string.

```
print('Family of ' + str(5) + ' otters')
```

Similarly, arithmetic operators will not work as intended if both its operands are not numbers.

```
print('9' / 3)
```

```
TypeError:
unsupported operand type(s) for /: 'str' and 'int'
```

String Operators

Expression	Examples	Results
`String + string`	`‘Python’ + ‘Code’` `‘Good’ + ‘ day, ’ + ‘world!’`	`‘PythonCode’` `‘Good day, world!`
`String * integer`	`‘Hello’ * 2` `3 * ‘bye ’`	`‘HelloHello’` `‘bye bye bye ' `