Python Programming

Python is a general-purpose programming language that is becoming ever more popular for data science & machine learning. Companies worldwide are using Python to harvest insights from their data and gain a competitive edge.

- Python Programming Fundamentals
- Python Object Oriented Programming
- Python Data Structures

Python Programming Fundamentals

Style Guide for Python Code - PEP 8

Link: https://www.python.org/dev/peps/pep-0008/ (https://www.python.org/dev/peps/pep-0008/)

Link: https://realpython.com/python-pep8/ (https://realpython.com/python-pep8/) - How to Write Beautiful Python Code With PEP 8

Python Basics

An introduction to the basic concepts of Python. Learn how to use Python interactively and by using a script.

- Python Built-in Functions
- Python Variables
- Python Comments
- Python Basic Data Types
    - Integer, Floating Point, Complex, Boolean, String

Python Built-in Functions

Link: https://docs.python.org/3/library/functions.html (https://docs.python.org/3/library/functions.html)

In [1]:
```
# Basic Python Built-in Functions

# print(), help(), type(), dir(), len(), id(), hex(), issubclass(), ...
```

In [2]:
```
print?
```

```
Docstring:
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.
Optional keyword arguments:
file:  a file-like object (stream); defaults to the current sys.stdout.
sep:   string inserted between values, default a space.
end:   string appended after the last value, default a newline.
flush: whether to forcibly flush the stream.
Type:      builtin_function_or_method
```

In [3]:
```
help?
```

```
Signature:   help(*args, **kwds)
Type:        _Helper
String form: Type help() for interactive help, or help(object) for help about
object.
Namespace:   Python builtin
File:        c:\users\neuron\anaconda3\lib\_sitebuiltins.py
Docstring:
Define the builtin 'help'.

This is a wrapper around pydoc.help that provides a helpful message
when 'help' is typed at the Python interactive prompt.

Calling help() at the Python prompt starts an interactive help session.
Calling help(thing) prints help for the python object 'thing'.
```

In [ ]:

In [4]:
```
print('Hey Python')          # display or print string value or data
```

```
Hey Python
```

In [5]:
```
print(45)                    # print integer value
```

```
45
```

In [6]:
```
print(45.67)                 # print float value
```

```
45.67
```

In [7]:
```
print(True)              # boolean value
```
```
True
```

In [8]:
```
print(4 + 5j)            # complex value
```
```
(4+5j)
```

In [9]:
```
print([3, 5, 'text'])     # list values
```
```
[3, 5, 'text']
```

In [10]:
```
print(3242, True, [34, 5, 5], "text", sep=' *** ')   # print mix values
```
```
3242 *** True *** [34, 5, 5] *** text
```

In [11]:
```
print('Python ', end='')
print('3')
```
```
Python 3
```

In [ ]:
```

```

In [12]:
```
print(dir(__builtins__))
```
```
['ArithmeticError', 'AssertionError', 'AttributeError', 'BaseException', 'Blo
ckingIOError', 'BrokenPipeError', 'BufferError', 'BytesWarning', 'ChildProces
sError', 'ConnectionAbortedError', 'ConnectionError', 'ConnectionRefusedErro
r', 'ConnectionResetError', 'DeprecationWarning', 'EOFError', 'Ellipsis', 'En
vironmentError', 'Exception', 'False', 'FileExistsError', 'FileNotFoundErro
r', 'FloatingPointError', 'FutureWarning', 'GeneratorExit', 'IOError', 'Impor
tError', 'ImportWarning', 'IndentationError', 'IndexError', 'InterruptedErro
r', 'IsADirectoryError', 'KeyError', 'KeyboardInterrupt', 'LookupError', 'Mem
oryError', 'ModuleNotFoundError', 'NameError', 'None', 'NotADirectoryError',
'NotImplemented', 'NotImplementedError', 'OSError', 'OverflowError', 'Pending
DeprecationWarning', 'PermissionError', 'ProcessLookupError', 'RecursionErro
r', 'ReferenceError', 'ResourceWarning', 'RuntimeError', 'RuntimeWarning', 'S
topAsyncIteration', 'StopIteration', 'SyntaxError', 'SyntaxWarning', 'SystemE
rror', 'SystemExit', 'TabError', 'TimeoutError', 'True', 'TypeError', 'Unboun
dLocalError', 'UnicodeDecodeError', 'UnicodeEncodeError', 'UnicodeError', 'Un
icodeTranslateError', 'UnicodeWarning', 'UserWarning', 'ValueError', 'Warnin
g', 'WindowsError', 'ZeroDivisionError', '__IPYTHON__', '__build_class__', '_
_debug__', '__doc__', '__import__', '__loader__', '__name__', '__package__',
'__spec__', 'abs', 'all', 'any', 'ascii', 'bin', 'bool', 'breakpoint', 'bytea
rray', 'bytes', 'callable', 'chr', 'classmethod', 'compile', 'complex', 'copy
right', 'credits', 'delattr', 'dict', 'dir', 'display', 'divmod', 'enumerat
e', 'eval', 'exec', 'execfile', 'filter', 'float', 'format', 'frozenset', 'ge
t_ipython', 'getattr', 'globals', 'hasattr', 'hash', 'help', 'hex', 'id', 'in
put', 'int', 'isinstance', 'issubclass', 'iter', 'len', 'license', 'list', 'l
ocals', 'map', 'max', 'memoryview', 'min', 'next', 'object', 'oct', 'open',
'ord', 'pow', 'print', 'property', 'range', 'repr', 'reversed', 'round', 'run
file', 'set', 'setattr', 'slice', 'sorted', 'staticmethod', 'str', 'sum', 'su
per', 'tuple', 'type', 'vars', 'zip']
```

Python Basic Data Types

In [13]:
```
# integer, float, complex
# boolean
# string
```

In [14]:
```
# int(), float(), complex(), bool(), str()
```

Python Integer

In [15]:
```
x = 5
```

In [16]:
```
type(x)
```

Out[16]:
```
int
```

In [17]:
```
print(dir(x))
```

```
['__abs__', '__add__', '__and__', '__bool__', '__ceil__', '__class__', '__del
attr__', '__dir__', '__divmod__', '__doc__', '__eq__', '__float__', '__floor_
_', '__floordiv__', '__format__', '__ge__', '__getattribute__', '__getnewargs
__', '__gt__', '__hash__', '__index__', '__init__', '__init_subclass__', '__i
nt__', '__invert__', '__le__', '__lshift__', '__lt__', '__mod__', '__mul__',
'__ne__', '__neg__', '__new__', '__or__', '__pos__', '__pow__', '__radd__',
'__rand__', '__rdivmod__', '__reduce__', '__reduce_ex__', '__repr__', '__rflo
ordiv__', '__rlshift__', '__rmod__', '__rmul__', '__ror__', '__round__', '__r
pow__', '__rrshift__', '__rshift__', '__rsub__', '__rtruediv__', '__rxor__',
'__setattr__', '__sizeof__', '__str__', '__sub__', '__subclasshook__', '__tru
ediv__', '__trunc__', '__xor__', 'as_integer_ratio', 'bit_length', 'conjugat
e', 'denominator', 'from_bytes', 'imag', 'numerator', 'real', 'to_bytes']
```

In [ ]:

In [18]:
```
long_value = 3534636346346346346326236262
```

In [19]:
```
type(long_value)
```

Out[19]:
```
int
```

In [ ]:

In [20]:
```
value = int(32456)
```

In [21]:
```
print(value)
```

```
32456
```

In [ ]:

Floating Point Data Type

```
In [22]: point = 4322.4234
```

```
In [23]: type(point)
```

```
Out[23]: float
```

```
In [24]: data = float(324444.43354)
```

```
In [25]: print(data)
```

```
324444.43354
```

Complex Data Type

```
In [26]: c1 = 3 + 7j
```

```
In [27]: type(c1)
```

```
Out[27]: complex
```

```
In [ ]:
```

```
In [28]: c2 = complex(34 + 25j)
```

```
In [29]: print(c2)
```

```
(34+25j)
```

```
In [ ]:
```

Boolean Data Type

```
In [30]: b = True
```

```
In [31]: remarks = bool(False)
```

```
In [ ]:
```

Python Keywords

```
In [32]: import keyword
In [33]: keyword.iskeyword('lambda')

Out[33]: True

In [34]: print(keyword.kwlist)

         ['False', 'None', 'True', '__peg_parser__', 'and', 'as', 'assert', 'async',
         'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'excep
         t', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda',
         'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with',
         'yield']

In [35]: len(keyword.kwlist)

Out[35]: 36
```

System Module

```
In [36]: import sys

In [37]: sys.version_info

Out[37]: sys.version_info(major=3, minor=9, micro=7, releaselevel='final', serial=0)

In [38]: sys.version

Out[38]: '3.9.7 (default, Sep 16 2021, 16:59:28) [MSC v.1916 64 bit (AMD64)]'

In [ ]:
```

Other Built-in Functions

```
In [39]: id?

         Signature: id(obj, /)
         Docstring:
         Return the identity of an object.

         This is guaranteed to be unique among simultaneously existing objects.
         (CPython uses the object's memory address.)
         Type:      builtin_function_or_method
```

```
In [40]: hex?

Signature: hex(number, /)
Docstring:
Return the hexadecimal representation of an integer.

>>> hex(12648430)
'0xc0ffee'
Type:      builtin_function_or_method
```

```
In [41]: id(5)

Out[41]: 2412589771184
```

```
In [42]: hex(id(5))

Out[42]: '0x231b98e69b0'
```

```
In [43]: test_value = 5
```

```
In [44]: hex(id(test_value))

Out[44]: '0x231b98e69b0'
```

```
In [ ]:
```

```
In [45]: issubclass?

Signature: issubclass(cls, class_or_tuple, /)
Docstring:
Return whether 'cls' is a derived from another class or is the same class.

A tuple, as in ``issubclass(x, (A, B, ...))``, may be given as the target to
check against. This is equivalent to ``issubclass(x, A) or issubclass(x, B)
or ...`` etc.
Type:      builtin_function_or_method
```

```
In [46]: issubclass(bool, int)

Out[46]: True
```

```
In [ ]:
```

```
In [47]: # help(int)
```

```
In [ ]:
```

```
In [48]: # Python Built-in Functions
         # print(), help(), dir(), type(), len(), id(), issubclass(), hex()

         # int(), float(), complex(), bool()
         # str()
```

```
In [ ]:
```

```
In [49]: # @mrizwanse
```

Happy Learning :)