Python Data Types
Built-in Data Types
In programming, data type is an important concept.

Variables can store data of different types, and different types can do different things.

Python has the following data types built-in by default, in these categories:

Text Type: str
Numeric Types: int, float, complex
Sequence Types: list, tuple, range
Mapping Type: dict
Set Types: set, frozenset
Boolean Type: bool
Binary Types: bytes, bytearray, memoryview
None Type: NoneType
Getting the Data Type
You can get the data type of any object by using the type() function:

ExampleGet your own Python Server

Print the data type of the variable x:

```
x = 5
print(type(x))
```

Setting the Data Type
In Python, the data type is set when you assign a value to a variable:

Example	Data Type	Try it
x = "Hello World"	str	
x = 20	int	
x = 20.5	float	
x = 1j	complex	
x = ["apple", "banana", "cherry"]	list	
x = ("apple", "banana", "cherry")	tuple	
x = range(6)	range	
x = {"name" : "John", "age" : 36}	dict	
x = {"apple", "banana", "cherry"}	set	
x = frozenset({"apple", "banana", "cherry"})	frozenset	
x = True	bool	
x = b"Hello"	bytes	
x = bytearray(5)	bytearray	
x = memoryview(bytes(5))	memoryview	
x = None	NoneType	

Setting the Specific Data Type
If you want to specify the data type, you can use the following constructor functions:

Example	Data Type	Try it
x = str("Hello World")	str	
x = int(20)	int	
x = float(20.5)	float	
x = complex(1j)	complex	
x = list(("apple", "banana", "cherry"))	list	
x = tuple(("apple", "banana", "cherry"))	tuple	
x = range(6)	range	
x = dict(name="John", age=36)	dict	
x = set(("apple", "banana", "cherry"))	set	
x = frozenset(("apple", "banana", "cherry"))	frozenset	
x = bool(5)	bool	
x = bytes(5)	bytes	
x = bytearray(5)	bytearray	
x = memoryview(bytes(5))	memoryview	

Data types
Data types are the classification of data items. Data types represents a kind of value which determines what can be done to that data.

What are the different types of data in Python?

Data Types	Examples	Explanation	Mutable/Immutable?
Strings	"Hello!", "23.34"	Text - anything between " " becomes string	Immutable
Integers	5364	Whole numbers	Immutable
Floats	3.1415	Decimal Numbers	Immutable
Booleans	True, False	Truth values that represent Yes/No	Immutable
Lists	[1,2,3,4,5]	A collection of data, sits between [ ]	Mutable
Tuples	(1,2,3,4,5)	A collection of data, sits between ( )	Immutable
Dictionaries	{"a":1, "b":2, "c":3}	A collection of data, sets between { }	Mutable

Type conversion
To convert variables from one type to another (i.e. integers to floats), we use type conversions as follows:

Data Type	Syntax
strings	str()
integer	int()
floats	float()
lists	list()

Python Casting
Specify a Variable Type
There may be times when you want to specify a type on to a variable. This can be done with casting. Python is an object-oriented language, and as such it uses classes to define data types, including its primitive types.

Casting in python is therefore done using constructor functions:

int() - constructs an integer number from an integer literal, a float literal (by removing all decimals), or a string literal (providing the string represents a whole number)
float() - constructs a float number from an integer literal, a float literal or a string literal (providing the string represents a float or an integer)
str() - constructs a string from a wide variety of data types, including strings, integer literals and float literals
ExampleGet your own Python Server
Integers:

```
x = int(1)   # x will be 1
y = int(2.8) # y will be 2
z = int("3") # z will be 3
```

ExampleGet your own Python Server
Floats:

```
x = float(1)     # x will be 1.0
y = float(2.8)   # y will be 2.8
z = float("3")   # z will be 3.0
w = float("4.2") # w will be 4.2
```

ExampleGet your own Python Server
Strings:

```
x = str("s1") # x will be 's1'
y = str(2)    # y will be '2'
```

```
z = str(3.0)  # z will be '3.0'
```