Data Types in Python

In Python, **data types** define the kind of value a variable can hold.
They help the interpreter understand how the data should be stored and what operations can be performed on it.

Python has several built-in data types. These are categorized as:

- **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`

🔷 Mutable vs Immutable Types

- **Mutable Types**: `list`, `dict`, `set`, `bytearray`
- **Immutable Types**: `int`, `float`, `str`, `tuple`, `frozenset`, `bool`, `bytes`

Immutable objects cannot be changed after creation.

🔷 `type()` function

You can check the data type of a variable using the `type()` function:

```
type(variable_name)
```

```
number = 10
pi = 3.14
my_string = "Hello"
print(type(number))     # <class 'int'>
print(type(pi))         # <class 'float'>
print(type(my_string))  # <class 'str'>

```

```
<class 'int'>
<class 'float'>
<class 'str'>
```

```
# Examples of built-in data type:
text = "Hello, Python!"              # str
print(text)
print(type(text))
```

```
Hello, Python!
<class 'str'>
```

```
integer = 42                         # int
print(integer)
print(type(integer))
```

```
42
<class 'int'>
```

```
floating = 3.14                      # float
print(integer)
print(type(integer))
```

```
42
<class 'int'>
```

```
my_list = [1,2,3.5,'Pyshaala']    # list
print(my_list)
print(type(my_list))
```

```
[1, 2, 3.5, 'Pyshaala']
<class 'list'>
```

🔷 None Data Type

- None signifies that a variable or expression does not currently hold a meaningful value.

- It representing the absence of a value or a null object.
- It is distinct from an empty string (""), the boolean False, or the numerical 0
- `Default Return Value:`
- Functions in Python that do not explicitly include a return statement implicitly return None

🔷 isinstance() function

- `isinstance(obj, class)` checks if an object is an instance of a particular class/type.

```
num = 'python'
print(type(num))             # <class 'int'>
print(isinstance(num, int))  # True
```

```
<class 'str'>
False
```

🔷 Type Casting

1. Type casting allows you to convert data from one type to another.
2. Convert data types explicitly if needed using casting functions.

Functions for casting:

- `int()`, `float()`, `str()`, `list()`, `tuple()`, `dict()`, etc.

```
number = 5
my_num = str(number)
print(my_num)
print(type(my_num))
print(type(number))
```

```
5
<class 'str'>
<class 'int'>
```

```
name = '41'
print(type(name))
print(int(name))
```

```
<class 'str'>
41
```

```
result = int("5")            # Converts string to integer
total_bill = float(3)        # Converts int to float
my_string_number = str(42)   # Converts int to string
my_list = list((1, 2, 3))    # Converts tuple to list
my_tupal = tuple([1, 2, 3])  # Converts list to tuple
```

```
print(type(total_bill))

```

```
<class 'float'>
```

```
# print(type('5'))
number = int('5')
print(type(number))
```

```
my_name = '1'
num_1 = int(my_name)
```

🔷 Do's and Don'ts

✅ **Do:**

```
age = 25
price = 99.99
```

❌ **Don't:**

```
age = "twenty five"  # Unless you specifically want a string
```

Real-Life Example

Imagine you're developing a small app for a computer shop:

- `item_name` → **str**
- `item_price` → **float**
- `item_quantity` → **int**
- `is_available` → **bool**

Example:

```
1
2 item_name = "Mouse"
3 item_price = 120.50
4 item_quantity = 15
5 is_available = True
6
7 print("Item:", item_name)
8 print("Price per item:", item_price)
9 print("Quantity in stock:", item_quantity)
10 print("Available:", is_available)
11
```

```
Item: Mouse
Price per item: 120.5
Quantity in stock: 15
Available: True
```

Practice Exercises

1. Create variables for your name, age, height, and whether you like Python.
2. Print each variable and its data type.
3. Convert your height into an integer and print it.
4. Create a list of 5 fruits and print its type.

Summary

1. What is Data Types
2. Different available built-in data types in python
3. Mutable vs Immutable Types
4. type() function
5. None Data Type
6. isinstance() function
7. Type Casting
8. Real-Life Example
9. Practice Exercises