Computational Thinking and Programming - 1

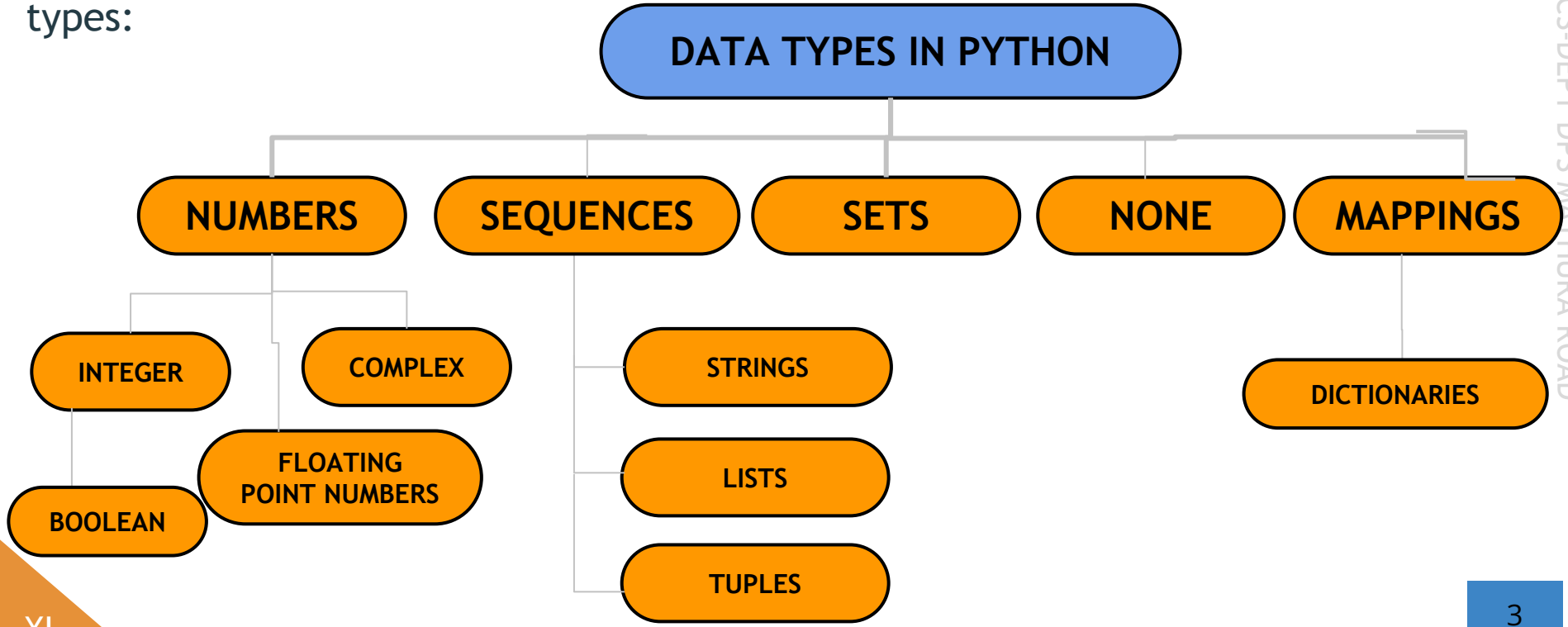
Python - Data Types

Standard Data Types of Python

Data can be of many types viz. Character, integer, real, string etc. Numbers without fractions indicate integer data, number with fractions represent real data, True and False indicate boolean data, anything enclosed within quotes indicate string data in Python. Since the data to be dealt with are of many types, a programming language must be capable to handle all types of data.

Different Data Types in Python

Every value in Python has a datatype. Python has the following built-in core data types:



Data Types in Python - Number

Number data types stores numeric values only. It is further classified into `int`, `float and complex`.

Type/Class	Description	Examples
`int`	Integer numbers	-12,-3,0,99,370
`float`	Real or floating numbers	-8.09,90.87,8E+9,7.33E-9
`complex`	Complex numbers	8 - 9j, 2 + 3j

Boolean data type (`bool`) is a subtype of integer. It is a unique data type, consisting of two constants, True and False. Boolean **`True`** is non-zero, non-null and non-empty and Boolean **`False`** is the value zero.

Number Data Type

int - These are whole numbers, having no fractional parts. They are numeric values with no decimal point. They can be positive or negative.

float - A number containing a decimal point or an exponent sign represents float data type. The float data type can include both positive and negative floating point numbers. The fractional number can be written in two forms:

- Fractional form (normal decimal notation) e.g. 3.2901, 0.009 etc.
- Exponent Notation e.g. 3.50062E09, 0.43E-02 etc.

They have two advantages over integers, they represent values between the integers and they represent larger range of values. But they also suffer from one disadvantage, that the floating point operations are usually slower than integer operations. In Python, they have a **precision of 15 digits** (double precision).

Number Data Type (contd…)

complex - Mathematically, a complex number is a number of the form A+Bi, where A represents the real part and i is the imaginary number. The real and imaginary part of a complex number are floating point number.

Python represents complex number using complex data type by specifying the number in the form **real + imag j** i.e the imaginary part of the number in Python is represented by j (instead of i).

For example for Complex data, 3 + 4j , 3 denotes the real part and 4 denotes the imaginary part. The real and imaginary part of the complex number is stored as a float.

Number Data Type (contd...)

The range of the numbers represented through the numeric data type is given below:

Data Type	Range
int (Integer)	An unlimited range, subject to the memory availability
float (Floating point numbers)	An unlimited range, subject to the memory availability
complex (Complex numbers)	Same as the floating point numbers as the real and imaginary parts are float values
bool (Boolean)	Only two values - True and False

Sequence Data Types

A Python sequence is an ordered collection of items where each item is indexed by an integer. There are three Sequence data types in Python, namely:

- String
- Tuple
- Lists

String Data Type

A **string data type** lets you hold string data i.e, any number of valid characters enclosed in a set of quotation marks. A string can hold any type of known characters ie., letters, numbers and special characters.

A Python string is a **sequence of characters** and each character can be individually accessed using its index. Strings are stored as individual characters in contiguous locations, with two way index for each location.

➢ **0, 1, 2,.... In the forward direction**

➢ **-1, -2, -3.... In the backward direction**

For example : "computer science", 'DPS MATHURA ROAD'

List Data Type

List represents a list of comma separated values of any data type between square brackets ([ ]).

```
>>> L=[1,2,3,4,5]
>>> print(L)
[1, 2, 3, 4, 5]
>>> type(L)
<class 'list'>
```

Tuple Data Type

A Tuple is a ordered collection of elements that cannot be changed i.e. they are not modifiable (immutable). Tuples are group of comma separated values of any data type within parentheses().

```
>>> tuple1=(1,2,3,4,5,6,7)
>>> tuple1
(1, 2, 3, 4, 5, 6, 7)
>>> print(tuple1)
(1, 2, 3, 4, 5, 6, 7)
```

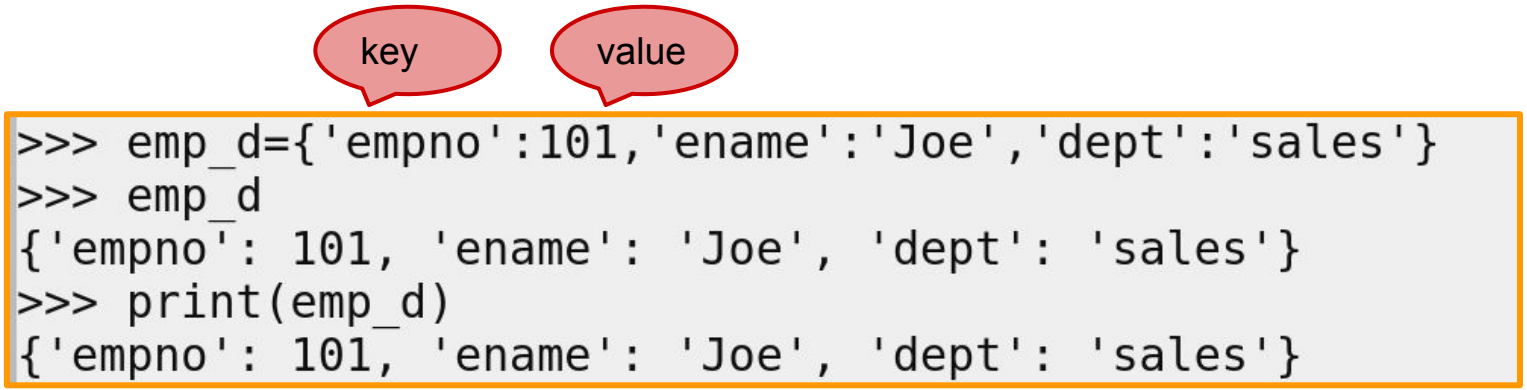
NONE

None is a special data type with a single value. It is used to signify the absence of value in a situation. None supports no special operations, and it is neither False nor 0 (zero).

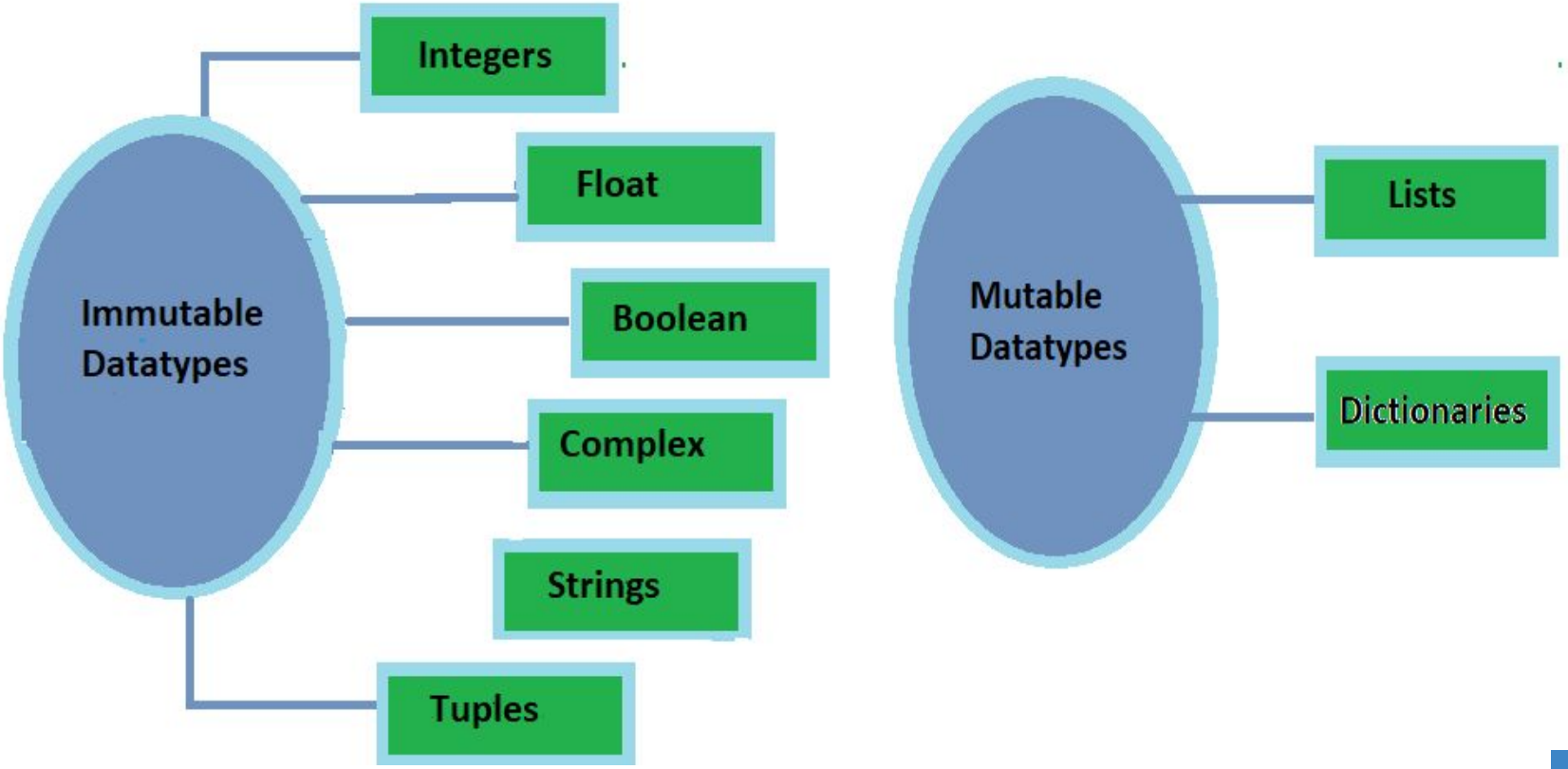
```
>>> a=None
>>> type(a)
<class 'NoneType'>
```

Mapping - Dictionary Data Type

Mapping is an unordered data type in Python. Currently, there is only one standard mapping data type in Python called dictionary. A dictionary is an unordered set of comma separated **key:value** pairs, within { }, with the requirement that within a dictionary, no two keys can be the same (there are unique keys within a dictionary)



Mutable and Immutable Data Types



Mutable and Immutable Data Types

- Variables whose values can be changed after they are created and assigned are called **mutable**.
- Variables whose values cannot be changed after they are created and assigned are called **immutable**. In Python, the following data types are **immutable** :

integers, floating point numbers, string, boolean, tuples

- When an attempt is made to update the value of an immutable variable, the old variable is destroyed and a new variable is created by the same name in memory.
- Python data types can be classified into mutable and immutable.

Mutable and Immutable Data Types

Consider there are three variables as shown:

```
>>> a=10
>>> b=a
>>> c=10
>>> print(id(a))
1962437952
>>> print(id(b))
1962437952
>>> print(id(c))
1962437952
```

Variable names are only indicators or references to value objects, ie. data values.The variables themselves do not store any values, they are not storage containers. It appears from the above statements that the value of the variables are changing, but it is not so.

Mutable and Immutable Data Types

In fact the variable names are made to refer to new immutable integer objects. Changing in place means changing or modifying the same value in the same memory location.

```
>>> a=78
>>> b=90
>>> c=12
>>> print(id(a))
1962440128
>>> print(id(b))
1962440512
>>> print(id(c))
1962438016
```

Each time you change a variable's value, its reference memory address also changes. These variables are NOT STORAGE CONTAINERS, they are not fixed memory address where the value changes. Hence they are IMMUTABLE.

Mutable and Immutable Data Types

Mutable data types

There are three mutable data types in Python: lists, dictionaries and sets.

```
>>> list1=[1,2,3,4,5,6,7]
>>> list1
[1, 2, 3, 4, 5, 6, 7]
>>> print(id(list1))
2732764117320
>>> list1[0]
1
>>> list1[0]=400
>>> list1
[400, 2, 3, 4, 5, 6, 7]
>>> print(id(list1))
2732764117320
```

Thus we see from this code that values are modified but still memory address of mutable data type remains same.

THANK YOU!

DEPARTMENT OF COMPUTER SCIENCE