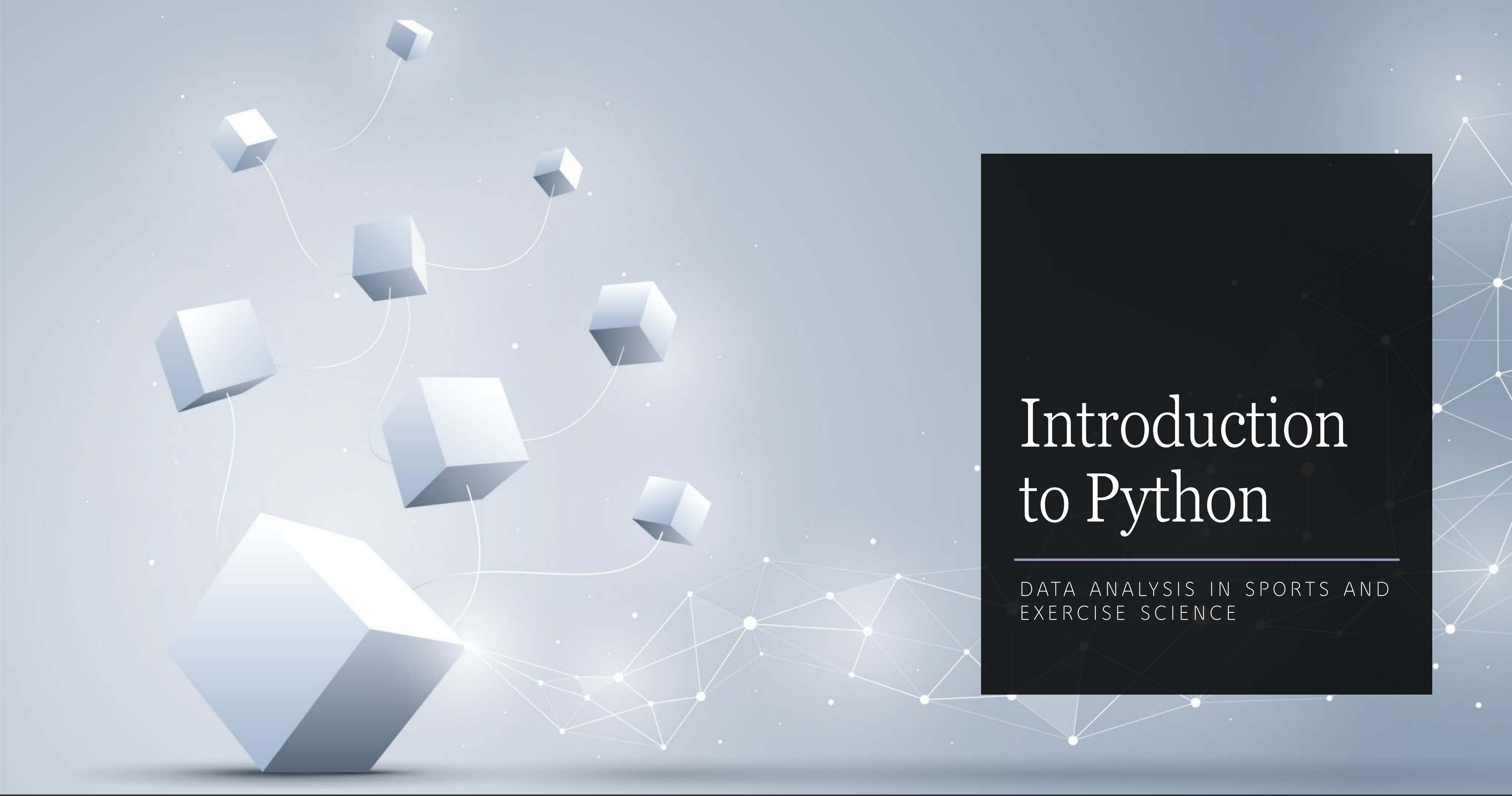


Introduction to Python

DATA ANALYSIS IN SPORTS AND EXERCISE SCIENCE

Outline of the course

- ➢Introduction to python
- ➢Basic operators
- ➢Libraries in python and their installation
- ➢Python data structure
- ➢Control flow & loops

What is Python

- Python is a high-level, general-purpose and a very popular programming language.
- Python programming language (latest Python 3) is being used in web development, Machine Learning applications, along with all cutting edge technology in Software Industry.

Why learn Python

- Designed for clear, logical code that is easy to read and learn.
- There are lot of libraries and frameworks written in python allowing users to apply Python in wide variety of tasks.
- Great documentation online

  https://docs.python.org/3/

Installation of Python

- Download Anaconda distribution from https://www.anaconda.com/products/distribution.

For this course we will work on Jupyter Notebook.

Refer to the following links for installation steps for different OS:

https://docs.anaconda.com/anaconda/install/windows/ (WINDOWS)

https://docs.anaconda.com/anaconda/install/mac-os/ (MAC)

https://docs.anaconda.com/anaconda/install/linux/ (LINUX)

Basic operators in python

- Arithmetic Operators
- Comparison (Relational) Operators
- Assignment Operators
- Logical Operators

Arithmetic operators

Operator	Description	Syntax
+	Addition: adds two operands	x + y
–	Subtraction: subtracts two operands	x – y
*	Multiplication: multiplies two operands	x * y
/	Division (float): divides the first operand by the second	x / y
//	Division (floor): divides the first operand by the second	x // y
%	Modulus: returns the remainder when the first operand is divided by the second	x % y
**	Power: Returns first raised to power second	x ** y

Comparison (Relational) Operators

Operator	Description	Syntax
>	Greater than: True if the left operand is greater than the right	x > y
<	Less than: True if the left operand is less than the right	x < y
==	Equal to: True if both operands are equal	x == y
!=	Not equal to – True if operands are not equal	x != y
>=	Greater than or equal to True if the left operand is greater than or equal to the right	x >= y
<=	Less than or equal to True if the left operand is less than or equal to the right	x <= y

Assignment Operators

Operator	Description	Syntax
=	Assign value of right side of expression to left side operand	x = y + z
+=	Add AND: Add right-side operand with left side operand and then assign to left operand	a+=b a=a+b
-=	Subtract AND: Subtract right operand from left operand and then assign to left operand	a-=b a=a-b
*=	Multiply AND: Multiply right operand with left operand and then assign to left operand	a*=b a=a*b
/=	Divide AND: Divide left operand with right operand and then assign to left operand	a/=b a=a/b
%=	Modulus AND: Takes modulus using left and right operands and assign the result to left operand	a%=b a=a%b
//=	Divide(floor) AND: Divide left operand with right operand and then assign the value(floor) to left operand	a//=b a=a//b
**=	Exponent AND: Calculate exponent(raise power) value using operands and assign value to left operand	a**=b a=a**b

Logical Operators

Operator	Description	Syntax
and	Logical AND: True if both the operands are true	x and y
or	Logical OR: True if either of the operands is true	x or y
not	Logical NOT: True if the operand is false	not x

Libraries for data analytics

- Numpy
- Pandas
- Matplotlib
- Scipy
- Scikit-learn

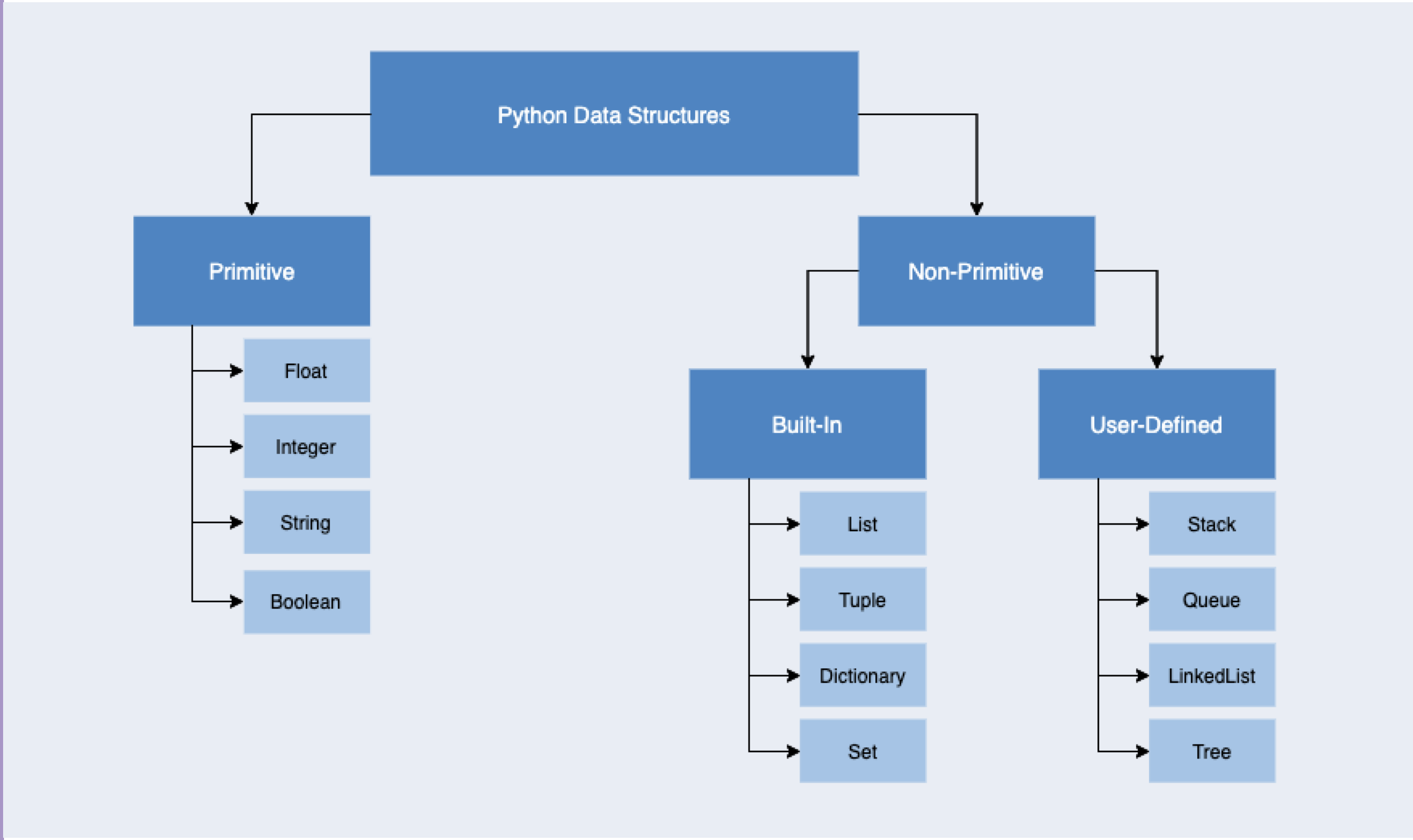
Installation of libraries

- Open anaconda navigator
- Go to Environments-> in Search packages enter the library to be installed.
- Go on and install the required libraries

OR

- Open anaconda prompt-> enter command “conda install pandas” or “pip install pandas”
- Pip and conda are package manager for any software (installation, upgrade and uninstallation)

Python data structures



*Source:https://miro.medium.com/max/2190/1*2JFd94q0vzsEcr-LB-220g.png*

1. Numbers(int, float)

➢Types of Numbers in Python :Integer and float

Integers are whole numbers, positive or negative ex: +4,-7,9 etc

Float have a decimal point in them, or use an exponential (e) to define the number. Ex: 1.2,-0.5

➢Basic Arithmetic

➢Variable Assignment in Python

Variable name creating rules

- Names can not start with a number.
- There can be no spaces in the name, use _ instead. Can't use any of these symbols :'",<>/?|\()!@#$%^&*~-+
- It's considered best practice that names are lowercase.
- Avoid using the characters 'l' , 'O' , or 'I' as single character variable names.
- Avoid using words that have special meaning in Python like "list" and "str"

2. String

Strings are sequence of character, using the syntax of either single quotes or double quotes.

Ex: “Hello” ,“I didn’t had coffee.”

H	e	l	l	o
0(0)	1(-4)	2(-3)	3(-2)	4(-1)

- Creating Strings
- Printing Strings
- String Indexing and Slicing
- String Properties
- String Methods
- Print Formatting

3. Lists

Lists are ordered sequences that can hold variety of object types.

They use [] bracket and comma to separate objects. Ex:[1,2,3,4,5]

- Creating lists
- Indexing and Slicing Lists
- Basic List Methods

4. Dictionaries

Dictionaries are unordered mapping for storing objects. In lists its ordered sequence, dictionaries have key-value pairing.

Key-value pair allows users to grab objects quickly without the need to know the index location.

Uses curly braces and colons to signify the keys and the associated values.

{‘key1’:’value1’,’key2’:’value2’}

- ➢Constructing a Dictionary
- ➢Accessing objects from a dictionary
- ➢Basic Dictionary Methods

When to use list/dictionary

- **Dictionaries:** Object is retrieved by key name.

Unordered and cannot be sorted.

- **Lists:** Objects retrieved by location

Ordered sequence and can be indexed or sliced.

5. Tuples

Tuples are very similar to lists. However, they have one key difference that is **immutability**. Once an element is inside the tuple it cannot be re-assigned.

Tuple uses parenthesis: **(1,2,3)**

- Constructing Tuples
- Basic Tuple Methods
- Immutability
- When to use Tuples

6. Control flow

When we want to execute certain piece of code when certain condition has been met.
For ex: if my cat is hungry(condition), I will feed my cat(action).

Control flow syntax makes use of colons and identation (whitespace).

- If
- elif
- else
- For loop
- While loop

Syntax of different control flows

if	**if some_condition**
	    #execute some code

if/else	if some_condition
	    #execute some code
	else:
	    #do something else

if/elif/else	if some_condition
	    #execute some code
	elif some_other_condition
	    #execute something different
	else:
	    #do something else

for	my_lists=[1,2,3]
	for item_name in my_lists
	      print(item_name)

while	while some_Boolean_condition
	        #do_something