Python Day 1:

1) **What is Python?**
    1) Interpreted, interactive and Object-Oriented
    2) It supports the use of modules and packages
    3) It is very useful in RAD(Rapid Application Development)
2) **Meanings of:**
    1) Interpreted:
        1) Processed at runtime by the interpreter
        2) Code does not need to be compiled before execution
    2) Interactive:
        1) Python prompt can interact with the interpreter directly
    3) Object-Oriented:
        1) It supports OOPs
    4) Beginner Friendly
    5) Standard Library:
        1) Very portable and cross-platform compatible- eg: Unix, Windows, MacOs
    6) GUI(Graphical-user-Interface):
        1) Supports GUI applications and these application can be created and ported to many system calls
    7) Scalable:
        1) Provides better structure and support to large programs than shell scripting
    8) Portable:
        1) Can run on variety of h/w platforms and has the interface on all the platforms
    9) Extendable(Extensions):
        1) Low-level modules that can be added to Python Interpreter
        2) These modules enables us to customise our programming tools
    10) Databases:
        1) Python supports mostly all commercial databases- MySQL, PL/SQL, Mongo etc.
3) **History:**
    1) Developed by Guido van Possum in 1980
    2) Versions:
        1) Python 1.0 -> lambdas, map, filters, reduce
        2) Python 2.0 -> list, garbage collection

3) Python 3.0(Python 3k) -> rectifies the flaws of fundamental language

4) How We can run:

1) Directly run using command prompt(Python shell)
    1) Useful for executing only single set of instruction
2) Using script files:
    1) It can be used to write block of codes
    2) Run: **python <file-name>.py**
3) Using IDE:
    1) Pycharm IDE, VsCode

5) Python Keywords:

1) Are special reserved words , that has special meaning for the interpreter
    1) Eg: break, continue, for, def etc.
2) We cannot use these keywords as variables

6) Python Identifiers:

1) Name given to entities-> class, functions, variables etc
2) Best practices while naming:
    1) Class name should start with **uppercase** and all other identifiers should start with **lowercase**
    2) Private identifiers should start with **underscore(_)**
    3) Magic Methods: use double underscore(__) and don't use them anywhere else
        1) Eg: __init__, __len__, __add__
    4) Always prefer to use longer variable names > length 1
        1) Eg: index=1 is better than I=1
    5) To combine words in identifiers use underscore(_)
        1) Eg: get_user_details
    6) Use camel-casing for variables:
        1) Eg: userName

7) Python Variables:

1) Dynamically-typed: we don't need to specify the data-type. The interpreter automatically interprets it during runtime

8) Commenting:

1) Single-line:
    1) #
2) Multiple-line:
    1) #-> in multiple lines
    2) “”””” -> doc-string

9) Operators:

1) Arithmetic Operators:
    1) +, -, *, /, %, //, **
2) Comparison Operators:
    1) >, <, >=, <=, ==, !=
3) Logical Operators:
    1) And, or, not

10) Data-Types:

1) Based on Mutability:
    1) Mutable:
        1) List, Dictionary, Sets
    2) Immutable:
        1) Number, String, Tuples
2) Based on functionality:
    1) Numeric -> int, float, complex
    2) Sequence -> list, tuple, string
        1) List:
            1) Similar to ArrayList of Java
            2) Dynamic size -> no size is required to instantiate a list
            3) Mutable
            4) []
        2) Tuple:
            1) Similar to list only difference it is immutable
            2) Faster than lists because they cannot change dynamically
            3) ()
        3) String:
            1) Immutable
    3) Boolean:
        1) True, False
    4) Mappings: Dictionary, Sets
        1) Sets:
            1) Mutable-> but we cannot change one particular index but a set as a whole
            2) **Unordered** collection of **unique** elements
            3) {}
        2) Dictionary:
            1) Unordered collection of key-value pairs
            2) Used when we have huge amount of data

3) Optimised for data-retrieval which is done based on the key
4) {} -> key:value
5) Key and value can be of any type

3) Difference between List and Dictionary:

List	Dictionary
Elements are in order	Elements are unordered
List contains data types like integers, string , Objectsetc	It is used to store data values like a map
They can be accessed via numeric index. Eg: list[0]	Elements are accessed using key values

11) Control Structures:

1) Sequential:
   1) Everything happens in proper sequence I.e. as mentioned
2) Alternative/Branching:
   1) If-elif-else
3) Looping/Iterative:
   1) For, While, Do-While
   2) break, continue, pass

12) Input & Output:

1) In python we can use the input() command to take user-input

13) String Functions:

1) Strings in python are immutable
2) Can be created using single, double or triple quotes
3) Can be created using the str() function
4) String indexing -> starts from 0
5) Concatenation -> + or *
6) Length -> len()
7) String functions:
   1) strip(Java eq: trim()): removes the leading spaces
   2) upper(Java eq: toUpperCase()): converts string to uppercase

3) lower(Java eq: toLowerCase()): converts string to lowercase
4) replace(Java eq: -> no direct function -> we can use StringBuilder): replaces the occurrence of the string we want
5) split(Java eq: .split()): return list of string after separating through the passed delimiter
6) find(Java eq: indexOf()) -> returns the index of first occurrence in the string otherwise -1
7) max -> prints the greatest value in the string based on ASCII positioning
8) Min-> prints the least value in the string based on ASCII positioning

14) **Number Functions:**
1) Type-casting:
    1) int(x), float(x), complex(x)
2) Mathematical Calculations:
    1) Import math
    2) Math.sqrt, math.floor(least-value), math.ceil(greatest-value), math.pow, math.pi

15) Date And Time Functions (reference 1/1/1970 12:00am):
1) Import time
    1) .time(), .localTime(), .ascTime()
2) Import calendar
3) Import datetime