Python notes

❖ Why python?

Python is an interpreted programming language and easy to understand and it will execute line by line and it is easy to debug, and it cannot compile at once.

❖ Data types in python?

In python 7 types of data types:

Text data types: str(string),

numerical data types: int, char, float

sequences data types: list and tuple

mapping data types: dict

set data types: set

Boolean data types: bool

❖ What is list?

List is a mutable data structure and used to store collection of data and it is defined as square brackets [] and it allows to add an element and remove an element and it allows duplicate.

❖ What is a tuple?

Tuple is immutable, ordered collection elements in python and once tuple is created it cannot add, remove and modified its elements. It is defined as parenthesis ().

❖ What is the difference between list and tuple?

List is a mutable and it is defined as square brackets and it allows the add, remove, modify the elements. Tuple is immutable, ordered it is defined as parenthesis () and once tuple is created it cannot add, remove and modify its elements.

❖ What is mutable and immutable?

Mutable can change the object after it's created and immutable cannot change the object.

❖ What is a set?

Sets are an unordered collection of unique elements. defined as flower brackets {} and it is mutable and duplicate elements are automatically removed.

❖ What is dictionary?

Dictionary is a unordered collection of key value pairs and are used to store data in the form of key value pairs, where each key is unique.

❖ What is variables?

Variables are containers for storing data values.

❖ What is decorators?

Decorators is a function that takes another function as an argument and return the function.

❖ What is class?

In OOP, a class is a blueprint or template that defines the properties (attributes) and behaviors (methods) that objects of that class will have. It serves as a blueprint for creating objects.

❖ What is an object?

An object is a instance of a class so that class can have attributes and function and attributes gives an information of a class it’s called as data and with the help of object we can access the attributes and function. So that is called objects

❖ What is abstraction?

Abstraction is a hiding the implementation details and showing essential details

❖ What is encapsulation?

Encapsulation is a binding data and function into single entity.so by using access specifiers are public, private, protected so public is accessed by any function in any

class and private is accessed by only class which data and function declared and protected can accessed by only class.

❖ What is inheritance?

Inheritance is a child class inherits the properties of parents class. so all attributes and function in child class can access the parents class .

Type of inheritance:

Single inheritance, multiple inheritance, multi-level inheritance,

❖ What is polymorphism?

Polymorphism is implementing same method in different ways. It enables the same method to be used with objects of different classes, providing flexibility and extensibility

What is slicing in Python?

- As the name suggests, ‘slicing’ is taking parts of.
- Syntax for slicing is **[start : stop : step]**
- **start** is the starting index from where to slice a list or tuple
- **stop** is the ending index or where to stop.
- **step** is the number of steps to jump.
- Default value for **start** is 0, **stop** is number of items, **step** is 1.
- Slicing can be done on **strings, arrays, lists**, and **tuples**.

What is the difference between Python Arrays and lists?

- Arrays in python can only contain elements of the same data types i.e., data type of array should be homogeneous. It is a thin wrapper around C language arrays and consumes far less memory than lists.
- Lists in python can contain elements of different data types i.e., data type of lists can be heterogeneous. It has the disadvantage of consuming large memory.

What is break, continue and pass in Python?

Break	The break statement terminates the loop immediately and the control flows to the statement after the body of the loop.
Continue	The continue statement terminates the current iteration of the statement, skips the rest of the code in the current iteration and the control flows to the next iteration of the loop.
Pass	As explained above, the pass keyword in Python is generally used to fill up empty blocks and is similar to an empty statement represented by a semi-colon in languages such as Java, C++, JavaScript, etc.

```
pat = [1, 3, 2, 1, 2, 3, 1, 0, 1, 3]
for p in pat:
  pass
  if (p == 0):
    current = p
    Break
  Elif (p % 2 == 0):
    continue
  print(p)  # output => 1 3 1 3 1
print(current)  # output => 0
```

What are the common built-in data types in Python?

There are several built-in data types in Python. Although, Python doesn't require data types to be defined explicitly during variable declarations type errors are likely to occur if the knowledge of data types and their compatibility with each other are neglected. Python provides type() and isinstance() functions to check the type of these variables. These data types can be grouped into the following categories-

- **None Type:**
  None keyword represents the null values in Python. Boolean equality operation can be performed using these NoneType objects.

Class Name	Description
NoneType	Represents the **NULL** values in Python.

- **Numeric Types:**
  There are three distinct numeric types - **integers, floating-point numbers**, and **complex numbers**. Additionally, **booleans** are a sub-type of integers.

Class Name	Description
int	Stores integer literals including hex, octal and binary numbers as integers
float	Stores literals containing decimal values and/or exponent signs as floating-point numbers
complex	Stores complex numbers in the form (A + Bj) and has attributes: real and imag
bool	Stores boolean value (True or False).

Note: *The standard library also includes* ***fractions*** *to store rational numbers and* ***decimal*** *to store floating-point numbers with user-defined precision.*

- **Sequence Types:**
  According to Python Docs, there are three basic Sequence Types - **lists, tuples,** and **range** objects. Sequence types have the in and not in operators defined for their traversing their elements. These operators share the same priority as the comparison operations.

Class Name	Description
list	Mutable sequence used to store collection of items.
tuple	Immutable sequence used to store collection of items.
range	Represents an immutable sequence of numbers generated during execution.
str	Immutable sequence of Unicode code points to store textual data.

Note: The standard library also includes additional types for processing:
1. **Binary data** such as bytearray bytes memoryview , and
2. **Text strings** such as str.

- **Mapping Types:**

A mapping object can map hashable values to random objects in Python. Mappings objects are mutable and there is currently only one standard mapping type, the ***dictionary***.

Class Name	Description
dict	Stores comma-separated list of **key: value** pairs

- **Set Types:**
  Currently, Python has two built-in set types - **set** and **frozenset**. **set** type is mutable and supports methods like add() and remove(). **frozenset** type is immutable and can't be modified after creation.

Class Name	Description
set	Mutable unordered collection of distinct hashable objects.
frozenset	Immutable collection of distinct hashable objects.

Note: *set is mutable and thus cannot be used as key for a dictionary. On the other hand, frozenset is immutable and thus, hashable, and can be used as a dictionary key or as an element of another set.*

- **Modules:**
  Module is an additional built-in type supported by the Python Interpreter. It supports one special operation, i.e., **attribute access**: mymod.myobj, where mymod is a module and **myobj** references a name defined in m's symbol table. The module's symbol table resides in a very special attribute of the module **__dict__**, but direct assignment to this module is neither possible nor recommended.

- **Callable Types:**
  Callable types are the types to which function call can be applied. They can be **user-defined functions, instance methods, generator functions**, and some other **built-in functions, methods** and **classes**.
  Refer to the documentation at docs.python.org for a detailed view of the **callable types**.

What are negative indexes and why are they used?

- Negative indexes are the indexes from the end of the list or tuple or string.
- **Arr[-1]** means the last element of array **Arr[]**

Explain split() and join() functions in Python?

- You can use **split()** function to split a string based on a delimiter to a list of strings.
- You can use **join()** function to join a list of strings based on a delimiter to give a single string.

What is a lambda function?

A lambda function is an anonymous function. This function can have any number of parameters but, can have just one statement.
In the example, we defined a lambda function(**upper**) to convert a string to its upper case using upper().

What is List Comprehension? Give an Example.

List comprehension is a way to create lists using a concise syntax. It allows us to generate a new list by applying an **expression** to each item in an existing **iterable** (such as a **list** or **range**). This helps us to write cleaner, more readable code compared to traditional looping techniques.

For example, if we have a list of integers and want to create a new list containing the square of each element, we can easily achieve this using list comprehension.

```
a = [2,3,4,5]

res = [val ** 2 for val in a]

print(res)
```

What is the difference between a shallow copy and a deep copy?

Below is the tabular Difference between the Shallow Copy and Deep Copy:

Shallow Copy	Deep Copy
Shallow Copy stores the references of objects to the original memory address.	Deep copy stores copies of the object's value.
Shallow Copy reflects changes made to the new/copied object in the original object.	Deep copy doesn't reflect changes made to the new/copied object in the original object.
Shallow Copy stores the copy of the original object and points the references to the objects.	Deep copy stores the copy of the original object and recursively copies the objects as well.
A shallow copy is faster.	Deep copy is comparatively slower.

What are Iterators in Python?

In Python, iterators are used to iterate a group of elements, containers like a list. Iterators are collections of items and they can be a list, tuples, or a dictionary. Python iterator implements __itr__ and the next() method to iterate the stored elements. We generally use loops to iterate over the collections (list, tuple) in Python.

What are Generators in Python?

In Python, the generator is a way that specifies how to implement iterators. It is a normal function except that it yields expression in the function. It does not implement __itr__ and __next__ method and reduces other overheads as well.

What is Polymorphism in Python?

Polymorphism means the ability to take multiple forms. Polymorphism allows different classes to be treated as if they are instances of the same class through a common interface. This means that a method in a parent class can be overridden by a method with the same name in a child class, but the child class can provide its own specific implementation. This allows the same method to operate differently depending on the object that invokes it. Polymorphism is about overriding, not overloading; it enables methods to operate on objects of different classes, which can have their own attributes and methods, providing flexibility and reusability in the code.

Define encapsulation in Python?

Encapsulation is the process of hiding the internal state of an object and requiring all interactions to be performed through an object's methods. This approach:

- Provides better control over data.
- Prevents accidental modification of data.
- Promotes modular programming.

Python achieves encapsulation through **public**, **protected** and **private** attributes.

How do you do data abstraction in Python?

Data Abstraction is providing only the required details and hides the implementation from the world. The focus is on exposing only the essential features and hiding the complex implementation behind an interface. It can be achieved in Python by using interfaces and abstract classes.

What is slicing in Python?

Python Slicing is a string operation for extracting a part of the string, or some part of a list. With this operator, one can specify where to start the slicing, where to end and specify the step. List slicing returns a new list from the existing list.

Syntax:

substring = s[start : end : step]

What is a Class?

A class is a collection of objects. Classes are blueprints for creating objects. A class defines a set of attributes and methods that the created objects (instances) can have.

Some points on Python class:

- Classes are created by keyword class.
- Attributes are the variables that belong to a class.
- Attributes are always public and can be accessed using the dot (.) operator.

What is an Objects?

An Object is an instance of a Class. It represents a specific implementation of the class and holds its own data.

An object consists of:

- **State:** It is represented by the attributes and reflects the properties of an object.
- **Behavior:** It is represented by the methods of an object and reflects the response of an object to other objects.
- **Identity:** It gives a unique name to an object and enables one object to interact with other objects.

Python Inheritance

Inheritance allows a class (child class) to acquire properties and methods of another class (parent class). It supports hierarchical classification and promotes code reuse.

Types of Inheritance:

1. **Single Inheritance:** A child class inherits from a single parent class.
2. **Multiple Inheritance:** A child class inherits from more than one parent class.
3. **Multilevel Inheritance:** A child class inherits from a parent class, which in turn inherits from another class.

4. **Hierarchical Inheritance:** Multiple child classes inherit from a single parent class.

What is the difference between "is" and "==" in Python?

The "**is**" operator checks if two objects are the same object in memory. It returns True if both objects are identical, meaning they have the same memory address.

On the other hand, the "==" operator checks if two objects have the same value. It returns True if the values of the two objects are equal, regardless of whether they are the same object in memory.

How do you define a function in Python?

Answer: You define a function in Python using the def keyword followed by the function name and a set of parentheses containing parameters.

Explain the difference between return and print in a function.

Answer: return is used to specify the value that a function should produce as its result.

print is used to display information on the console but does not affect the function's return value.

What is a docstring, and why is it used in Python functions?

Answer: A docstring is a string literal used to provide documentation for functions, classes, or modules. It helps users understand the purpose and usage of the function.

Explain the purpose of the lambda keyword in Python?

The lambda keyword is used to create anonymous (nameless) functions, often for simple operations. It returns a function object.

What are recursive functions, and why are they used in Python?

Recursive functions which is defines function can call themselves. They are used to solve problems that can be broken down into smaller, similar subproblems.

What is a decorator in Python, and how is it used to modify the behavior of functions?

A decorator is a function that takes another function as input and returns a modified version of that function. It is used to add functionality to functions without changing their code.