**Question 1.1 : Write the answer of these questions**

1. **Difference between static and dynamic variable**

**Static Variable:** A static variable is a class variable that is shared among all instances of a class. They are defined inside a class but outside any instance methods.

**Dynamic Variable:** A dynamic variable is an instance variable that is unique to each instance of a class. They are usually defined within methods.

class Example:

static\_var = 0 # Static variable

def \_\_init\_\_(self, dynamic\_var):

self.dynamic\_var = dynamic\_var # Dynamic variable

instance1 = Example(10)

instance2 = Example(20)

print(Example.static\_var) # Output: 0

print(instance1.dynamic\_var) # Output: 10

print(instance2.dynamic\_var) # Output: 20

1. **Explain the purpose of pop, popitem, clear() in a dictionary with suitable example**

**pop:** Removes a specified key and returns the corresponding value. If the key is not found, it raises a KeyError unless a default value is provided.

**popitem:** Removes and returns the last (key, value) pair as a tuple. Raises KeyError if the dictionary is empty.

**clear:** Removes all items from the dictionary.

1. **Explain frozen set with suitable examples**

A frozenset is an immutable version of a set. Once created, elements cannot be added or removed.

1. **Differentiate between mutable and immutable data types in python with examples**

**Mutable:** Objects that can be changed after their creation. Example is list and dictionary

**Immutable:** Objects that cannot be changed after their creation. Example is tuple and string

1. **What is an init? Explain with an example.**

It is a constructor method in Python, used to initialize the object's attributes.

1. **Explain docstring with an example.**

A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition. It is used for documentation.

Syntax is “ “ “ I love pizza “ “ “.

1. **What are unit tests in python?**

Unit tests are used to test the functionality of specific sections of code. Python’s built-in unittest module can be used to create and run tests.

1. **What is break, continue and pass in python?**

**break:** Exits the nearest enclosing loop.

**continue:** Skips the rest of the code inside the loop for the current iteration and jumps to the next iteration.

**pass:** Does nothing and can be used as a placeholder.

1. **What is the use of self in python?**

self is used to represent the instance of the class. It is used to access variables that belong to the class.

1. **What are global, protected and private attributes in python?**

**Global:** Variables declared outside of any function or class.

**Protected:** Variables prefixed with a single underscore (\_), meant for internal use only.

**Private:** Variables prefixed with a double underscore (\_\_), meant to be hidden from outside.

1. **What are modules and packages in python?**

**Module:** A single file (or files) that are imported under one import and used.

**Package:** A collection of modules in directories that gives a package hierarchy.

1. **What are lists and tuples? What are key differences between two**

**List:** Mutable, ordered collection of items.

**Tuple:** Immutable, ordered collection of items.

1. **What is an Interpreted language & dynamically typed language?Write 5 differences between them**

| **Interpreted language** | **Dynamically typed** |
| --- | --- |
| Executes code line by line. | Variable types are determined at runtime. |
| Easier to debug. | Harder to debug. |
| Generally slower execution. | Generally faster execution. |
| No need for explicit type declaration | Need for explicit type declaration. |

1. **What are dict and list comprehensions?**

| **Dict comprehension** | **List comprehension** |
| --- | --- |
| Provides a concise way to create dictionaries. | Provides a concise way to create lists |

1. **What are decorators in python? Explain it with an example. Write down its use cases.**

Decorators are functions that modify the behavior of other functions or methods.

1. **How is memory managed in python?**

Python uses automatic memory management, with a garbage collector to manage memory allocation and deallocation.

1. **What is lambda in python? Why is it used?**

A lambda function is a small anonymous function defined with the lambda keyword.

Example is add= lambda x, y: x+y

print(add(5,3)) ## output is 8

1. **Explain the split and join() functions in python.**

Split(): splits a string into list while join() joins a list of strings into a single string.

Example is text= “Hello”

Split\_text = text.split()

print(split\_text) ## Output is [‘H’, ‘e’, ‘l’, ‘l’, ‘o’]

Joined\_text = “ “.join(split\_text)

print(joined\_text) ## output is Hello

1. **What are iterators, iterable & generators in python?**

**Iterable:** An object that can return an iterator.

**Iterator:** An object representing a stream of data.

**Generator:** A type of iterator that is defined using a function with yield statements.

1. **What is the difference between xrange and range in python?**

range() returns a list of numbers while xrange() returns an xrange object(python 2 only). In python 3, range() behaves like xrange().

1. **Pillars of OOPs.**

Encapsulation, Abstraction, Inheritance, Polymorphism

1. **How will you check if a class is a child of another class?**

class Parent:

pass

class Child(Parent):

pass

print(issubclass(Child, Parent)) # Output: True

1. **How does inheritance work in python? Explain all types of inheritance with an example.**

| **Single** | **Multiple** | **Multilevel** | **Hierarchical** | **Hybrid** |
| --- | --- | --- | --- | --- |
| Inherits from one base class | Inherits from multiple base class | Inherits from a derived class | Multiple derived class from one base class | Combination of two or more types of inheritence |

1. **What is encapsulation? Explain it with an example.**

Encapsulation is the building of data and methods that operate on the data within one unit. Example : a class

class Encapsulated:

def \_\_init\_\_(self):

self.\_\_private\_var = 10

def get\_var(self):

return self.\_\_private\_var

obj = Encapsulated()

print(obj.get\_var()) # Output: 10

1. **What is polymorphism? Explain it with an example.**

Polymorphism allows methods to do different things on the object it is acting upon.

class Cat:

def sound(self):

return "Meow"

class Dog:

def sound(self):

return "Bark"

def make\_sound(animal):

print(animal.sound())

cat = Cat()

dog = Dog()

make\_sound(cat) # Output: Meow

make\_sound(dog) # Output: Bark

**Question 1.2 : Identify invalid identifiers and explain why:**

1. Serial\_no. - Valid
2. 1st\_Room - Invalid (Identifiers cannot start with a digit)
3. Hundred$ - Invalid (Identifiers cannot contain special characters like$)
4. Total\_Marks - Valid
5. total-Marks - Invalid (Identifiers cannot contain hyphens)
6. Total Marks - Invalid (Identifiers cannot contain spaces)
7. True - Invalid (Cannot use Python keywords or built-in identifiers)
8. \_Percentag - Valid