**ASSIGNMENT 15**

**TRADITIONAL OOPS PART 1**

**RESEARCH WORK**

**Q1: What is the difference between a class and an object in Python?**

**Ans:** The basic difference between class and object in python is that class is a blueprint,template, or model for creating instances which are called objects. Once a class is created, we can create several objects from it without writing the whole code.

**Q2: Explain encapsulation and data hiding with code example.**

**Ans:** Encapsulation and data hiding is actually the bundling of data and methods into a single unit. It refers to showing the data you want and restricting access objects’ private or protected attributes and methods. Single underscore (\_) is used to define protected attributes or methods and double underscore (\_\_) is used to define private attributes and methods. Getter and setter methods are used to access private and protected attributes and methods.

**EXAMPLE:**

class GmailAccount:

def \_\_init\_\_(self, name, username):

self.name = name # public

self.\_username = username # protected

self.\_\_password = "n123" # private

# Getter method for protected attribute

def get\_username(self):

return self.\_username

# Setter method for protected attribute

def set\_username(self, new\_username):

if isinstance(new\_username, str) and "@" in new\_username:

self.\_username = new\_username

return f"Username changed to {self.\_username}"

else:

return "Username must be a string and contain '@'"

# Getter for private attribute (denied)

def get\_password(self):

return "Permission denied"

# Method to display account information

def display(self):

return f"Name: {self.name}\nUsername: {self.\_username}"

# Creating object

user1 = GmailAccount("Nehan", "nehan123@gmail.com")

# Usage

print(user1.display())

print(user1.get\_username())

print(user1.set\_username("nehan456@gmail.com"))

print(user1.get\_password())

print(user1.display())

**Q3: What is inheritance and how is it implemented in Python?**

**Ans:** Inheritance in Python is a feature of object-oriented programming where a class (child) can acquire properties and methods from another class (parent). It promotes code reusability and is implemented using the syntax class Child(Parent): when a class Parent is created first also super() method can be used to call attributes from parent class.

**Q4: What is method overriding in subclasses?**

**Ans:** Method overriding in subclasses means redefining a method in the child class that already exists in the parent class. It allows the subclass to provide a specific implementation of the method that replaces the parent’s version with the same method name.

**Q5: Why is self required in instance methods?**

**Ans:** self is a must-have first parameter in instance methods because it refers to the current object of the class. It allows access to instance variables and other methods, enabling each object to maintain its own state.