

Student Name: _____

Roll No: _____

Section: _____

Lab Series No. 04.

Lab 04 –Class Variables and Data Hiding.

Lab Objectives:

1. Class and Instance Variables
2. Data Hiding
3. Self Parameter
4. Modification to attributes inside the class

1. Class and Instance Variables

In Python, instance variables are variables whose value is assigned inside a constructor or method with self. Class variables are variables whose value is assigned in class.

Program 1: Write a Python program to create variable inside the class with string value and accessing it.

Code:

```
# Python program to show that the variables with a value assigned in
class declaration, are class variables and variables inside methods
and constructors are instance variables.
```

```
# Class for Computer Science Student
class CSStudent:

    # Class Variable
    trade = 'Computer Science Software Engineering'

    # The init method or constructor
    def __init__(self, roll):

        # Instance Variable
        self.roll = roll

# Objects of CSStudent class
Faisal = CSStudent(315)
Farhan = CSStudent(316)

print(Faisal.trade)
print(Faisal.roll)
print(Faisal.trade)
print(Farhan.roll)
```

Student Name: _____

Roll No: _____

Section: _____

```
# Class variables can be accessed using class name also
print(CSStudent.trade)
```

Program 2: Write a Python program to create instance variable inside method and accessing it.

Code:

```
# Python program to show that the variables with a value assigned in
class declaration, are class variables and variables inside methods
and constructors are instance variables.
```

```
# Class for Computer Science Student
class CSStudent:

    # Class Variable
    trade1 = 'Computer Science'
    trade2 = 'Software Engineering'

    # The init method or constructor
    def __init__(self, roll):

        # Instance Variable
        self.roll = roll

    # Adds an instance variable
    def setAddress(self, address):
        self.address = address

    # Retrieves instance variable
    def getAddress(self):
        return self.address

# Objects of CSStudent class
Faisal = CSStudent(315)
Faisal.setAddress("Gulistan e Johar")
print(Faisal.getAddress())
```

2. Data Hiding

In Python, we use double underscore (Or_) before the attributes name and those attributes will not be directly visible outside.

Student Name: _____

Roll No: _____

Section: _____

Program 3: Write a Python program which can create a hidden variable inside the class. Then try to access it.

Code:

```
class MyClass:

    # Hidden member of MyClass
    __hiddenVariable = 0

    # A member method that changes
    # __hiddenVariable
    def add(self, increment):
        self.__hiddenVariable += increment
        print (self.__hiddenVariable)

# Executing the Code
myObject = MyClass()
myObject.add(2)
myObject.add(5)

# This line causes error
print (myObject.__hiddenVariable)
```

Program 4: Write a Python program which can create a hidden variable inside the class. Then try to access it by using tricky method.

Code:

```
# A Python program to demonstrate that hidden
# members can be accessed outside a class
class MyClass:

    # Hidden member of MyClass
    __hiddenVariable = 10

# Executing code
myObject = MyClass()
print(myObject._MyClass__hiddenVariable)
```

Student Name: _____

Roll No: _____

Section: _____

4. The Self Parameter

The self-parameter is a reference to the current instance of the class, and is used to access variables that belongs to the class.

It does not have to be named self, you can call it whatever you like, but it has to be the first parameter of any function in the class:

Program 5: Write a Python program which use other than self keyword for reference .

Code:

```
class Person:
    def __init__(a, name, age, salary, profession):
        a.name = name
        a.age = age
        a.salary = salary
        a.profession = profession

    def mydetail(a):
        print("Assalam o Alekum, my name is " + a.name + ", my age is " + str(a.age) + ". Now a days earning :"+ str(a.salary) + ", its really lovely to be a " + a.profession )

person1 = Person("Syed Faisal Ali", 44, 1234567, "Research Scientist")
person1.mydetail()
```

Program 6: Write a Python program which can modify the attributes inside the class.

Code:

```
class Person:
    def __init__(a, name, age, salary, profession):
        a.name = name
        a.age = age
        a.salary = salary
        a.profession = profession

    def mydetail(a):
        print("Assalam o Alekum, my name is " + a.name + ", my age is " + str(a.age) + ". Now a days earning :"+ str(a.salary) + ", its really lovely to be a " + a.profession )

person1 = Person("Syed Faisal Ali", 42, 1234567, "Research Scientist")
person1.mydetail()
```

Student Name: _____

Roll No: _____

Section: _____

```
print("Need to modify the age and salary inside the class")
person1.age = 44
person1.salary = 7654321
person1.mydetail()
```

Program 7: Write a Python program which uses the child class to call its own method and then parent class method.

Code:

```
class Parent:          # define parent class
    parentAttr = 100
    def __init__(self):
        print ("Calling parent constructor")

    def parentMethod(self):
        print ('Calling parent method')

    def setAttr(self, attr):
        Parent.parentAttr = attr

    def getAttr(self):
        print ("Parent attribute :", Parent.parentAttr)

class Child(Parent): # define child class
    def __init__(self):
        print ("Calling child constructor")

    def childMethod(self):
        print ('Calling child method')

c1 = Child()          # instance of child
c1.childMethod()      # child calls its method
c1.parentMethod()     # calls parent's method
c1.setAttr(200)       # again call parent's method
c1.getAttr()          # again call parent's method
```

Student Name: _____

Roll No: _____

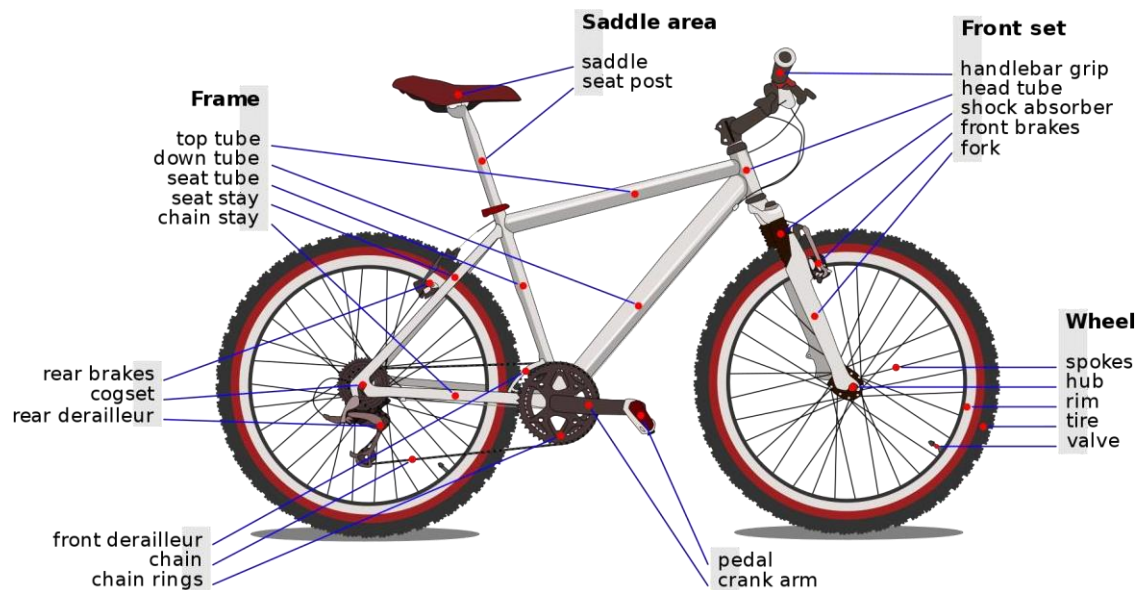
Section: _____

Programming Exercise

Task1: Define what you understand by the classes, objects and functionalities for the following scenarios.

Task 2: Make a class for insurance which can have multiple insurance policies. Design its UML and Case diagram before coding. Later modify the attributes in child classes.

Task 3: Create a bike class and its components in light with the concept of Object Oriented. Later create multiple bikes with different attributes based on customer requirements.



Task 7: You are working in Galaxy Computer as Programmer. The main business is selling computer, laptops its accessories, used computer etc. Details can be find from <https://galaxy.pk/>. Your task is to analyze the website, define which classes will be require and create the complete program which can take user name and its requirements then give the total amount of computer/ laptop or accessories.