CECOS UNIVERSITY

Name: Muhammad Ozair
ID: 4356-2023B
Lab Task: 4
Subject: Software Construction and Development
Department: BSSE
Semester: 5th
Date: 4 December 2025

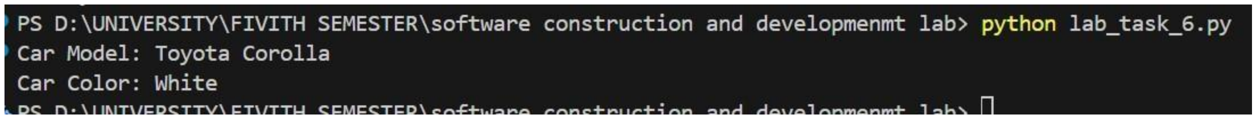
1. Create a Car class with attributes for model and color, and print them using an object. Code:

```
class Car:     def __init__(self, model, color):
        self.model = model         self.color = color

# Object creation car1 = Car("Toyota
Corolla", "White")

# Printing attributes print("Car
Model:", car1.model) print("Car Color:", car1.color)
```

output:

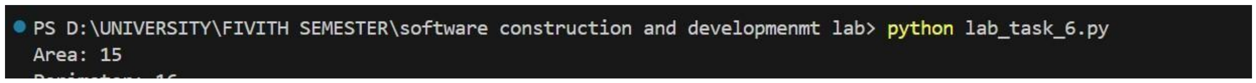


```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Car Model: Toyota Corolla
Car Color: White
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab>
```

2. Calculate the area and perimeter of a rectangle using a class with length and width attributes. Code:

```
class Rectangle:     def __init__(self,
length, width):
        self.length = length
self.width = width       def
area(self):
        return self.length * self.width
def perimeter(self):
        return 2 * (self.length + self.width)
r = Rectangle(5, 3) print("Area:", r.area())
print("Perimeter:", r.perimeter())
```

output:

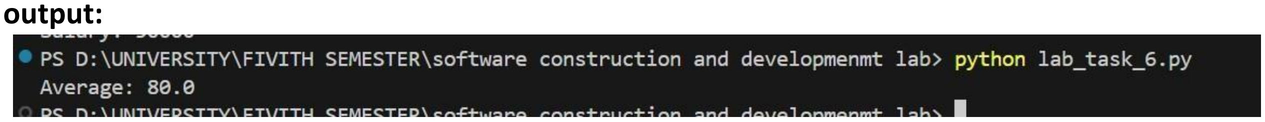


```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Area: 15
Perimeter: 16
```

3. Create a student class that takes name and marks of 3 subjects as arguments in constructor. Then create a method to print the average. Code:

```
class Student:     def __init__(self, name,
m1, m2, m3):
        self.name = name
self.m1 = m1         self.m2
= m2         self.m3 = m3
def average(self):
        return (self.m1 + self.m2 + self.m3) / 3
s = Student("Rashid", 80, 70, 90)
print("Average:", s.average())
```

output:

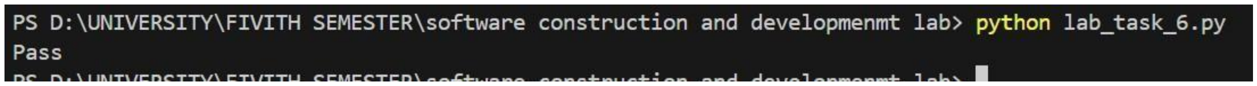


```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Average: 80.0
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab>
```

4. Check if a student has passed or failed based on their marks using a method inside a student class.

```
class Student:     def __init__(self,
marks):
        self.marks = marks
def result(self):
if self.marks >= 40:
print("Pass")         else:
print("Fail")   s =
Student(55)  s.result()
```

Output:

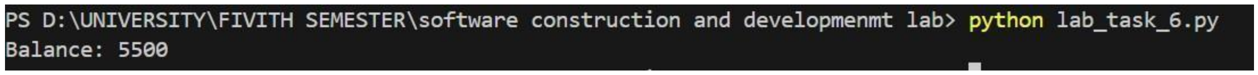


```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Pass
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab>
```

5.	**Create account class with 2 attributes – balance and account number & Create a method for debit, credit and printing balance.**

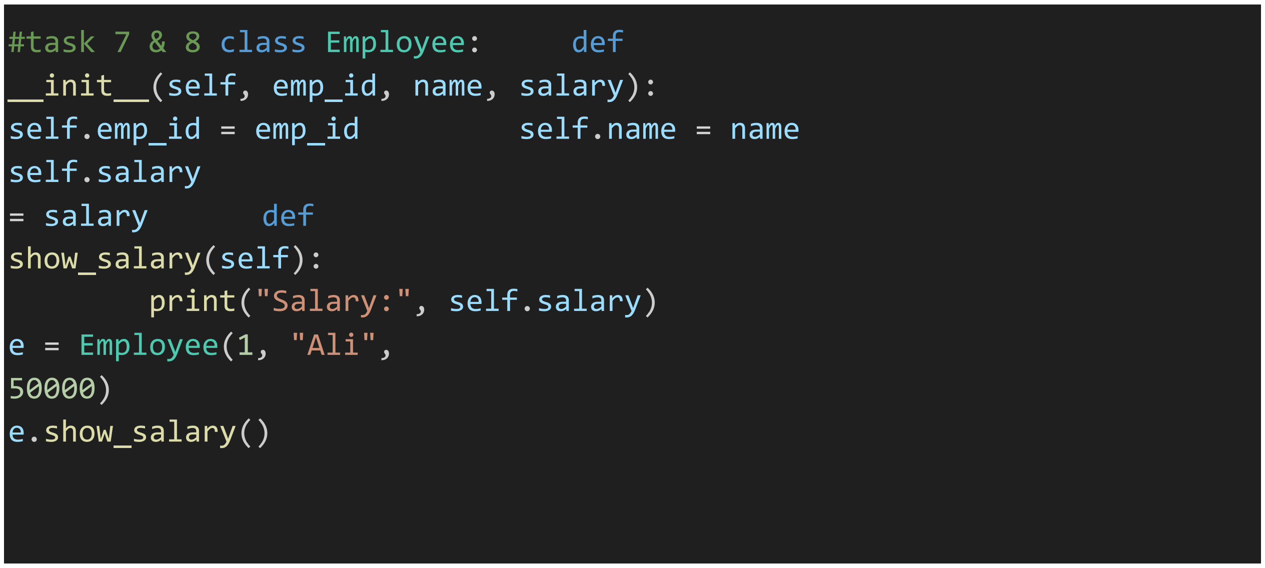
```
#task 5 & 6 class Account:     def __init__(self,
acc_no, balance):
        self.acc_no = acc_no
self.balance = balance
def credit(self, amount):
self.balance += amount
def debit(self, amount):
self.balance -= amount       def
show_balance(self):
print("Balance:", self.balance)   a =
Account(101,
5000)
a.credit(1000)
a.debit(500)
a.show_balance()
```

Output:



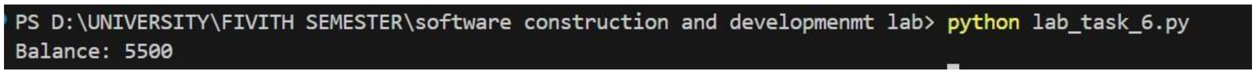
```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Balance: 5500
```

7. Define a class Employee with attributes id, name, and salary.
& Define a method to calculate and display annual salary in the Employee class.



```
#task 7 & 8 class Employee:     def
__init__(self, emp_id, name, salary):
self.emp_id = emp_id         self.name = name
self.salary
= salary      def
show_salary(self):
        print("Salary:", self.salary)
e = Employee(1, "Ali",
50000)
e.show_salary()
```

Output:

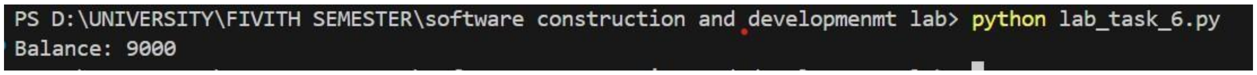


```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Balance: 5500
```

9. A banking system needs to protect account balance details.
Create a Bank Account class where the balance is private and can only be accessed using deposit and withdraw methods.

```
#task 9 class BankAccount:     def
__init__(self, balance):
        self.__balance = balance   # private variable
def deposit(self, amount):
self.__balance += amount
def withdraw(self, amount):
self.__balance -= amount
def show_balance(self):
        print("Balance:", self.__balance)
b =
BankAccount(10000)  b.deposit(2000)
b.withdraw(3000)
b.show_balance()
```

Output

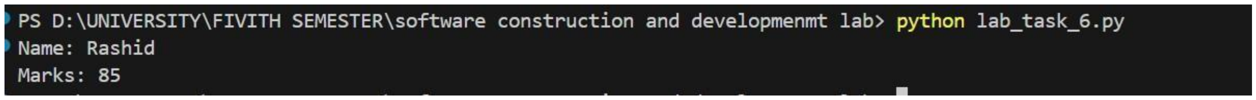


```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Balance: 9000
```

10. A school management system stores student marks securely.
Create a student class with private attributes for name and marks and display them using public methods.

```
class Student:      def __init__(self,
name, marks):
        self.__name = name
self.__marks = marks
def display(self):
        print("Name:", self.__name)
print("Marks:", self.__marks)   s =
Student("Rashid",
85)
s.display()
```

Output:

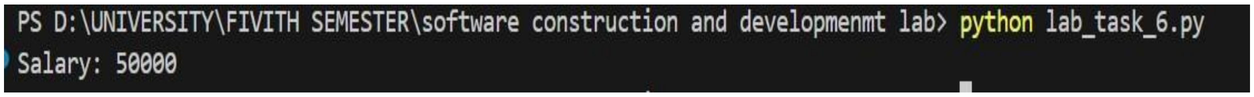


```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Name: Rashid
Marks: 85
```

11. A company payroll system should not allow direct access to salary data. Create an employee class with a private salary attribute and methods to update and display the salary.

```
lass Employee:      def
__init__(self, salary):
self.__salary = salary
def update_salary(self,
new_salary):
        self.__salary = new_salary
def show_salary(self):
        print("Salary:", self.__salary)
e =
Employee(40000)
e.update_salary(50000)
e.show_salary()
```

Output:



```
PS D:\UNIVERSITY\FIVITH SEMESTER\software construction and developmenmt lab> python lab_task_6.py
Salary: 50000
```