

**Name:** Saeed Ahmed

**ID:** BIT-23F-018

**Section**: IT- 3 A

**Department:** BS-IT

**Course:** Artificial Intelligence (AI)

**Lab Task:**  Lab #08

**Assigned By:** Aqsa Umar

* **Task 01:-** Write a Python class named Car that represents a car. The class should have the following attributes:
* make: the car's make (e.g., "Toyota")
* model: the car's model (e.g., "Corolla")
* year: the car's manufacturing year (e.g., 2020)
* mileage: the number of miles driven by the car.

The class should have the following methods:

* \_\_init\_\_(self): Constructor to initialize the car's attributes.
* display\_info(): Displays the car's information (make, model, year, mileage).
* drive(miles): Increases the mileage by the specified number of miles.

**Program:-**

class Car:

def \_\_init\_\_(self, make, model, year, mileage):

self.make = make

self.model = model

self.year = year

self.mileage = mileage

def display\_info(self):

print(f"Make: {self.make}")

print(f"Model: {self.model}")

print(f"Year: {self.year}")

print(f"Mileage: {self.mileage} miles")

def drive(self, miles):

self.mileage += miles

print(f"Driven {miles} miles. New mileage is {self.mileage} miles.")

car = Car("Toyota", "Corolla", 2020, 15000)

car.display\_info()

car.drive(500)

car.display\_info()

* **Task 02:-** Write a Python class named Student that represents a student. The class should have the following attributes:
* name: the student's name.
* age: the student's age.
* marks: a list of the student's marks.

The class should have the following methods:

* \_\_init\_\_(self): Constructor to initialize the student's attributes.
* add\_marks(self, marks): Adds a list of marks to the student's marks list.
* average\_marks(self): Calculates and returns the average of the student's marks.
* display\_info(self): Displays the student's information (name, age, average marks).

**Program:-**

class Student:

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

self.name = name

self.age = age

self.marks = []

def add\_marks(self, marks):

self.marks.extend(marks)

def average\_marks(self):

if not self.marks:

return 0

return sum(self.marks) / len(self.marks)

def display\_info(self):

avg\_marks = self.average\_marks()

print(f"Name: {self.name}")

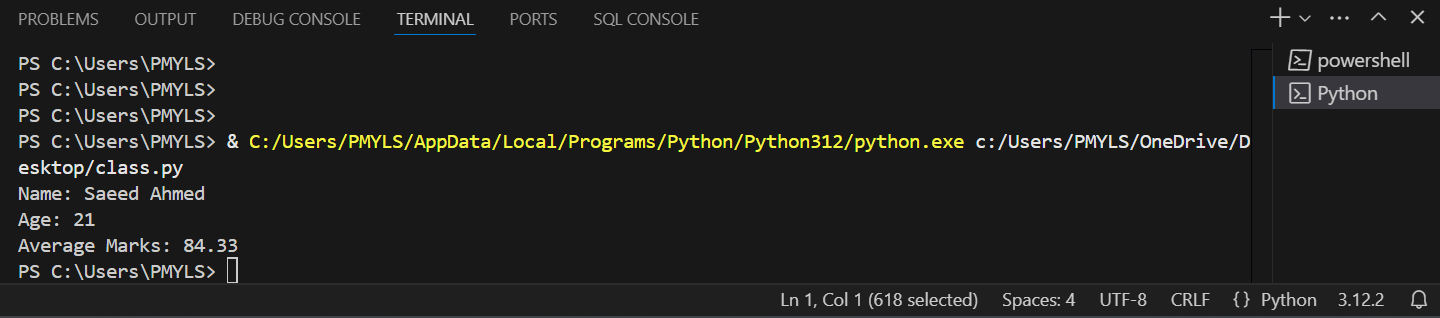
print(f"Age: {self.age}")

print(f"Average Marks: {avg\_marks:.2f}")

student = Student("Saeed Ahmed", 21)

student.add\_marks([85, 90, 78])

student.display\_info()



**Task 03:**Write a Python class named BankAccount that represents a bank account. The class should have the following attributes:

* account\_holder: the name of the account holder.
* balance: the balance of the account.

The class should have the following methods:

* \_\_init\_\_(self): Constructor to initialize the account holder's name and balance.
* deposit(self, amount): Deposits an amount into the account.
* withdraw(self, amount): Withdraws an amount from the account if there are sufficient funds.
* display\_balance(self): Displays the current balance of the account.

**Program:-**

class BankAccount:

def \_\_init\_\_(self, account\_holder, balance=0):

self.account\_holder = account\_holder

self.balance = balance

def deposit(self, amount):

if amount > 0:

self.balance += amount

print(f"Deposited {amount}. New balance: {self.balance}")

else:

print("Deposit amount must be positive.")

def withdraw(self, amount):

if amount > self.balance:

print("Insufficient funds.")

elif amount <= 0:

print("Withdrawal amount must be positive.")

else:

self.balance -= amount

print(f"Withdrew {amount}. New balance: {self.balance}")

def display\_balance(self):

print(f"Account holder: {self.account\_holder}")

print(f"Current balance: {self.balance}")

account = BankAccount("Saeed Ahmed", 100000)

account.display\_balance()

account.deposit(5000)

account.withdraw(3000)

account.display\_balance()

