PRACTICE PROBLEM SET

WEEK 3

SUBMITTED BY:

USMAN RASHEED SIDDIQUI (CS-24038)

SUBMITTED TO:

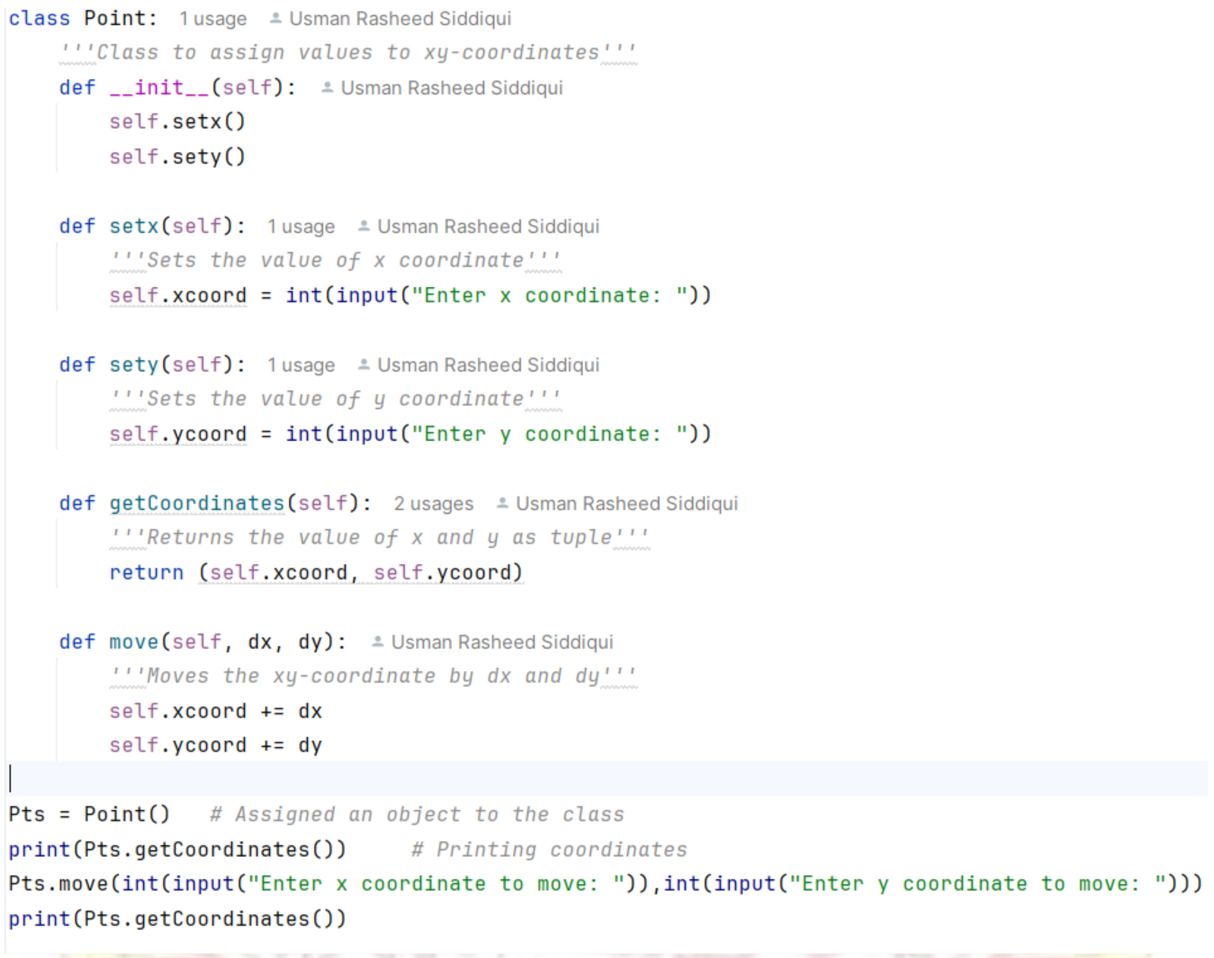
MS. FAUZIA YASIR

NED UNIVERSITY OF ENGINEERING AND TECHNOLOGY

CS-116

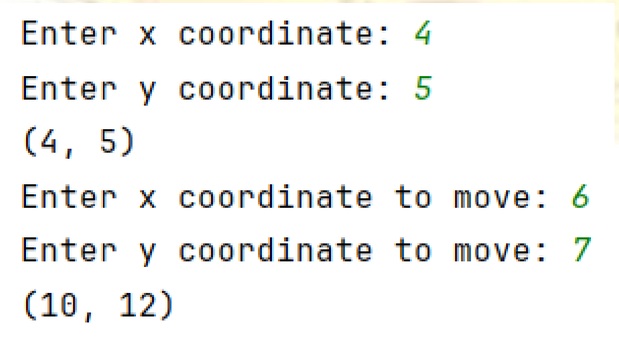
PRACTICE PROBLEM 1:

INPUT:



```
class Point:  1 usage  Usman Rasheed Siddiqui
    '''Class to assign values to xy-coordinates'''
    def __init__(self):  Usman Rasheed Siddiqui
        self.setx()
        self.sety()

    def setx(self):  1 usage  Usman Rasheed Siddiqui
        '''Sets the value of x coordinate'''
        self.xcoord = int(input("Enter x coordinate: "))

    def sety(self):  1 usage  Usman Rasheed Siddiqui
        '''Sets the value of y coordinate'''
        self.ycoord = int(input("Enter y coordinate: "))

    def getCoordinates(self):  2 usages  Usman Rasheed Siddiqui
        '''Returns the value of x and y as tuple'''
        return (self.xcoord, self.ycoord)

    def move(self, dx, dy):  Usman Rasheed Siddiqui
        '''Moves the xy-coordinate by dx and dy'''
        self.xcoord += dx
        self.ycoord += dy

Pts = Point()   # Assigned an object to the class
print(Pts.getCoordinates())    # Printing coordinates
Pts.move(int(input("Enter x coordinate to move: ")),int(input("Enter y coordinate to move: ")))
print(Pts.getCoordinates())
```

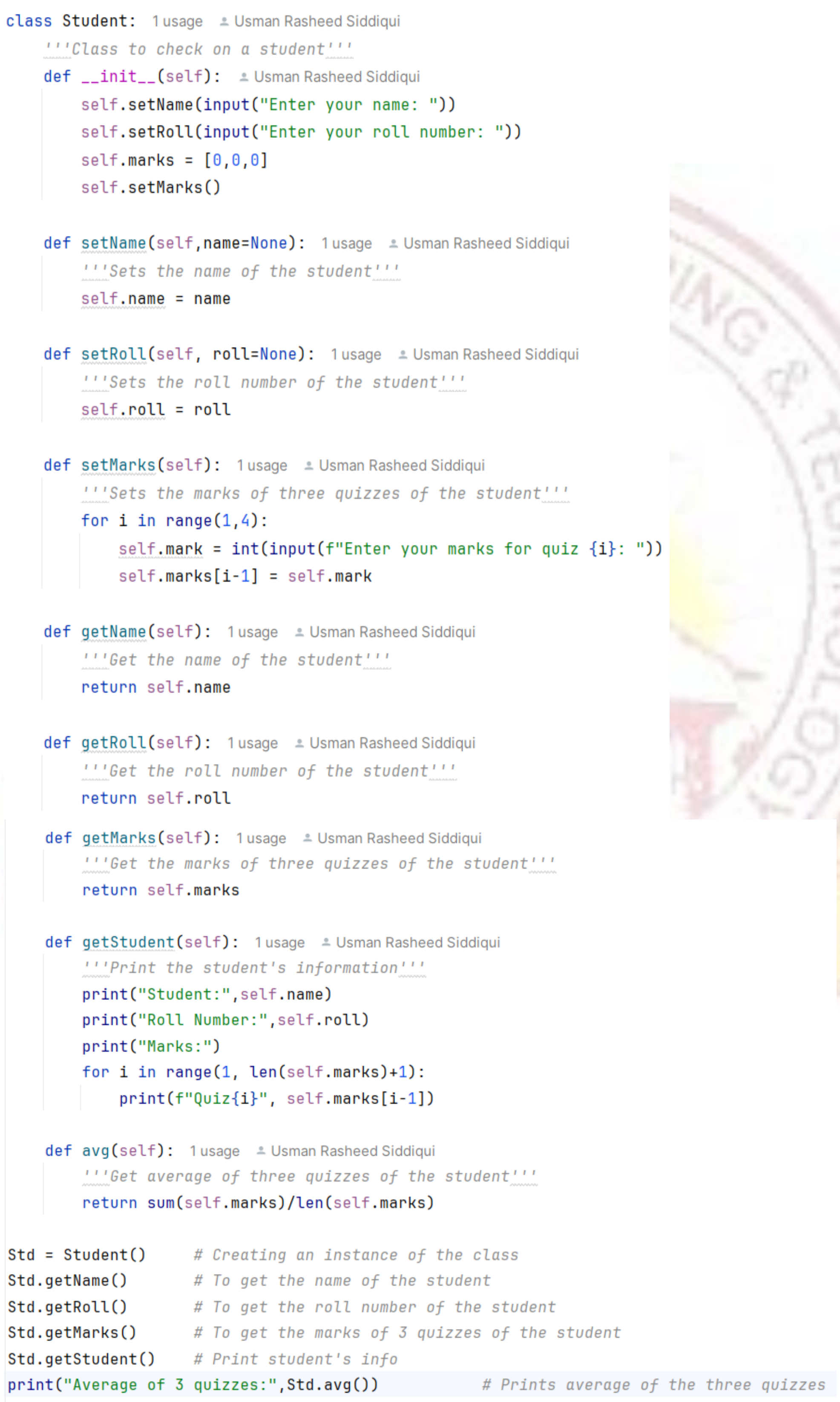
OUTPUT:



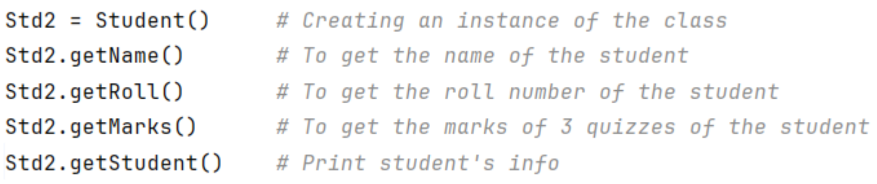
```
Enter x coordinate: 4
Enter y coordinate: 5
(4, 5)
Enter x coordinate to move: 6
Enter y coordinate to move: 7
(10, 12)
```

PRACTICE PROBLEM 1:

INPUT:

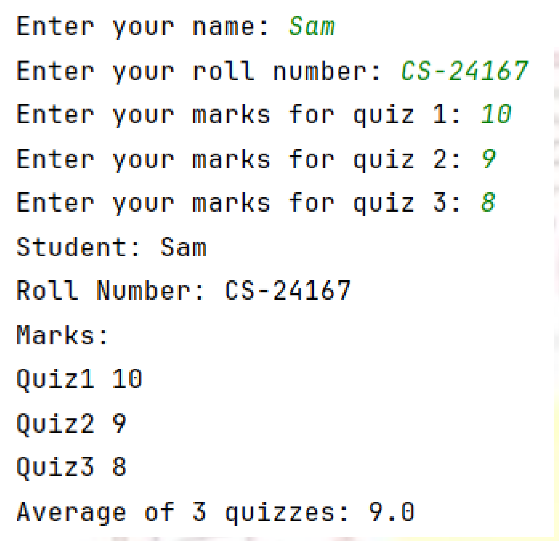


```python
class Student:  1 usage  Usman Rasheed Siddiqui
    '''Class to check on a student'''
    def __init__(self):  Usman Rasheed Siddiqui
        self.setName(input("Enter your name: "))
        self.setRoll(input("Enter your roll number: "))
        self.marks = [0,0,0]
        self.setMarks()

    def setName(self,name=None):  1 usage  Usman Rasheed Siddiqui
        '''Sets the name of the student'''
        self.name = name

    def setRoll(self, roll=None):  1 usage  Usman Rasheed Siddiqui
        '''Sets the roll number of the student'''
        self.roll = roll

    def setMarks(self):  1 usage  Usman Rasheed Siddiqui
        '''Sets the marks of three quizzes of the student'''
        for i in range(1,4):
            self.mark = int(input(f"Enter your marks for quiz {i}: "))
            self.marks[i-1] = self.mark

    def getName(self):  1 usage  Usman Rasheed Siddiqui
        '''Get the name of the student'''
        return self.name

    def getRoll(self):  1 usage  Usman Rasheed Siddiqui
        '''Get the roll number of the student'''
        return self.roll

    def getMarks(self):  1 usage  Usman Rasheed Siddiqui
        '''Get the marks of three quizzes of the student'''
        return self.marks

    def getStudent(self):  1 usage  Usman Rasheed Siddiqui
        '''Print the student's information'''
        print("Student:",self.name)
        print("Roll Number:",self.roll)
        print("Marks:")
        for i in range(1, len(self.marks)+1):
            print(f"Quiz{i}", self.marks[i-1])

    def avg(self):  1 usage  Usman Rasheed Siddiqui
        '''Get average of three quizzes of the student'''
        return sum(self.marks)/len(self.marks)

Std = Student()      # Creating an instance of the class
Std.getName()        # To get the name of the student
Std.getRoll()        # To get the roll number of the student
Std.getMarks()       # To get the marks of 3 quizzes of the student
Std.getStudent()     # Print student's info
print("Average of 3 quizzes:",Std.avg())          # Prints average of the three quizzes
```

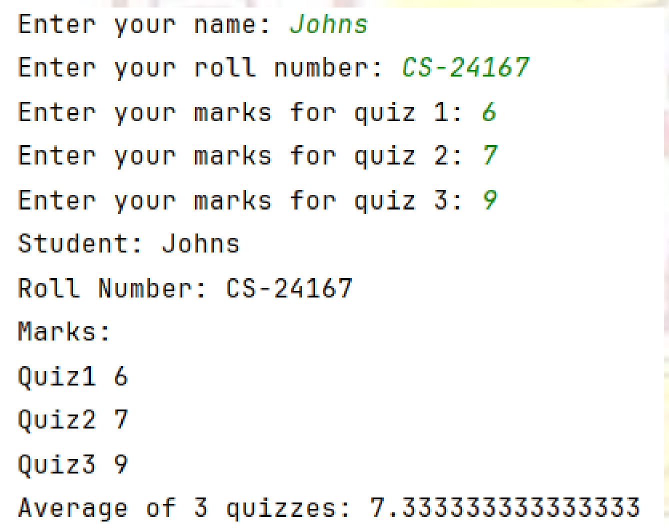


```
Std2 = Student()      # Creating an instance of the class
Std2.getName()        # To get the name of the student
Std2.getRoll()        # To get the roll number of the student
Std2.getMarks()       # To get the marks of 3 quizzes of the student
Std2.getStudent()     # Print student's info
```

OUTPUT:



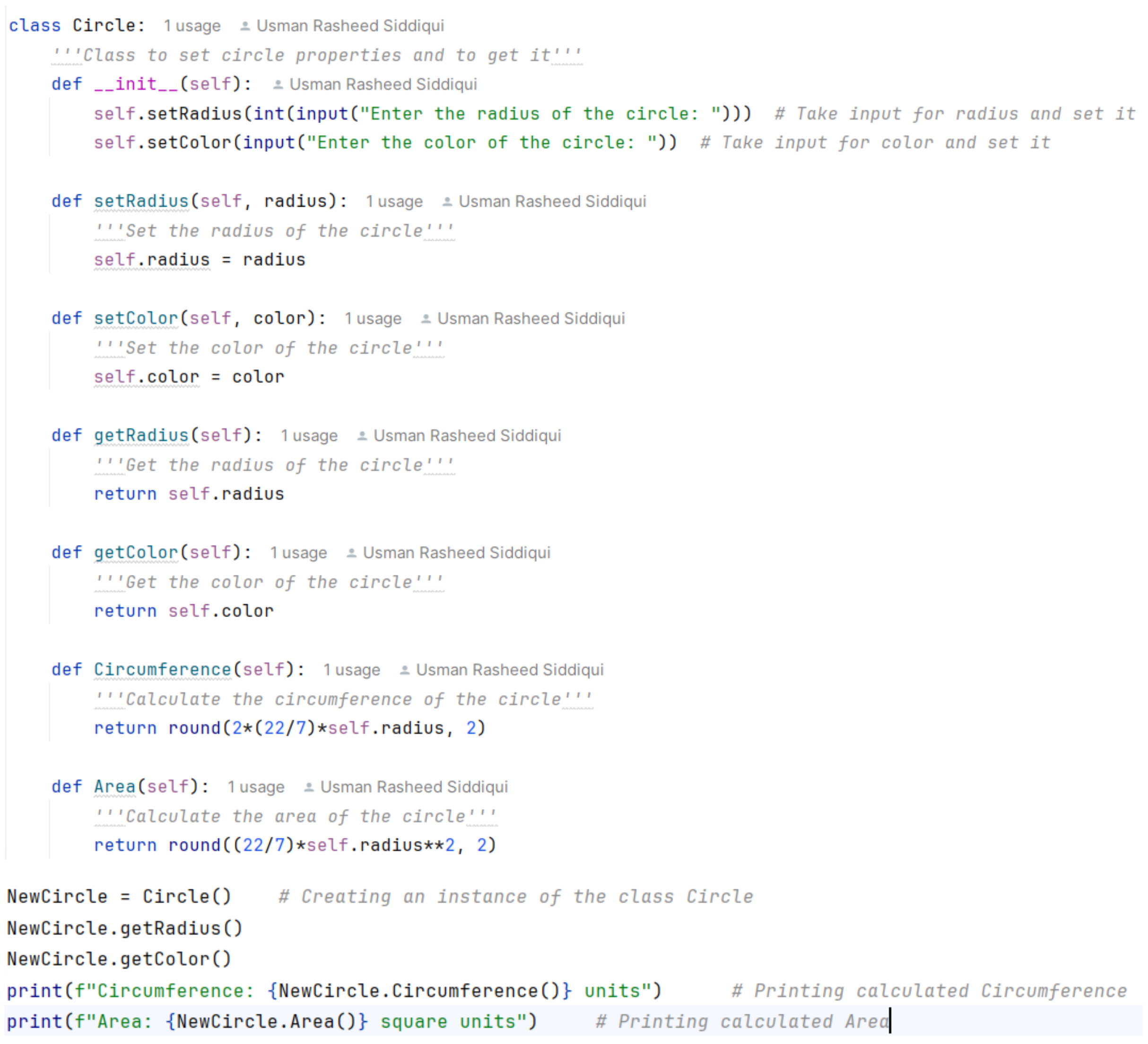
```
Enter your name: Sam
Enter your roll number: CS-24167
Enter your marks for quiz 1: 10
Enter your marks for quiz 2: 9
Enter your marks for quiz 3: 8
Student: Sam
Roll Number: CS-24167
Marks:
Quiz1 10
Quiz2 9
Quiz3 8
Average of 3 quizzes: 9.0
```



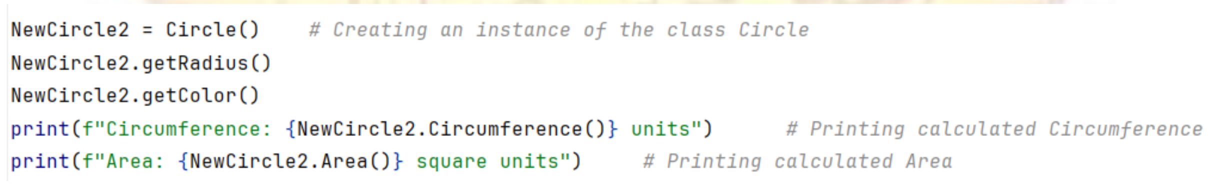
```
Enter your name: Johns
Enter your roll number: CS-24167
Enter your marks for quiz 1: 6
Enter your marks for quiz 2: 7
Enter your marks for quiz 3: 9
Student: Johns
Roll Number: CS-24167
Marks:
Quiz1 6
Quiz2 7
Quiz3 9
Average of 3 quizzes: 7.333333333333333
```

PRACTICE PROBLEM 3:

INPUT:

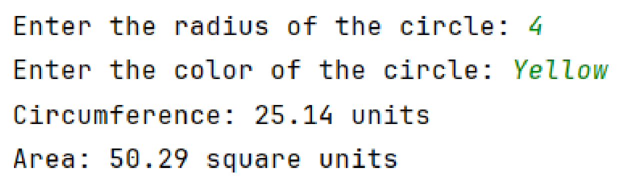


```
class Circle:  1 usage  Usman Rasheed Siddiqui
    '''Class to set circle properties and to get it'''
    def __init__(self):  Usman Rasheed Siddiqui
        self.setRadius(int(input("Enter the radius of the circle: ")))  # Take input for radius and set it
        self.setColor(input("Enter the color of the circle: "))  # Take input for color and set it

    def setRadius(self, radius):  1 usage  Usman Rasheed Siddiqui
        '''Set the radius of the circle'''
        self.radius = radius

    def setColor(self, color):  1 usage  Usman Rasheed Siddiqui
        '''Set the color of the circle'''
        self.color = color

    def getRadius(self):  1 usage  Usman Rasheed Siddiqui
        '''Get the radius of the circle'''
        return self.radius

    def getColor(self):  1 usage  Usman Rasheed Siddiqui
        '''Get the color of the circle'''
        return self.color

    def Circumference(self):  1 usage  Usman Rasheed Siddiqui
        '''Calculate the circumference of the circle'''
        return round(2*(22/7)*self.radius, 2)

    def Area(self):  1 usage  Usman Rasheed Siddiqui
        '''Calculate the area of the circle'''
        return round((22/7)*self.radius**2, 2)

NewCircle = Circle()    # Creating an instance of the class Circle
NewCircle.getRadius()
NewCircle.getColor()
print(f"Circumference: {NewCircle.Circumference()} units")      # Printing calculated Circumference
print(f"Area: {NewCircle.Area()} square units")      # Printing calculated Area
```

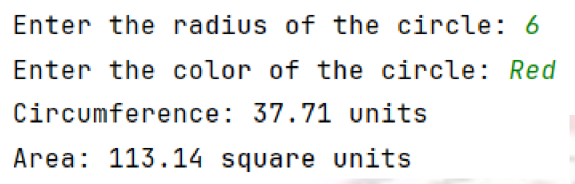


```
NewCircle2 = Circle()    # Creating an instance of the class Circle
NewCircle2.getRadius()
NewCircle2.getColor()
print(f"Circumference: {NewCircle2.Circumference()} units")      # Printing calculated Circumference
print(f"Area: {NewCircle2.Area()} square units")      # Printing calculated Area
```

OUTPUT:



```
Enter the radius of the circle: 4
Enter the color of the circle: Yellow
Circumference: 25.14 units
Area: 50.29 square units
```

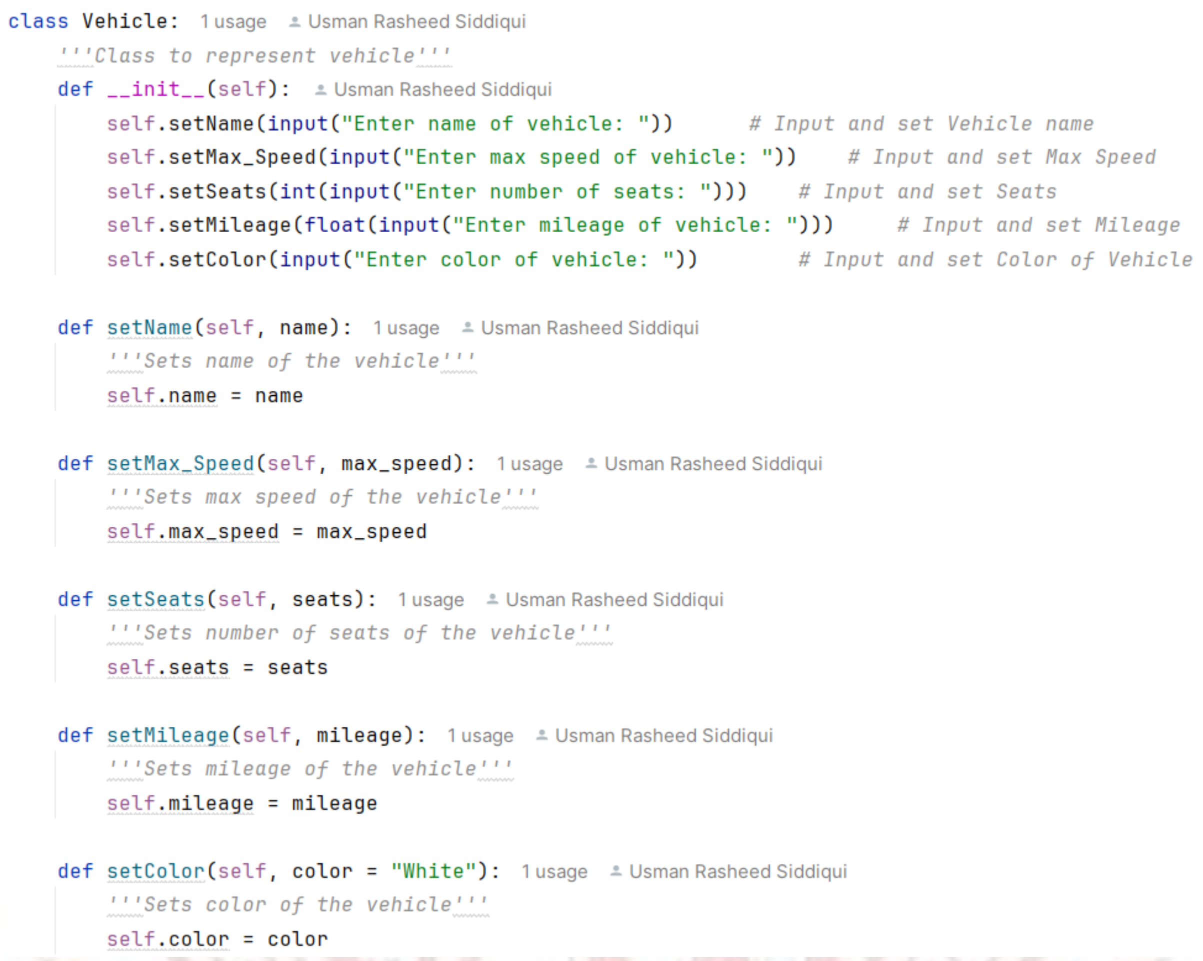


```
Enter the radius of the circle: 6
Enter the color of the circle: Red
Circumference: 37.71 units
Area: 113.14 square units
```

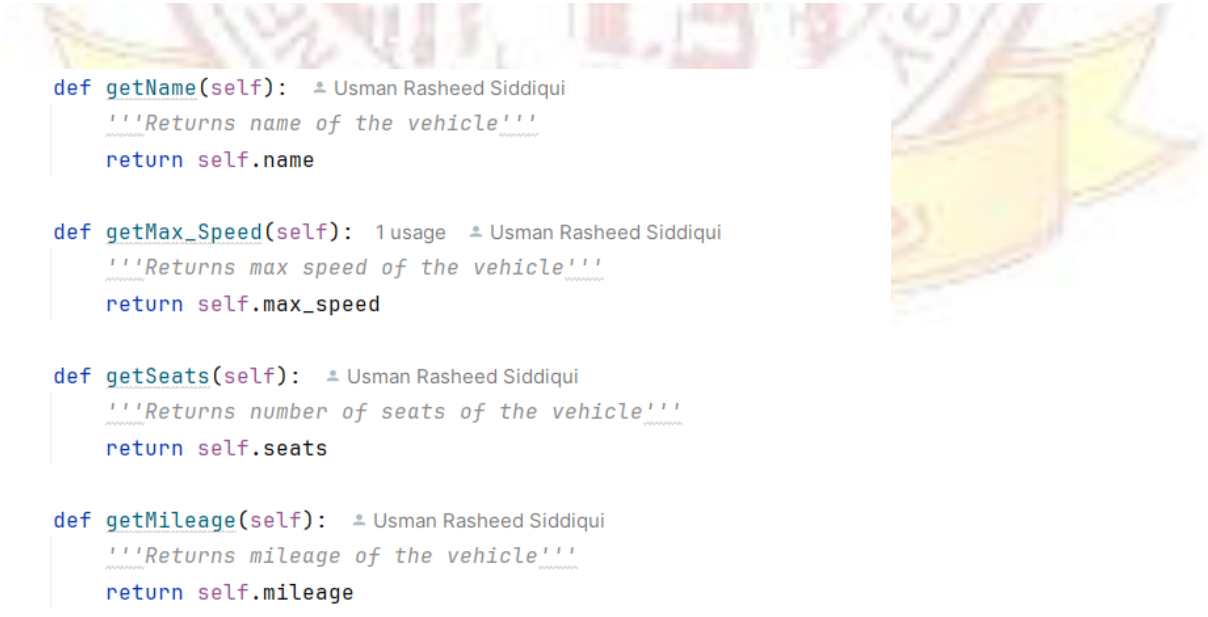
Dated: 4th February 2025

PRACTICE PROBLEM 4:

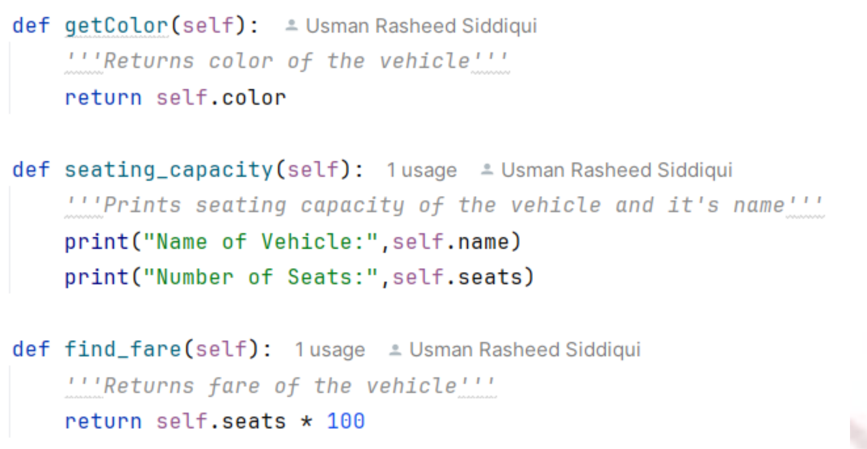
INPUT:



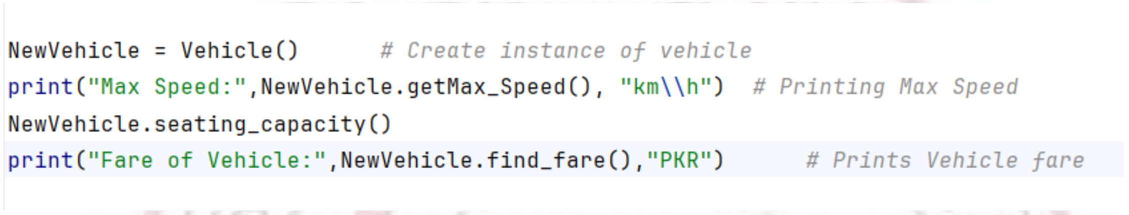
```
class Vehicle:  1 usage  Usman Rasheed Siddiqui
    '''Class to represent vehicle'''
    def __init__(self):  Usman Rasheed Siddiqui
        self.setName(input("Enter name of vehicle: "))       # Input and set Vehicle name
        self.setMax_Speed(input("Enter max speed of vehicle: "))    # Input and set Max Speed
        self.setSeats(int(input("Enter number of seats: ")))    # Input and set Seats
        self.setMileage(float(input("Enter mileage of vehicle: ")))     # Input and set Mileage
        self.setColor(input("Enter color of vehicle: "))        # Input and set Color of Vehicle

    def setName(self, name):  1 usage  Usman Rasheed Siddiqui
        '''Sets name of the vehicle'''
        self.name = name

    def setMax_Speed(self, max_speed):  1 usage  Usman Rasheed Siddiqui
        '''Sets max speed of the vehicle'''
        self.max_speed = max_speed

    def setSeats(self, seats):  1 usage  Usman Rasheed Siddiqui
        '''Sets number of seats of the vehicle'''
        self.seats = seats

    def setMileage(self, mileage):  1 usage  Usman Rasheed Siddiqui
        '''Sets mileage of the vehicle'''
        self.mileage = mileage

    def setColor(self, color = "White"):  1 usage  Usman Rasheed Siddiqui
        '''Sets color of the vehicle'''
        self.color = color
```



```
    def getName(self):  Usman Rasheed Siddiqui
        '''Returns name of the vehicle'''
        return self.name

    def getMax_Speed(self):  1 usage  Usman Rasheed Siddiqui
        '''Returns max speed of the vehicle'''
        return self.max_speed

    def getSeats(self):  Usman Rasheed Siddiqui
        '''Returns number of seats of the vehicle'''
        return self.seats

    def getMileage(self):  Usman Rasheed Siddiqui
        '''Returns mileage of the vehicle'''
        return self.mileage
```

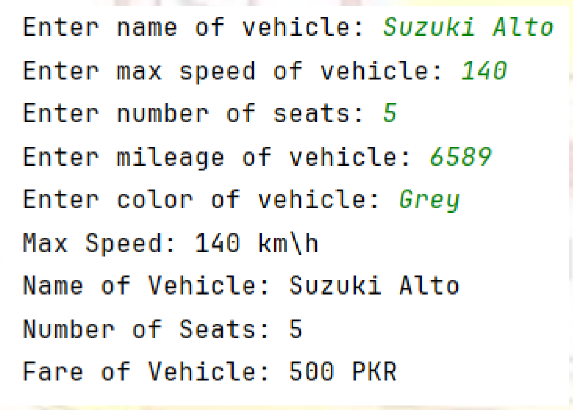


```
    def getColor(self):  Usman Rasheed Siddiqui
        '''Returns color of the vehicle'''
        return self.color

    def seating_capacity(self):  1 usage  Usman Rasheed Siddiqui
        '''Prints seating capacity of the vehicle and it's name'''
        print("Name of Vehicle:",self.name)
        print("Number of Seats:",self.seats)

    def find_fare(self):  1 usage  Usman Rasheed Siddiqui
        '''Returns fare of the vehicle'''
        return self.seats * 100
```

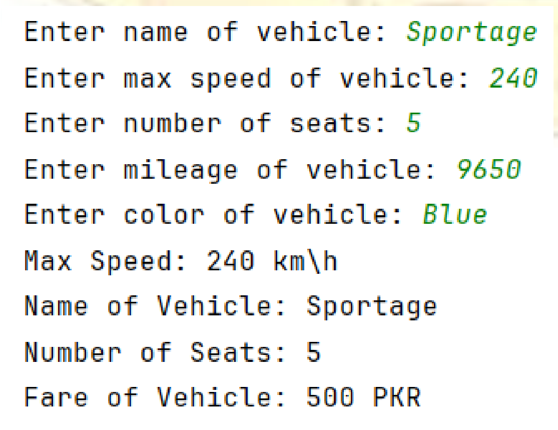


```
NewVehicle = Vehicle()      # Create instance of vehicle
print("Max Speed:",NewVehicle.getMax_Speed(), "km\\h")  # Printing Max Speed
NewVehicle.seating_capacity()
print("Fare of Vehicle:",NewVehicle.find_fare(),"PKR")      # Prints Vehicle fare
```

OUTPUT:



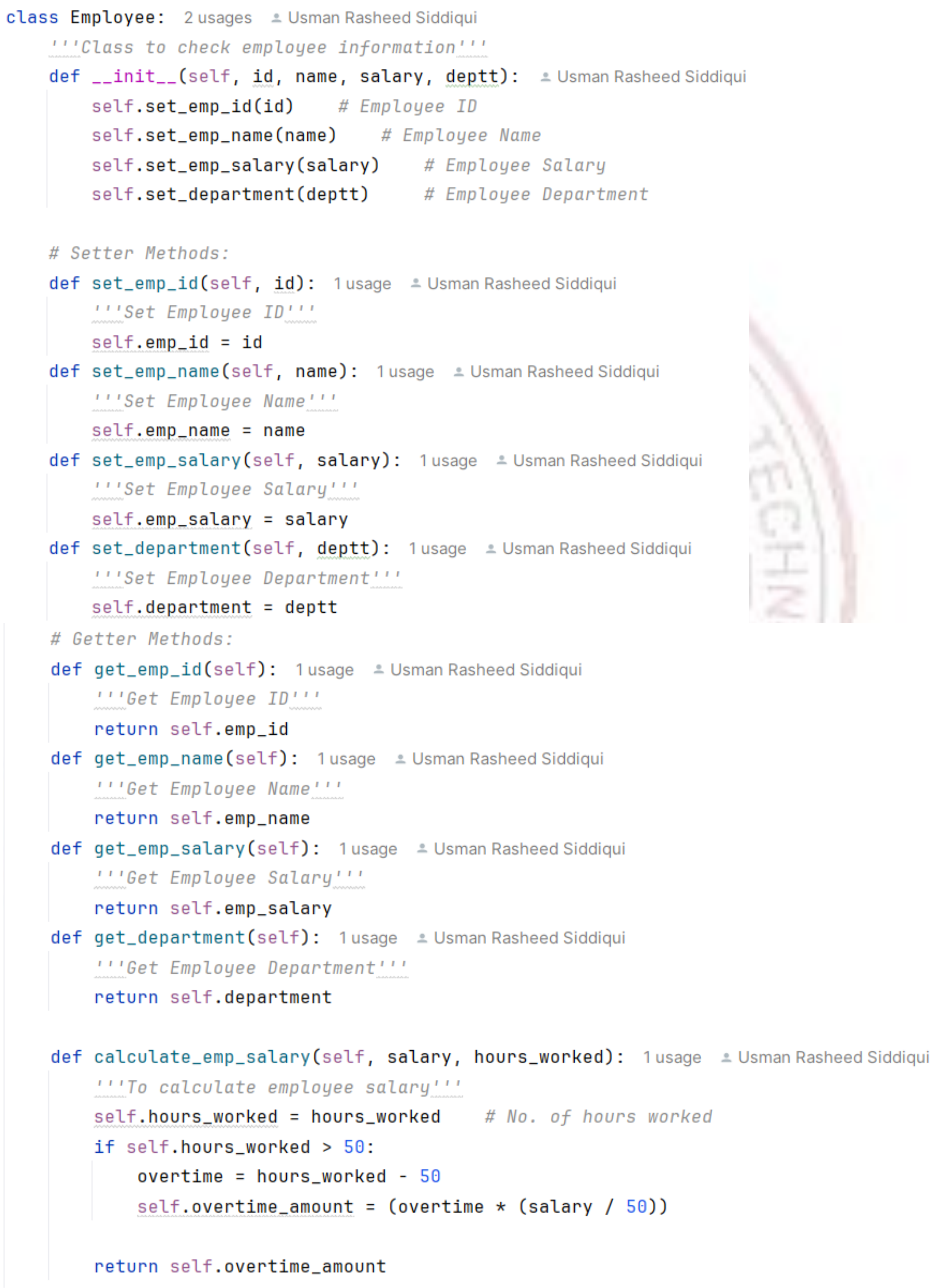
```
Enter name of vehicle: Suzuki Alto
Enter max speed of vehicle: 140
Enter number of seats: 5
Enter mileage of vehicle: 6589
Enter color of vehicle: Grey
Max Speed: 140 km\h
Name of Vehicle: Suzuki Alto
Number of Seats: 5
Fare of Vehicle: 500 PKR
```



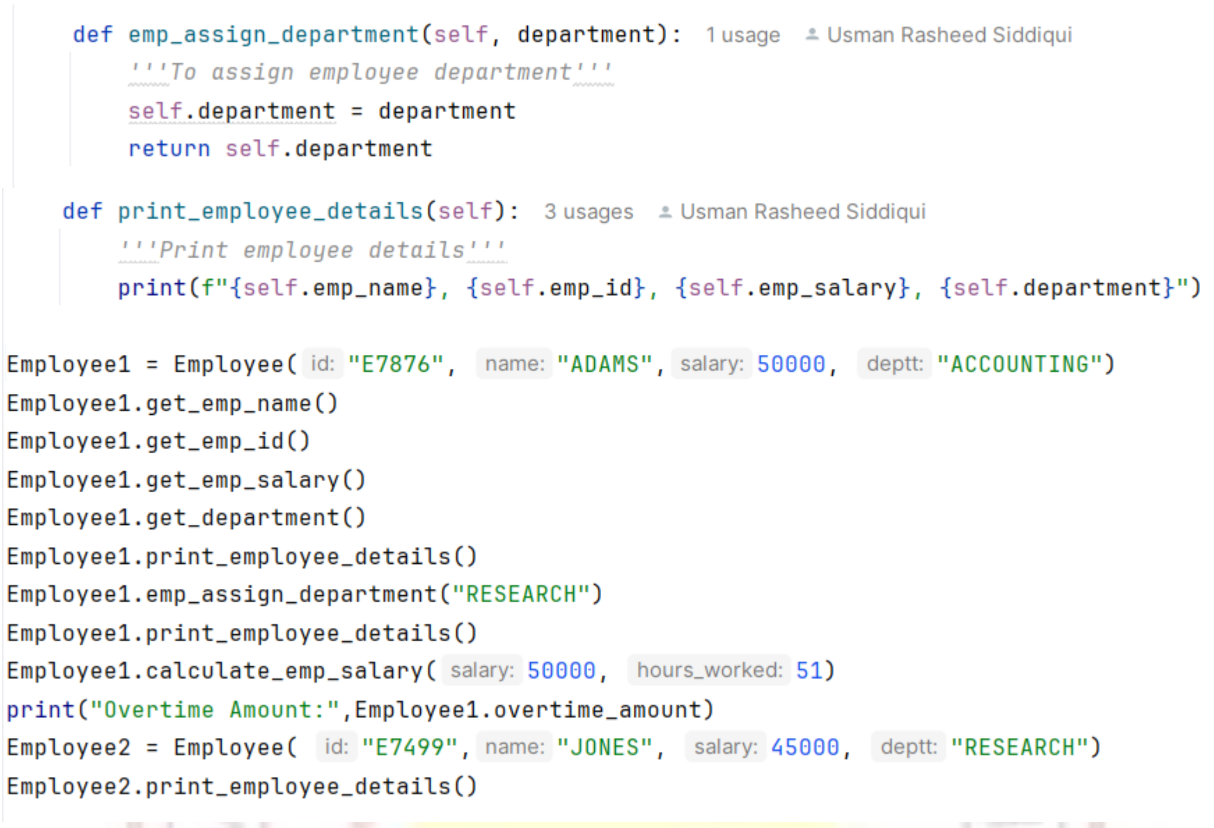
```
Enter name of vehicle: Sportage
Enter max speed of vehicle: 240
Enter number of seats: 5
Enter mileage of vehicle: 9650
Enter color of vehicle: Blue
Max Speed: 240 km\h
Name of Vehicle: Sportage
Number of Seats: 5
Fare of Vehicle: 500 PKR
```

PRACTICE PROBLEM 5:

INPUT:

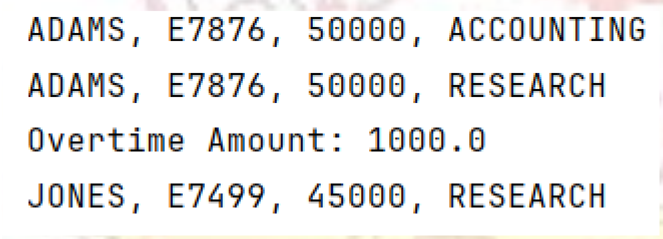


```
class Employee:   2 usages   Usman Rasheed Siddiqui
    '''Class to check employee information'''
    def __init__(self, id, name, salary, deptt):   Usman Rasheed Siddiqui
        self.set_emp_id(id)     # Employee ID
        self.set_emp_name(name)     # Employee Name
        self.set_emp_salary(salary)     # Employee Salary
        self.set_department(deptt)      # Employee Department

    # Setter Methods:
    def set_emp_id(self, id):   1 usage   Usman Rasheed Siddiqui
        '''Set Employee ID'''
        self.emp_id = id
    def set_emp_name(self, name):   1 usage   Usman Rasheed Siddiqui
        '''Set Employee Name'''
        self.emp_name = name
    def set_emp_salary(self, salary):   1 usage   Usman Rasheed Siddiqui
        '''Set Employee Salary'''
        self.emp_salary = salary
    def set_department(self, deptt):   1 usage   Usman Rasheed Siddiqui
        '''Set Employee Department'''
        self.department = deptt
    # Getter Methods:
    def get_emp_id(self):   1 usage   Usman Rasheed Siddiqui
        '''Get Employee ID'''
        return self.emp_id
    def get_emp_name(self):   1 usage   Usman Rasheed Siddiqui
        '''Get Employee Name'''
        return self.emp_name
    def get_emp_salary(self):   1 usage   Usman Rasheed Siddiqui
        '''Get Employee Salary'''
        return self.emp_salary
    def get_department(self):   1 usage   Usman Rasheed Siddiqui
        '''Get Employee Department'''
        return self.department

    def calculate_emp_salary(self, salary, hours_worked):   1 usage   Usman Rasheed Siddiqui
        '''To calculate employee salary'''
        self.hours_worked = hours_worked     # No. of hours worked
        if self.hours_worked > 50:
            overtime = hours_worked - 50
            self.overtime_amount = (overtime * (salary / 50))

        return self.overtime_amount
```



```
    def emp_assign_department(self, department):  1 usage  Usman Rasheed Siddiqui
        '''To assign employee department'''
        self.department = department
        return self.department

    def print_employee_details(self):  3 usages  Usman Rasheed Siddiqui
        '''Print employee details'''
        print(f"{self.emp_name}, {self.emp_id}, {self.emp_salary}, {self.department}")

Employee1 = Employee( id: "E7876",  name: "ADAMS", salary: 50000,  deptt: "ACCOUNTING")
Employee1.get_emp_name()
Employee1.get_emp_id()
Employee1.get_emp_salary()
Employee1.get_department()
Employee1.print_employee_details()
Employee1.emp_assign_department("RESEARCH")
Employee1.print_employee_details()
Employee1.calculate_emp_salary( salary: 50000,  hours_worked: 51)
print("Overtime Amount:",Employee1.overtime_amount)
Employee2 = Employee( id: "E7499", name: "JONES",  salary: 45000,  deptt: "RESEARCH")
Employee2.print_employee_details()
```

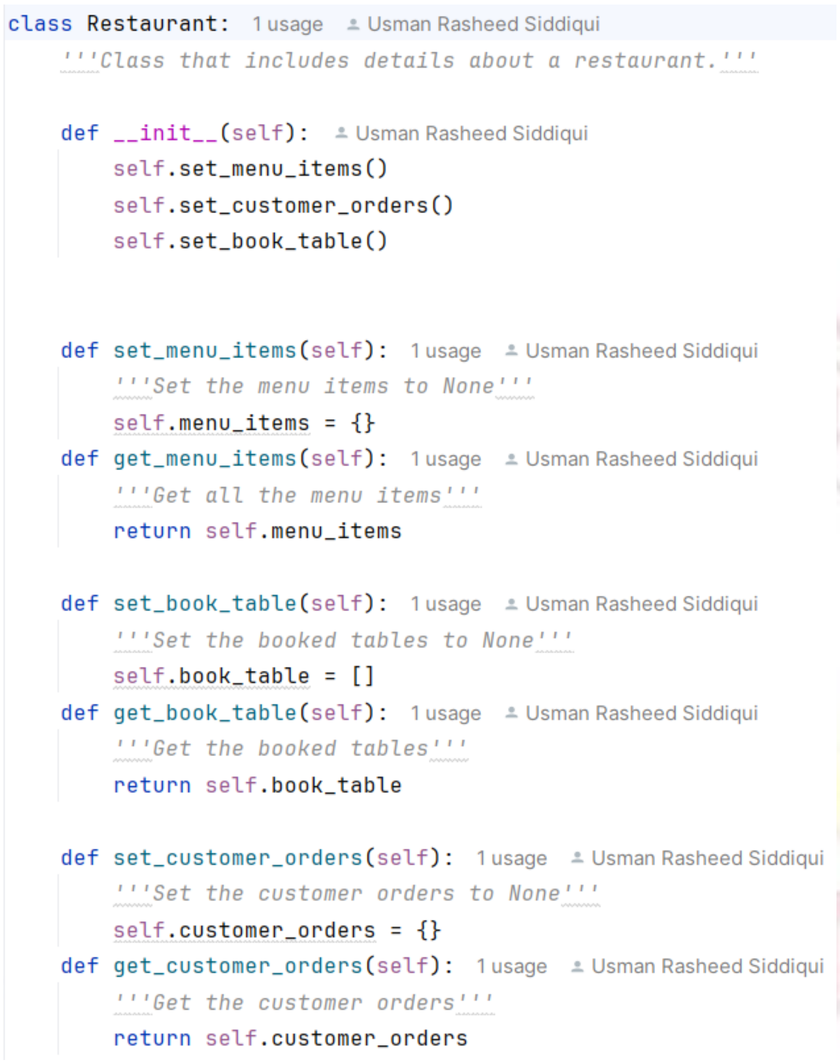
OUTPUT:



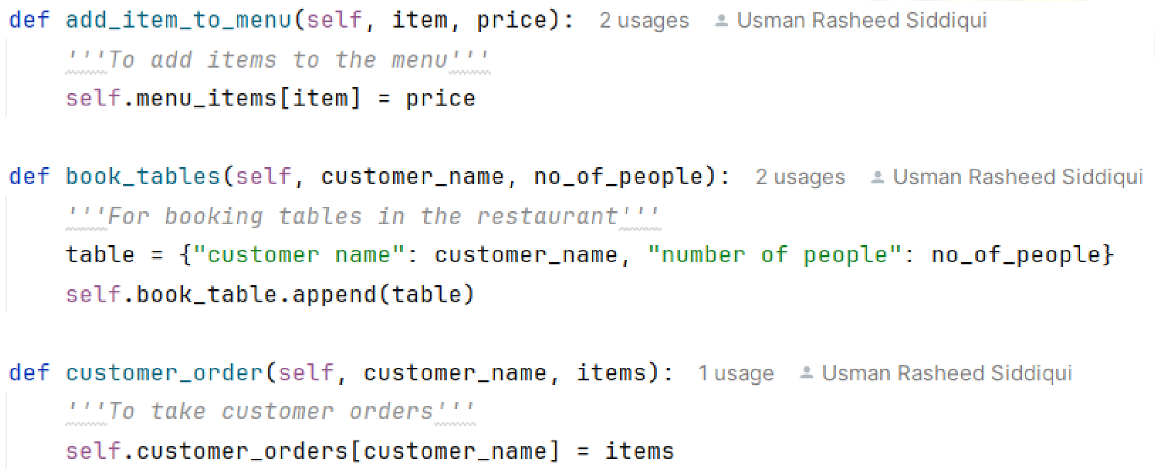
```
ADAMS, E7876, 50000, ACCOUNTING
ADAMS, E7876, 50000, RESEARCH
Overtime Amount: 1000.0
JONES, E7499, 45000, RESEARCH
```

PRACTICE PROBLEM 6:

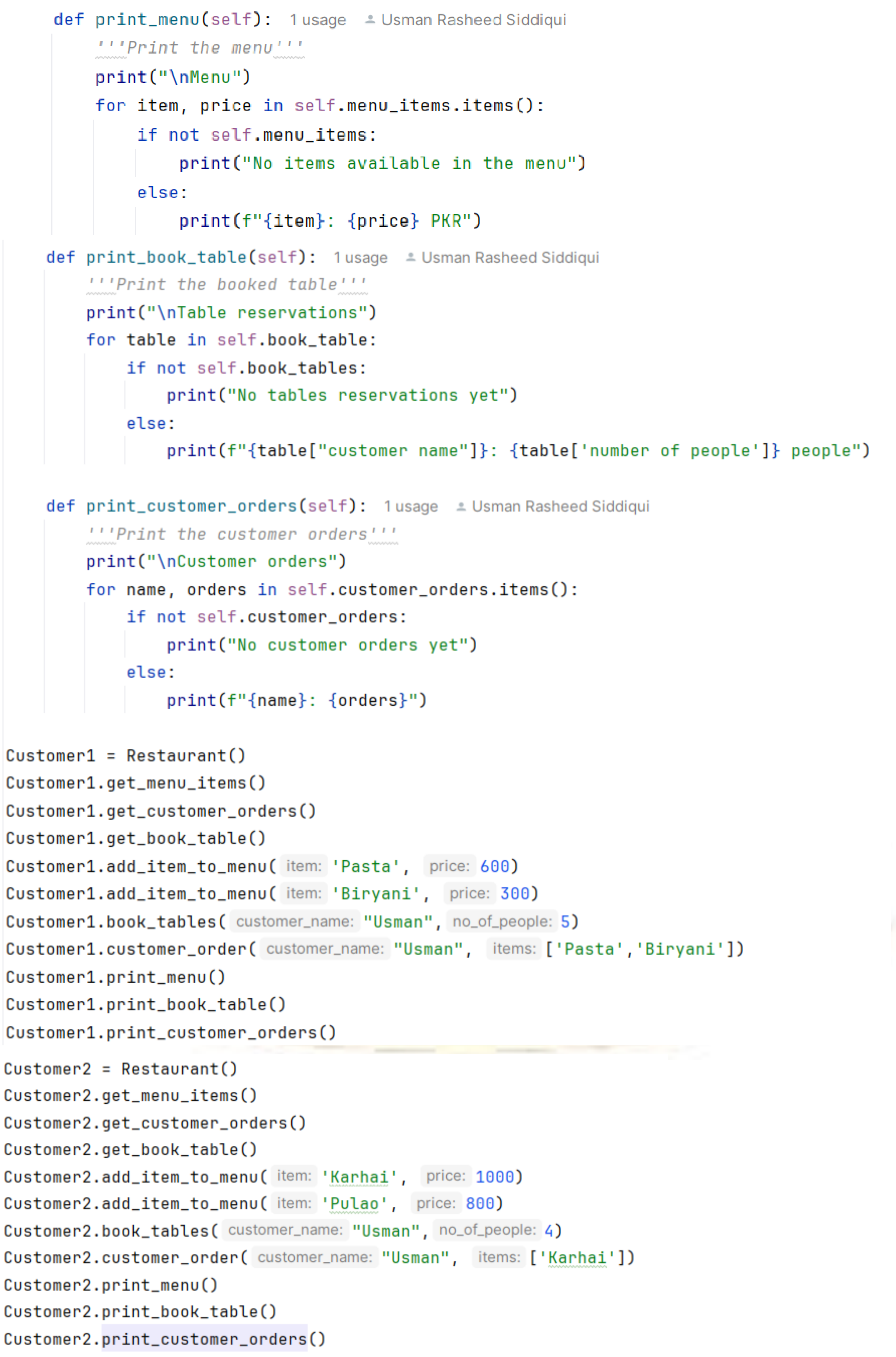
INPUT:



```
class Restaurant:  1 usage  Usman Rasheed Siddiqui
    '''Class that includes details about a restaurant.'''

    def __init__(self):  Usman Rasheed Siddiqui
        self.set_menu_items()
        self.set_customer_orders()
        self.set_book_table()


    def set_menu_items(self):  1 usage  Usman Rasheed Siddiqui
        '''Set the menu items to None'''
        self.menu_items = {}
    def get_menu_items(self):  1 usage  Usman Rasheed Siddiqui
        '''Get all the menu items'''
        return self.menu_items

    def set_book_table(self):  1 usage  Usman Rasheed Siddiqui
        '''Set the booked tables to None'''
        self.book_table = []
    def get_book_table(self):  1 usage  Usman Rasheed Siddiqui
        '''Get the booked tables'''
        return self.book_table

    def set_customer_orders(self):  1 usage  Usman Rasheed Siddiqui
        '''Set the customer orders to None'''
        self.customer_orders = {}
    def get_customer_orders(self):  1 usage  Usman Rasheed Siddiqui
        '''Get the customer orders'''
        return self.customer_orders
```

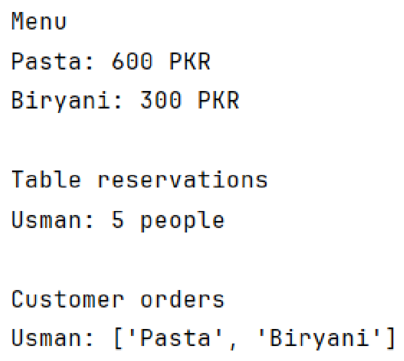


```
    def add_item_to_menu(self, item, price):  2 usages  Usman Rasheed Siddiqui
        '''To add items to the menu'''
        self.menu_items[item] = price

    def book_tables(self, customer_name, no_of_people):  2 usages  Usman Rasheed Siddiqui
        '''For booking tables in the restaurant'''
        table = {"customer name": customer_name, "number of people": no_of_people}
        self.book_table.append(table)

    def customer_order(self, customer_name, items):  1 usage  Usman Rasheed Siddiqui
        '''To take customer orders'''
        self.customer_orders[customer_name] = items
```

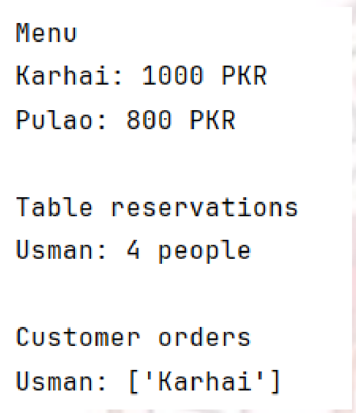


```
    def print_menu(self):  1 usage  Usman Rasheed Siddiqui
        '''Print the menu'''
        print("\nMenu")
        for item, price in self.menu_items.items():
            if not self.menu_items:
                print("No items available in the menu")
            else:
                print(f"{item}: {price} PKR")
    def print_book_table(self):  1 usage  Usman Rasheed Siddiqui
        '''Print the booked table'''
        print("\nTable reservations")
        for table in self.book_table:
            if not self.book_tables:
                print("No tables reservations yet")
            else:
                print(f"{table["customer name"]}: {table['number of people']} people")

    def print_customer_orders(self):  1 usage  Usman Rasheed Siddiqui
        '''Print the customer orders'''
        print("\nCustomer orders")
        for name, orders in self.customer_orders.items():
            if not self.customer_orders:
                print("No customer orders yet")
            else:
                print(f"{name}: {orders}")

Customer1 = Restaurant()
Customer1.get_menu_items()
Customer1.get_customer_orders()
Customer1.get_book_table()
Customer1.add_item_to_menu( item: 'Pasta',  price: 600)
Customer1.add_item_to_menu( item: 'Biryani',  price: 300)
Customer1.book_tables( customer_name: "Usman", no_of_people: 5)
Customer1.customer_order( customer_name: "Usman",  items: ['Pasta','Biryani'])
Customer1.print_menu()
Customer1.print_book_table()
Customer1.print_customer_orders()

Customer2 = Restaurant()
Customer2.get_menu_items()
Customer2.get_customer_orders()
Customer2.get_book_table()
Customer2.add_item_to_menu( item: 'Karhai',  price: 1000)
Customer2.add_item_to_menu( item: 'Pulao',  price: 800)
Customer2.book_tables( customer_name: "Usman", no_of_people: 4)
Customer2.customer_order( customer_name: "Usman",  items: ['Karhai'])
Customer2.print_menu()
Customer2.print_book_table()
Customer2.print_customer_orders()
```

OUTPUT:



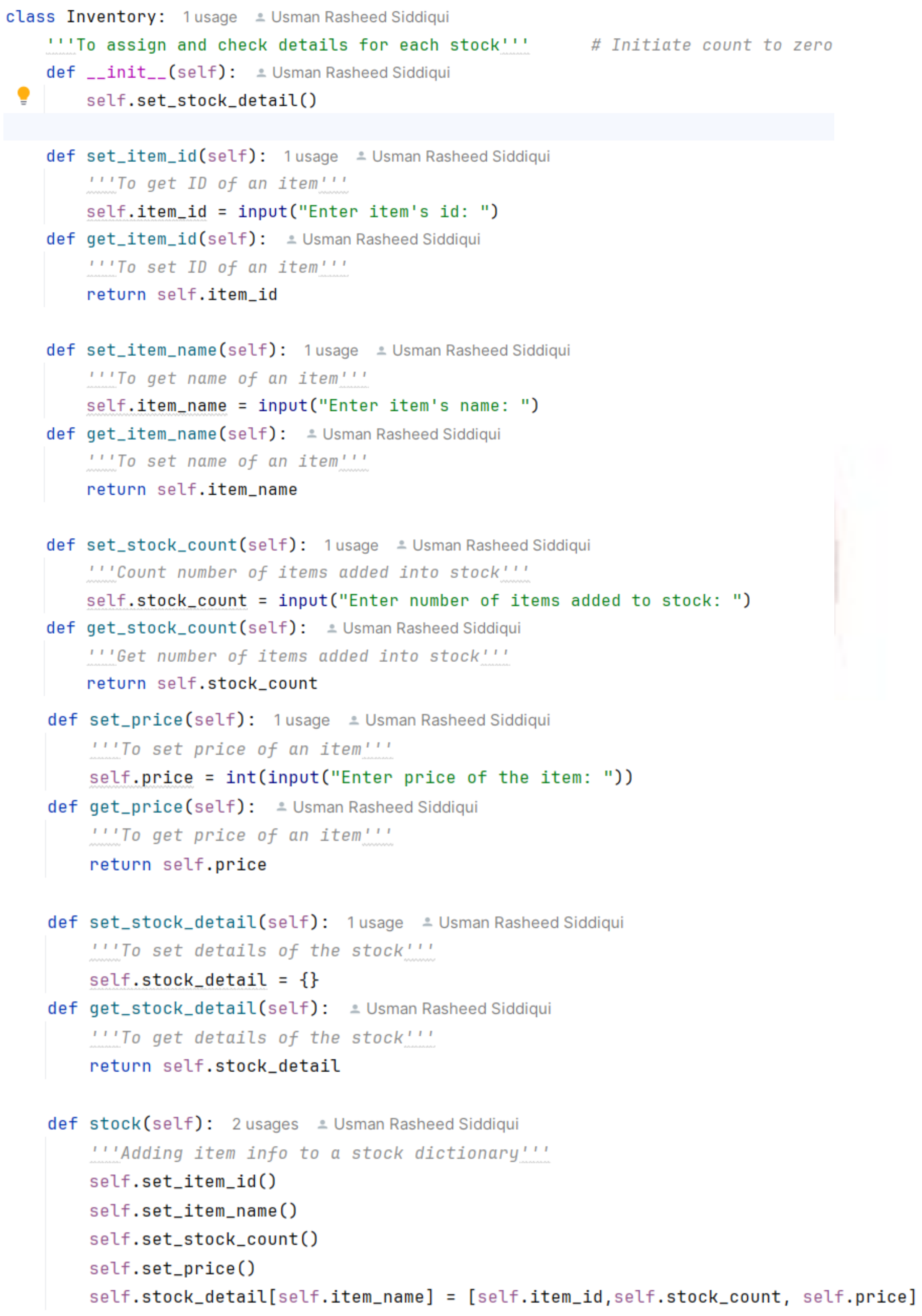
```
Menu
Pasta: 600 PKR
Biryani: 300 PKR

Table reservations
Usman: 5 people

Customer orders
Usman: ['Pasta', 'Biryani']
```



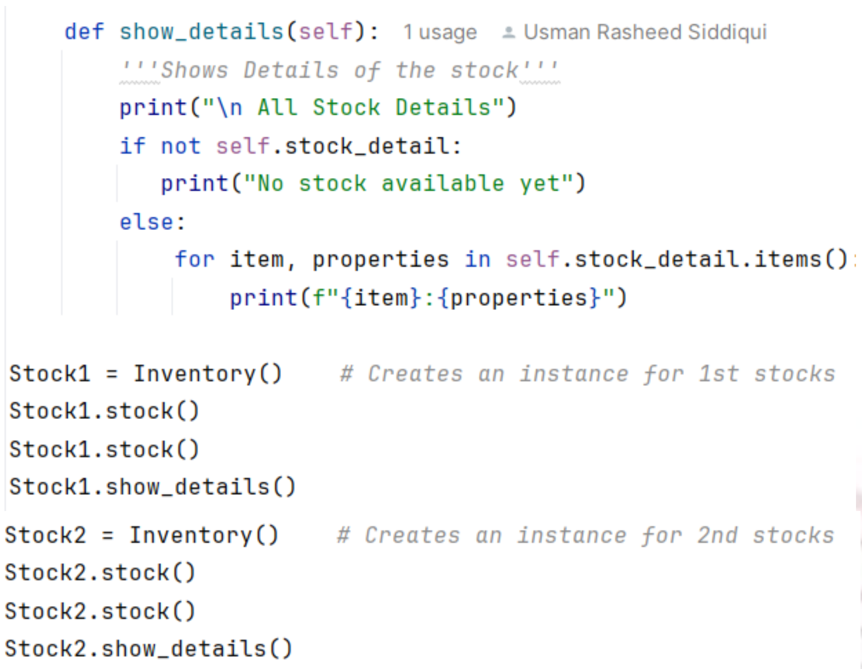
```
Menu
Karhai: 1000 PKR
Pulao: 800 PKR

Table reservations
Usman: 4 people

Customer orders
Usman: ['Karhai']
```

PRACTICE PROBLEM 7:

INPUT:

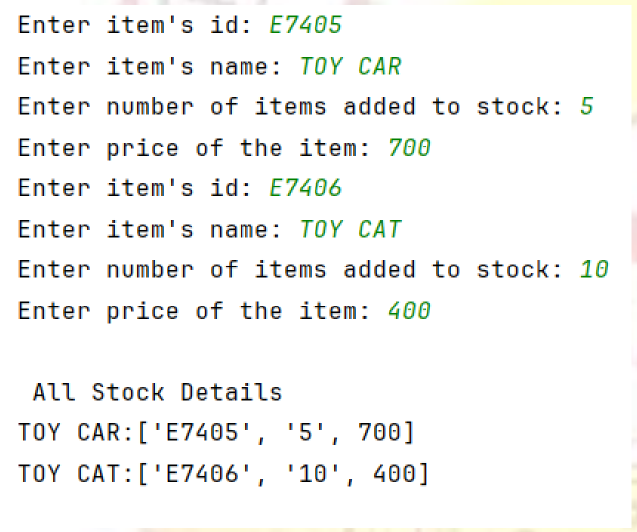


```
class Inventory:  1 usage  Usman Rasheed Siddiqui
    '''To assign and check details for each stock'''      # Initiate count to zero
    def __init__(self):  Usman Rasheed Siddiqui
        self.set_stock_detail()

    def set_item_id(self):  1 usage  Usman Rasheed Siddiqui
        '''To get ID of an item'''
        self.item_id = input("Enter item's id: ")
    def get_item_id(self):  Usman Rasheed Siddiqui
        '''To set ID of an item'''
        return self.item_id

    def set_item_name(self):  1 usage  Usman Rasheed Siddiqui
        '''To get name of an item'''
        self.item_name = input("Enter item's name: ")
    def get_item_name(self):  Usman Rasheed Siddiqui
        '''To set name of an item'''
        return self.item_name

    def set_stock_count(self):  1 usage  Usman Rasheed Siddiqui
        '''Count number of items added into stock'''
        self.stock_count = input("Enter number of items added to stock: ")
    def get_stock_count(self):  Usman Rasheed Siddiqui
        '''Get number of items added into stock'''
        return self.stock_count

    def set_price(self):  1 usage  Usman Rasheed Siddiqui
        '''To set price of an item'''
        self.price = int(input("Enter price of the item: "))
    def get_price(self):  Usman Rasheed Siddiqui
        '''To get price of an item'''
        return self.price

    def set_stock_detail(self):  1 usage  Usman Rasheed Siddiqui
        '''To set details of the stock'''
        self.stock_detail = {}
    def get_stock_detail(self):  Usman Rasheed Siddiqui
        '''To get details of the stock'''
        return self.stock_detail

    def stock(self):  2 usages  Usman Rasheed Siddiqui
        '''Adding item info to a stock dictionary'''
        self.set_item_id()
        self.set_item_name()
        self.set_stock_count()
        self.set_price()
        self.stock_detail[self.item_name] = [self.item_id,self.stock_count, self.price]
```

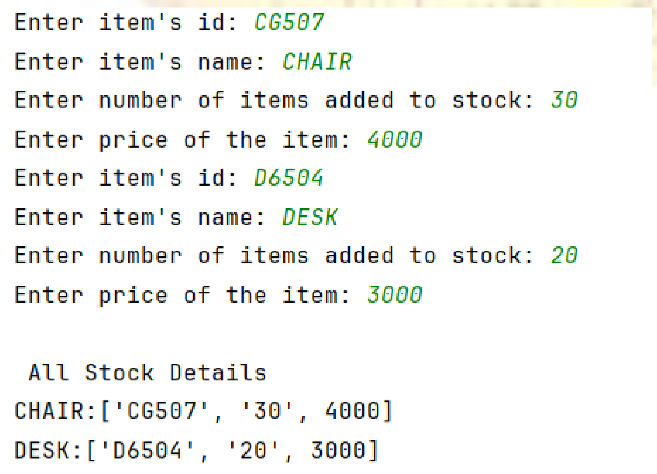


```
    def show_details(self):
        '''Shows Details of the stock'''
        print("\n All Stock Details")
        if not self.stock_detail:
            print("No stock available yet")
        else:
            for item, properties in self.stock_detail.items():
                print(f"{item}:{properties}")

Stock1 = Inventory()    # Creates an instance for 1st stocks
Stock1.stock()
Stock1.stock()
Stock1.show_details()
Stock2 = Inventory()    # Creates an instance for 2nd stocks
Stock2.stock()
Stock2.stock()
Stock2.show_details()
```

OUTPUT:



```
Enter item's id: E7405
Enter item's name: TOY CAR
Enter number of items added to stock: 5
Enter price of the item: 700
Enter item's id: E7406
Enter item's name: TOY CAT
Enter number of items added to stock: 10
Enter price of the item: 400

 All Stock Details
TOY CAR:['E7405', '5', 700]
TOY CAT:['E7406', '10', 400]
```



```
Enter item's id: CG507
Enter item's name: CHAIR
Enter number of items added to stock: 30
Enter price of the item: 4000
Enter item's id: D6504
Enter item's name: DESK
Enter number of items added to stock: 20
Enter price of the item: 3000

 All Stock Details
CHAIR:['CG507', '30', 4000]
DESK:['D6504', '20', 3000]
```