NAME: B. MANUDEEP REDDY

HT. NO: 2303A52380

ASS.NO : 6.4

1) PROMPT:
Create a Python class named Student with attributes name, roll_number, and marks.

Complete the class by adding:

1. A method to display student details

2. A method to check if the student's marks are above the class average

Use self attributes and if-else conditions.

Assume class average is passed as a parameter.

Also provide a sample usage and output.

Analyze the code for correctness and edge cases before generating final output.

```python
class Student:
    def __init__(self, name, roll_number, marks):
        self.name = name
        self.roll_number = roll_number
        self.marks = marks

    def display_details(self):
        print("Student Name:", self.name)
        print("Roll Number:", self.roll_number)
        print("Marks:", self.marks)

    def check_performance(self, class_average):
        if self.marks > class_average:
            return "Student performance is above class average"
        else:
            return "Student performance is below class average"


# Sample usage
student1 = Student("Rahul", 101, 85)
class_avg = 75

student1.display_details()
print(student1.check_performance(class_avg))
```

OUTPUT:

```
Student Name: Rahul
Roll Number: 101
Marks: 85
Student performance is above class average
```

2)

PROMPT:

Write a Python for loop that iterates over a list of integer sensor readings.

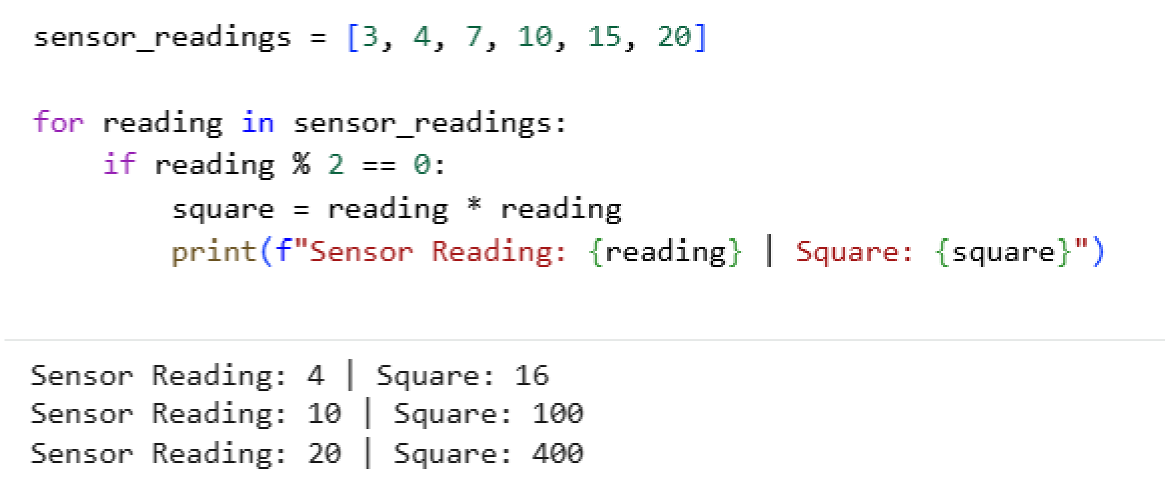
Inside the loop:

- Identify even numbers using the modulus operator

- Calculate the square of each even number

- Print the result in a clear, readable format

Use conditional statements and analyze correctness before final output.

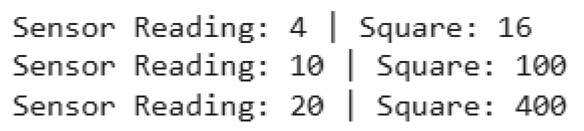


```python
sensor_readings = [3, 4, 7, 10, 15, 20]

for reading in sensor_readings:
    if reading % 2 == 0:
        square = reading * reading
        print(f"Sensor Reading: {reading} | Square: {square}")
```

```
Sensor Reading: 4 | Square: 16
Sensor Reading: 10 | Square: 100
Sensor Reading: 20 | Square: 400
```

OUTPUT:



```
Sensor Reading: 4 | Square: 16
Sensor Reading: 10 | Square: 100
Sensor Reading: 20 | Square: 400
```

3)

PROMPT:

Create a Python class named BankAccount with attributes account_holder and balance.

Complete the class by adding methods to:

- Deposit money into the account

- Withdraw money from the account

- Prevent withdrawal if balance is insufficient using if-else conditions

Use self attributes and print user-friendly messages.

Analyze the code for correctness and edge cases before generating final output.

```python
class BankAccount:
    def __init__(self, account_holder, balance):
        self.account_holder = account_holder
        self.balance = balance

    def deposit(self, amount):
        self.balance += amount
        print(f"Amount deposited: {amount}")
        print(f"Updated balance: {self.balance}")

    def withdraw(self, amount):
        if amount <= self.balance:
            self.balance -= amount
            print(f"Amount withdrawn: {amount}")
            print(f"Remaining balance: {self.balance}")
        else:
            print("Insufficient balance. Withdrawal cannot be processed.")


# Sample usage
account = BankAccount("Anita", 5000)

account.deposit(2000)
account.withdraw(3000)
account.withdraw(5000)
```

OUTPUT:

```
Amount deposited: 2000
Updated balance: 7000
Amount withdrawn: 3000
Remaining balance: 4000
Insufficient balance. Withdrawal cannot be processed.
```

4)

PROMPT:

Given a list of dictionaries where each dictionary represents a student with

name and score, generate a while loop that:

- Iterates through the list using an index

- Checks if the student's score is greater than 75

- Prints the names of eligible students in a clean format

Ensure correct index handling and conditional checks.

```
students = [
    {"name": "Amit", "score": 82},
    {"name": "Neha", "score": 68},
    {"name": "Ravi", "score": 90},
    {"name": "Sita", "score": 74}
]

index = 0

print("Students eligible for scholarship:")

while index < len(students):
    if students[index]["score"] > 75:
        print(students[index]["name"])
    index += 1
```

OUTPUT:

```
Students eligible for scholarship:
Amit
Ravi
```

5)

PROMPT:

Create a Python class named ShoppingCart with an empty list to store items.

Each item should have name, price, and quantity.

Generate methods to:

- Add items to the cart

- Remove items from the cart

- Calculate total bill using a loop

- Apply a discount if total exceeds a certain amount

Use loops, if-else conditions, and self attributes.

Analyze for correctness before producing final output.

```python
class ShoppingCart:
    def __init__(self):
        self.items = []

    def add_item(self, name, price, quantity):
        self.items.append({
            "name": name,
            "price": price,
            "quantity": quantity
        })
        print(f"Item added: {name}")

    def remove_item(self, name):
        for item in self.items:
            if item["name"] == name:
                self.items.remove(item)
                print(f"Item removed: {name}")
                return
        print("Item not found in cart")

    def calculate_total(self):
        total = 0
        for item in self.items:
            total += item["price"] * item["quantity"]

        if total > 5000:
            discount = total * 0.10
            total -= discount
            print(f"Discount applied: {discount}")

        print(f"Total bill amount: {total}")
        return total


# Sample usage
cart = ShoppingCart()
cart.add_item("Laptop", 40000, 1)
cart.add_item("Mouse", 500, 2)
cart.remove_item("Mouse")
cart.calculate_total()
```

OUTPUT:

```
Item added: Laptop
Item added: Mouse
Item removed: Mouse
Discount applied: 4000.0
Total bill amount: 36000.0
36000.0
```