Question 1

a) Write a code in Python for the following conditions:

Suppose there are different slabs of discount on a purchase:

- **20% on amount exceeding 10,000,**
- **10% for amount between 5,000 and 10,000,**
- **5% if it is between 1,000 and 5,000,**
- **No discount if the amount is less than 1,000.**

Solution:

```
def calculate_discount(amount):
    if amount < 1000:
        return 0
    elif amount >= 1000 and amount < 5000:
        return 0.05 * amount
    elif amount >= 5000 and amount <= 10000:
        return 0.10 * amount
    else:
        return 0.20 * amount
purchase_amount = float(input("Enter the purchase amount: "))
discount = calculate_discount(purchase_amount)
print(f"Discount: {discount}")
```

b) Prints all letters except 'a' and 'e' for the following string: "Venkataramarajuvaripeta"

Solution:

```
string = "Venkataramarajuvaripeta"

result = ""


for char in string:

   if char.lower() not in ['a', 'e']:

      result += char

print(result)
```

Question 2

a) Write a code in Python to display the list of numbers from 1 to 10 and print the result for the following index values:

1. **[1:6]**
2. **[-5:1]**

Solution:

```
numbers = list(range(1, 11))

print(numbers[1:6])

print(numbers[-5:1])
```

b) Write a code in Python to display the result of the sum of square numbers in the range of 1 to 10.

Solution:

```
sum_of_squares = sum([x**2 for x in range(1, 11)])

print(sum_of_squares)
```

Question 3

a) List out various programming editors used for executing Python code.

Answer: Some popular programming editors used for executing Python code include:

1. Visual Studio Code (VS Code)
2. PyCharm -
3. Jupyter Notebook -
4. Sublime Text -
5. Atom -
6. Notepad++
7. Spyder -
8. Thonny –

b) What is the use of the scope operator (::) in Python?

Answer:

```
global_var =


def my_function():
    local_var =
    print(local_var)


my_function()
print(global_var)
```

CIA 2

Question 1

a) Apply the following file operation methodology to implement code in Python: File Open

Solution:

```
with open("sample.txt", "r") as file:
    content = file.read()
    print("File contents:")
    print(content)
```

b) Apply the following file operation methodology to implement code in Python: File Write

Solution:

```
with open("output.txt", "w") as file:
    file.write("Hello, this is written to the file!\n")
    file.write("This is another line.\n")
print("Data has been written to 'output.txt'.")
```

Question 2

a) Apply class methodology to implement code in Python: Create a class FYCY with two data members (roll_number and student_name). Create an object to pass values into it.

Solution:

```
class FYCY:
    def __init__(self, roll_number, student_name):
        self.roll_number = roll_number
        self.student_name = student_name
```

```
    def display_info(self):
        print(f"Roll Number: {self.roll_number}")
        print(f"Student Name: {self.student_name}")
student1 = FYCY(101, "John Doe")
student1.display_info()
```

b) Apply class methodology to implement code in Python: Create two classes named SET and SCM with two data members (CSE and BCom) and create an object to pass values into it.

Solution:

```
class Department:
    def __init__(self, name, CSE, BCom):
        self.name = name
        self.CSE = CSE
        self.BCom = BCom
    def display_info(self):
        print(f"{self.name} Class Information:")
        print(f"CSE: {self.CSE}")
        print(f"BCom: {self.BCom}")
Department("SET", 50, 40).display_info()
print()
Department("SCM", 60, 55).display_info()
```

Question 3

a) Explain the use of Matplotlib in Python with an example.

Solution:

Matplotlib is a popular plotting library in Python used for creating static, interactive, and animated visualizations. It provides a wide range of plotting functions to create graphs, charts, histograms, etc.

```
import matplotlib.pyplot as plt

plt.plot([1, 2, 3, 4, 5], [2, 4, 6, 8, 10])
plt.title("Line Graph")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()
```

b) State the importance of the math library in Python.

Solution:

It helps you perform calculations like square roots, trigonometry, logarithms, and powers easily.

Essential for fields like data science, machine learning, finance, physics, engineering, etc.

Built-in functions avoid errors and give faster, more reliable results than manual calculations.

For ex

```
import math
print(math.sqrt(16))     # Output: 4.0
print(math.pi)           # Output: 3.141592653589793
```

```
print(math.factorial(5)) # Output: 120
```