Post Experiment Exercise:

Question 1:

C++ Code:

```
#include <iostream>
using namespace std;

// Base class: Staff
class Staff
{
protected:
  int code;
  string name;

public:
  // Constructor to initialize code and name
  Staff(int code, string name) : code(code), name(name) {}

  // Member function to get the code
  int getCode() { return code; }

  // Member function to get the name
  string getName() { return name; }
};

// Derived class: Teacher (inherits from Staff)
class Teacher : public Staff
{
protected:
  string subject;
  string publication[100];

public:
  // Constructor to initialize code, name, subject, and publications
  Teacher(int code, string name, string subject, string publication[100])
: Staff(code, name), subject(subject)
  {
    // Copy the publications array to the class member array
    for (int i = 0; i < 100; i++)
    {
      this->publication[i] = publication[i];
    }
  }

  // Member function to get the subject
  string getSubject() { return subject; }

  // Member function to get a specific publication by index
```

```
  string getPublication(int index) { return publication[index]; }
};

// Derived class: Officer (inherits from Staff)
class Officer : public Staff
{
protected:
  int grade;

public:
  // Constructor to initialize code, name, and grade
  Officer(int code, string name, int grade) : Staff(code, name),
grade(grade) {}
};

// Derived class: Typist (inherits from Staff)
class Typist : public Staff
{
protected:
  int speed;

public:
  // Constructor to initialize code, name, and speed
  Typist(int code, string name, int speed) : Staff(code, name),
speed(speed) {}

  // Member function to get the typing speed
  int getSpeed() { return speed; }
};

// Derived class: Regular (inherits from Typist)
class Regular : public Typist
{
protected:
  float salary;

public:
  // Constructor to initialize code, name, speed, and salary
  Regular(int code, string name, int speed, float salary) : Typist(code,
name, speed), salary(salary) {}

  // Member function to get the salary
  float getSalary() { return salary; }
};

// Derived class: Casual (inherits from Typist)
class Casual : public Typist
{
protected:
```

```
  float dailyWage;

public:
  // Constructor to initialize code, name, speed, and daily wage
  Casual(int code, string name, int speed, float dailyWage) :
Typist(code, name, speed), dailyWage(dailyWage) {}

  // Member function to get the daily wage
  float getDailyWage() { return dailyWage; }
};

int main()
{
  // Data for Teacher's publication
  string data[100] = {"Computer Paradigms"};

  // Create a Teacher object
  Teacher teacher1(123, "Ms. Mrinmoyee", "PCPF", data);

  // Output Teacher's details
  cout << endl
       << "Teacher Code: " << teacher1.getCode()
       << endl
       << "Teacher Name: " << teacher1.getName()
       << endl
       << "Subject: " << teacher1.getSubject()
       << endl
       << "Publication: " << teacher1.getPublication(0);

  return 0;
}
```

Output:

```
TERMINAL    PROBLEMS    OUTPUT    DEBUG CONSOLE    SERIAL MONITOR    COMMENTS

PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\1. Inheritance using Java & C++
nheritance using Java & C++\C++\" ; if ($?) { g++ pe1.cpp -o pe1 } ; if ($?)

Teacher Code: 123
Teacher Name: Ms. Mrinmoyee
Subject: PCPF
Publication: Computer Paradigms

PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\1. Inheritance using Java & C++
```

Post Experiment Exercise:

Question 3:

C++ Code:

```
#include <iostream>
using namespace std;

// Base class: Reverse
class Reverse
{
protected:
  string userString;
  string reversedString;

public:
  // Constructor to initialize userString
  Reverse(string userString) : userString(userString) {}

  // Function to reverse the string
  string reverseString(string data)
  {
    for (int i = 0; i < data.length(); i++)
    {
      reversedString += data[data.length() - i - 1];
    }
    return reversedString;
  }
};

// Derived class: Display (inherits from Reverse)
class Display : public Reverse
{
public:
  // Constructor to initialize userString through the base class
constructor
  Display(string userString) : Reverse(userString) {}

  // Function to display the userString
  void print()
  {
    cout << "Original String: " << userString << endl;
  }
};

int main()
{
  string str;
  cout << endl<< "Enter a string: ";
```

```
  cin >> str;

  // Create a Display object
  Display display(str);

  // Call the print function to display the original string
  display.print();

  // Call the reverseString function from the base class and display the
reversed string
  string reversed = display.reverseString(str);
  cout << "Reversed String: " << reversed << endl<<endl;

  return 0;
}
```

Output:

```
TERMINAL    PROBLEMS    OUTPUT    DEBUG CONSOLE    SERIAL MONITOR    COMMENTS

PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\1. Inheritance using Java &
nheritance using Java & C++\C++\" ; if ($?) { g++ pe3.cpp -o pe3 } ; if

Enter a string: AJAYKUMAR
Original String: AJAYKUMAR
Reversed String: RAMUKYAJA

PS C:\Users\Ajay kumar\Desktop\SEIT\PCPF\Lab\1. Inheritance using Java &
```