Tutorial 3 – Class and Object

1. **(Class Definition and Access Specifiers)** Suppose we need to develop a system to manage student records in a university. The foundation of such a system is to implement a `Student` class with the following members.
   - Private members
     - `string name`: the name of the student
     - `int age`: the age of the student
     - `double gpa`: the GPA of the student
   - Public members
     - A constructor that initializes the `name, age` and `gpa`
     - A method `displayDetails()` that prints out the student details
     - Getters – return private member values
       - `getName()`
       - `getAge()`
       - `getGPA()`
     - Setters – modify private member values
       - `setName(string newName)`
       - `setAge(int newAge)`
       - `setGPA(double newGPA)`

   You are requested to use the following code to get started. It also includes the test cases.

```cpp
#include <iostream>
#include <string>

class Student {
private: // TODO: define the private members here

public:
    // Constructor
    Student(std::string studentName, int studentAge, double studentGPA) {
        // TODO: Define the constructor
    }

    // Display function
    void displayDetails() const {
        std::cout << "Student Name: " << name << std::endl;
        std::cout << "Age: " << age << std::endl;
        std::cout << "GPA: " << gpa << std::endl;
    }

    // Getters
    // TODO: Implement the getters here
```

```
    // Setters
    // TODO: Implement the setters here

};

int main() {
    // Creating Student objects
    Student student1("Alice", 20, 3.8);
    Student student2("Charlie", 19, 3.5);

    // Display details of students
    std::cout << "Initial Student Details:\n";
    student1.displayDetails();
    std::cout << std::endl;
    student2.displayDetails();
    std::cout << std::endl;

    // Modify student1 details using setters
    student1.setName("Bob");
    student1.setAge(22);
    student1.setGPA(3.9);

    // Display updated details
    std::cout << "Updated Student Details:\n";
    student1.displayDetails();

    return 0;
}
```

Sample outputs should be:

```
Initial Student Details:
Student Name: Alice
Age: 20
GPA: 3.8

Student Name: Charlie
Age: 19
GPA: 3.5

Updated Student Details:
Student Name: Bob
Age: 22
GPA: 3.9
```

2. **(Copy Constructor)** Andy implemented a class called `VisitorCounter` to record and display the visitor count for a website. Below is his C++ implementation and the testing code.

   1) Please read the code and point out the possible issues in his implementation;

   2) If you are requested to implement the class, how would you fix the issues here?

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

class VisitorCounter {
private:
    int* count;  // Pointer to dynamically allocated memory for visit count

public:
    VisitorCounter(int initialCount) {
        count = new int(initialCount);
        cout << "Constructor called with the count as " << *count << endl;
    }

    ~VisitorCounter() {
        cout << "Destructor called with the count being " << *count << endl;
        delete count;
    }

    void increment() {
        (*count)++;
    }

    void display() const {
        cout << "Visitor Count: " << *count << endl;
    }
};

int main() {
    VisitorCounter counter(10);
    cout << "\nOriginal Counter:\n";
    counter.display();

    // Copy the counter
    VisitorCounter counterCopy = counter;
    cout << "counterCopy:\n";
    counterCopy.display();

    // Increase copied object's count
    counterCopy.increment();
    counterCopy.increment();
    cout << "\nAfter modifying copied counter...\n";
    cout << "Original Counter: "<<endl;
```

```
    counter.display();
    cout << "counterCopy: "<<endl;
    counterCopy.display();
    cout <<endl <<endl;

    return 0;
}
```

3. **(`this` Operator and Method Chaining)** You are asked to implement a class called `Pen` to manage the records of pens in an online shopping platform. It allows the setting of the color and price of a pen, as well as display them. Below are the detailed members of this class.
   - Private members
     - `std::string color`: the color of the pen
     - `double price`: the price of the pen
   - Public members
     - Constructor that initializes the color to black and price to 10
     - `setColor()`, the method to set the color of the pen
     - `setPrice()`, the method to set the price of the pen
     - `display()`, the method to display the pen details

   The main function and the starting code of the Pen class are given below.

```
#include <iostream>
#include <string>
class Pen {
private:
    std::string color;
    double price;

public:
    Pen(std::string initialColor, double initialPrice) {
        color = initialColor;
        price = initialPrice;
    }

    Pen& setColor(std::string newColor) {
        // TO-DO: Write your code here
        //

    }

    Pen& setPrice(double newPrice) {
        // TO-DO: Write your code here
        //

    }
```

```
    void display() const {
        // TO-DO: Write your code here
        //

    }
};

int main() {
    // Creating a Pen object and using method chaining
    Pen myPen("Blue", 1.5);
    std::cout<< "The original color and price of the pen: " << std::endl;
    myPen.display();

    std::cout<< std::endl<<"The color and price of the pen after setting: "
<< std::endl;
    myPen.setColor("Red")
         .setPrice(2.0)
         .display();

    return 0;
}
```

Sample output should be:

```
The original color and price of the pen:
Pen Color: Blue
Price: $1.5

The color and price of the pen after setting:
Pen Color: Red
Price: $2
```

4. **(Friend Function)** You are asked to develop a `Box` class that allows an external function to access and display the **private dimensions** (`length, width, and height`) of a box. Since these attributes are private, they cannot be accessed directly from outside the class. Your task is to:
   - Create a Box class with private members: `length, width, and height`.
   - Declare and define a friend function `displayDimensions()` that can access private members and display them.
   - Define a member function `calculateVolume()` that calculates the volume of the box.

   Below are the details of the members of the class `Box`:
   - Private Members:

    - `double length`: the length of the box
    - `double width`: the width of the box
    - `double height`: the height of the box
- Public Members:
    - A constructor function that initializes `length, width, and height`.
    - A member function `calculateVolume()` that calculates the volume
    - Friend Function `displayDimensions(const Box& b)` that accesses and prints private attributes.

Below is the starting code for you.

```
#include <iostream>

class Box {
private:
    double length;
    double width;
    double height;

public:
    // Constructor to initialize the box dimensions
    Box(double l, double w, double h) {
        length = l;
        width = w;
        height = h;
    }

    // Member function that can access the private members
    void calculateVolume() {
        // Caclulate and display the volume of the box
        // TODO: Write your code here
        //


    }

    // Declare a friend function to display private members
    // TODO: Write your code here
    //

};


// Define the friend function (that can access private members of Box)
// TODO: Write your code here
//

int main() {
```

```
    // Creating a Box object
    Box myBox(5.0, 3.0, 2.0);

    // Friend function accessing private data
    displayDimensions(myBox);

    // Member function accessing private data
    myBox.calculateVolume();

    return 0;
}
```

Sample output:

```
Box Dimensions:
Length: 5
Width: 3
Height: 2

Box Volume: 30 cubic units
```