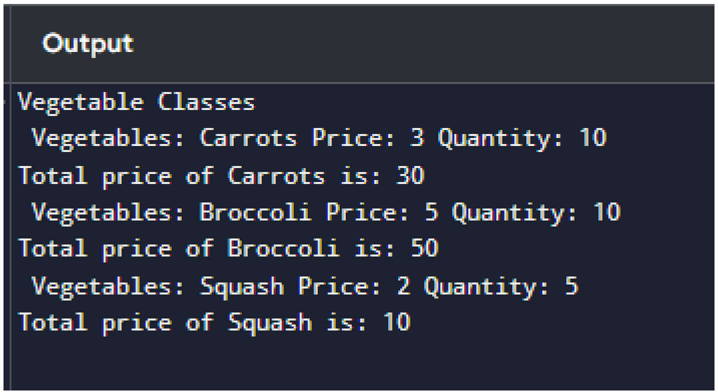
Activity No. 2.1	
Hands-on Activity 2.1 Arrays, Pointers and Dynamic Memory Allocation	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed: 8/7/25**
Section: CPE21S4	**Date Submitted: 8/7/25**
Name(s): Punay Heidee S.	**Instructor: Engr. Jimlord Quejado**

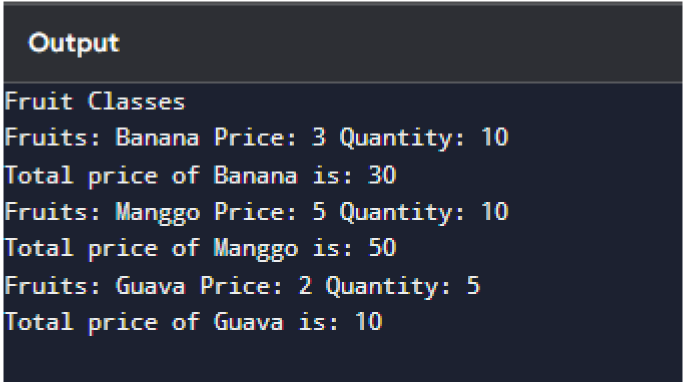
6. Output

Problem 1



Output

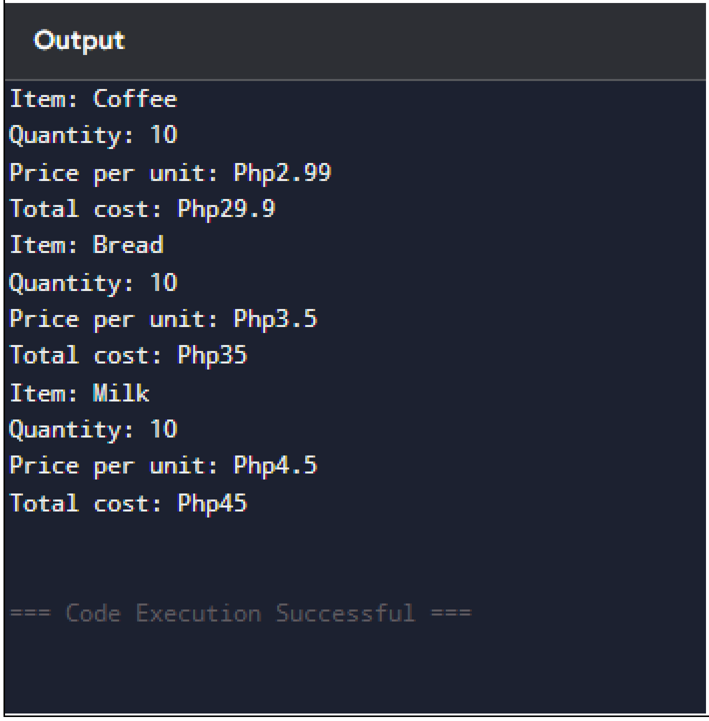
```
Vegetable Classes
 Vegetables: Carrots Price: 3 Quantity: 10
Total price of Carrots is: 30
 Vegetables: Broccoli Price: 5 Quantity: 10
Total price of Broccoli is: 50
 Vegetables: Squash Price: 2 Quantity: 5
Total price of Squash is: 10
```



Output

```
Fruit Classes
Fruits: Banana Price: 3 Quantity: 10
Total price of Banana is: 30
Fruits: Manggo Price: 5 Quantity: 10
Total price of Manggo is: 50
Fruits: Guava Price: 2 Quantity: 5
Total price of Guava is: 10
```

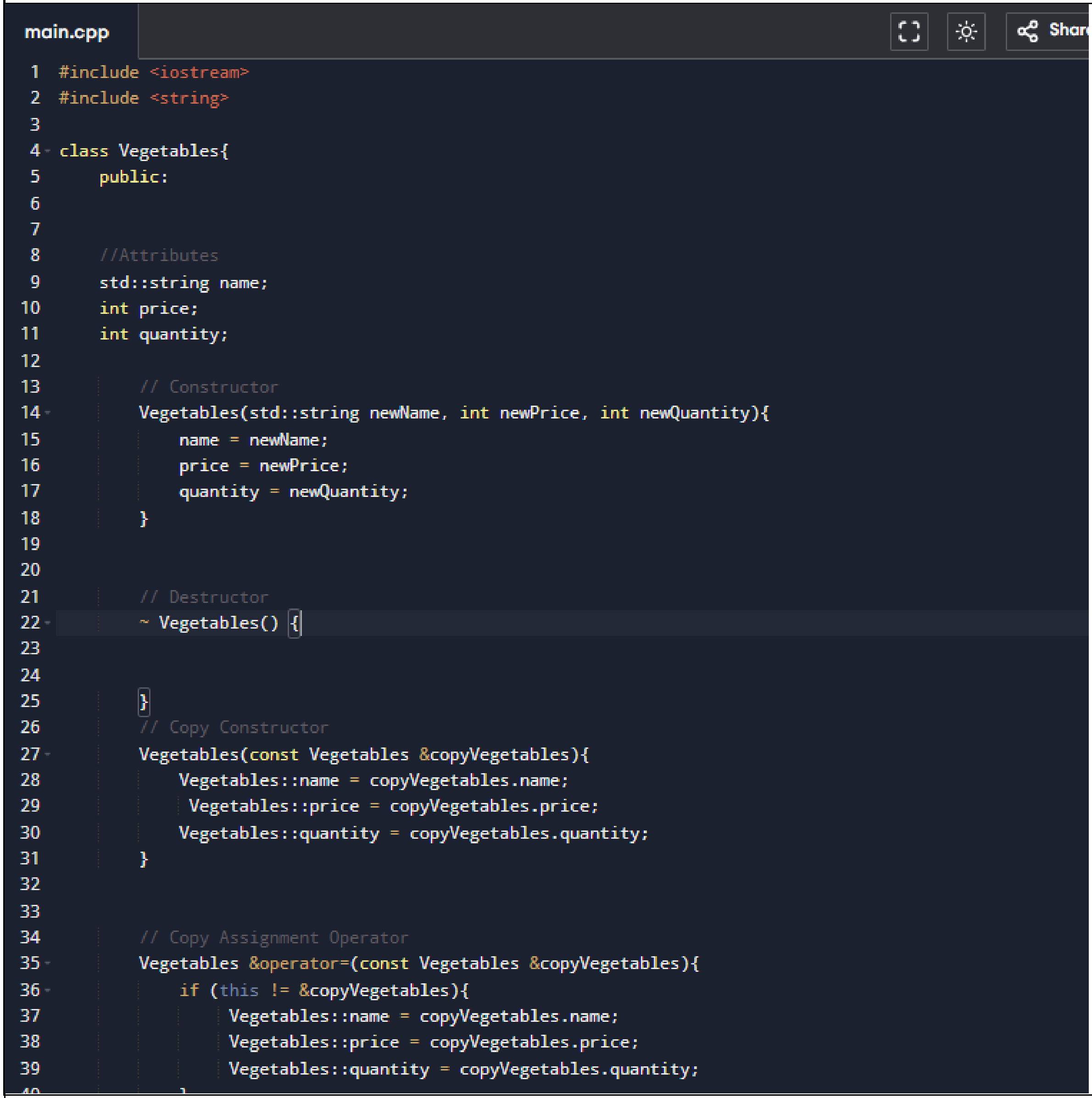
Problem 2



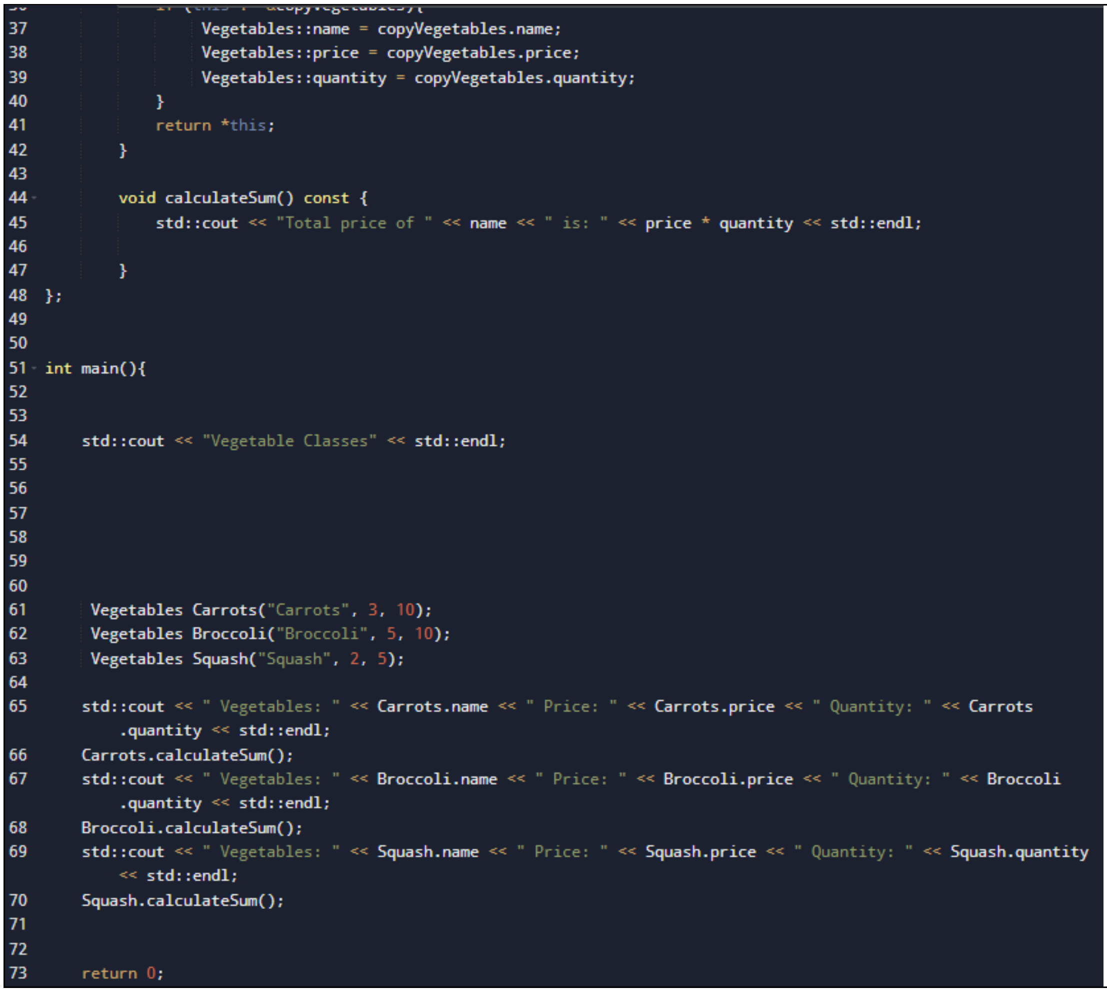
Output

```
Item: Coffee
Quantity: 10
Price per unit: Php2.99
Total cost: Php29.9
Item: Bread
Quantity: 10
Price per unit: Php3.5
Total cost: Php35
Item: Milk
Quantity: 10
Price per unit: Php4.5
Total cost: Php45


=== Code Execution Successful ===
```

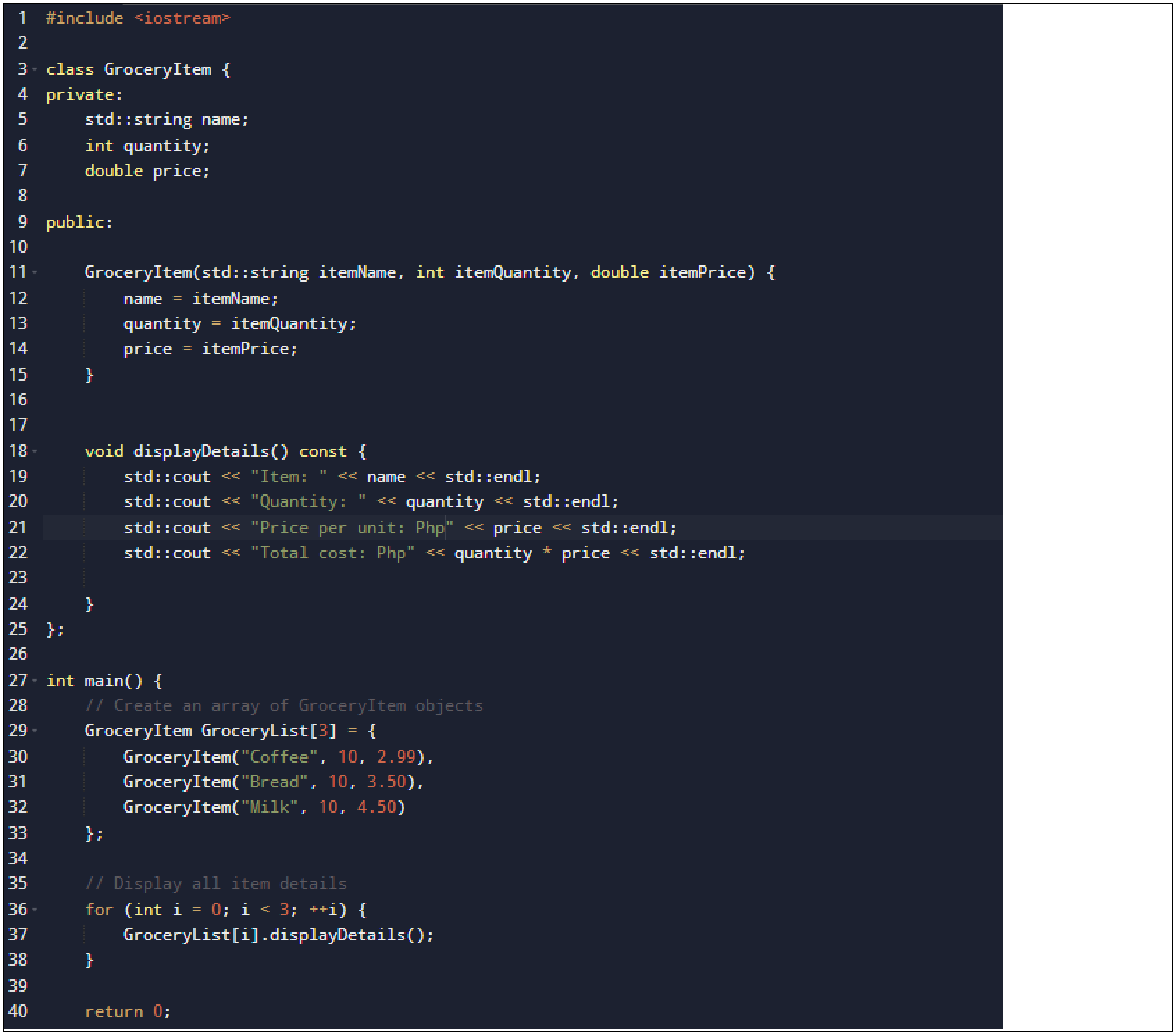
7. Supplementary Activity



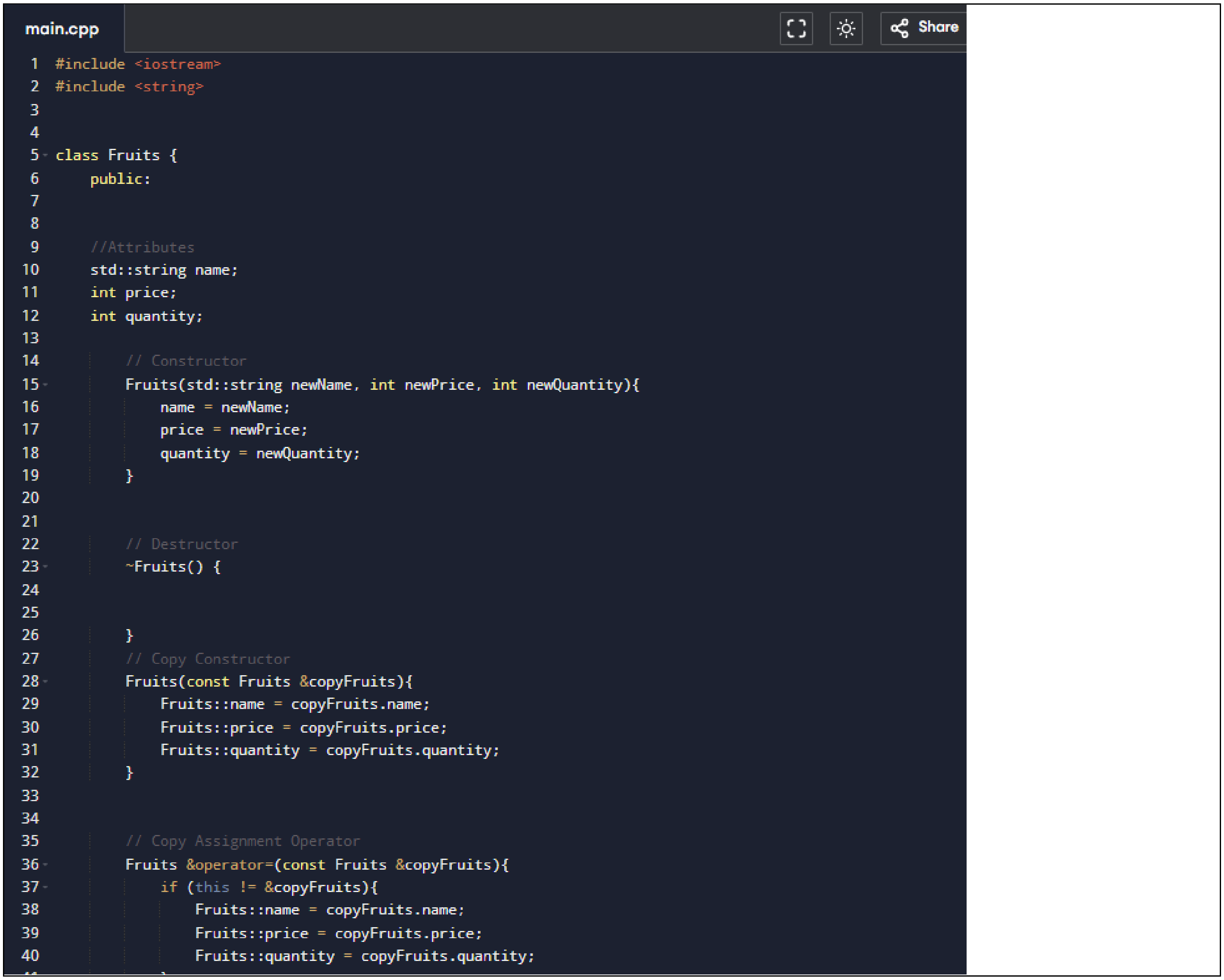
```cpp
#include <iostream>
#include <string>

class Vegetables{
    public:


    //Attributes
    std::string name;
    int price;
    int quantity;

        // Constructor
        Vegetables(std::string newName, int newPrice, int newQuantity){
            name = newName;
            price = newPrice;
            quantity = newQuantity;
        }


        // Destructor
        ~ Vegetables() {


        }
        // Copy Constructor
        Vegetables(const Vegetables &copyVegetables){
            Vegetables::name = copyVegetables.name;
             Vegetables::price = copyVegetables.price;
            Vegetables::quantity = copyVegetables.quantity;
        }


        // Copy Assignment Operator
        Vegetables &operator=(const Vegetables &copyVegetables){
            if (this != &copyVegetables){
                Vegetables::name = copyVegetables.name;
                Vegetables::price = copyVegetables.price;
                Vegetables::quantity = copyVegetables.quantity;
            }
```



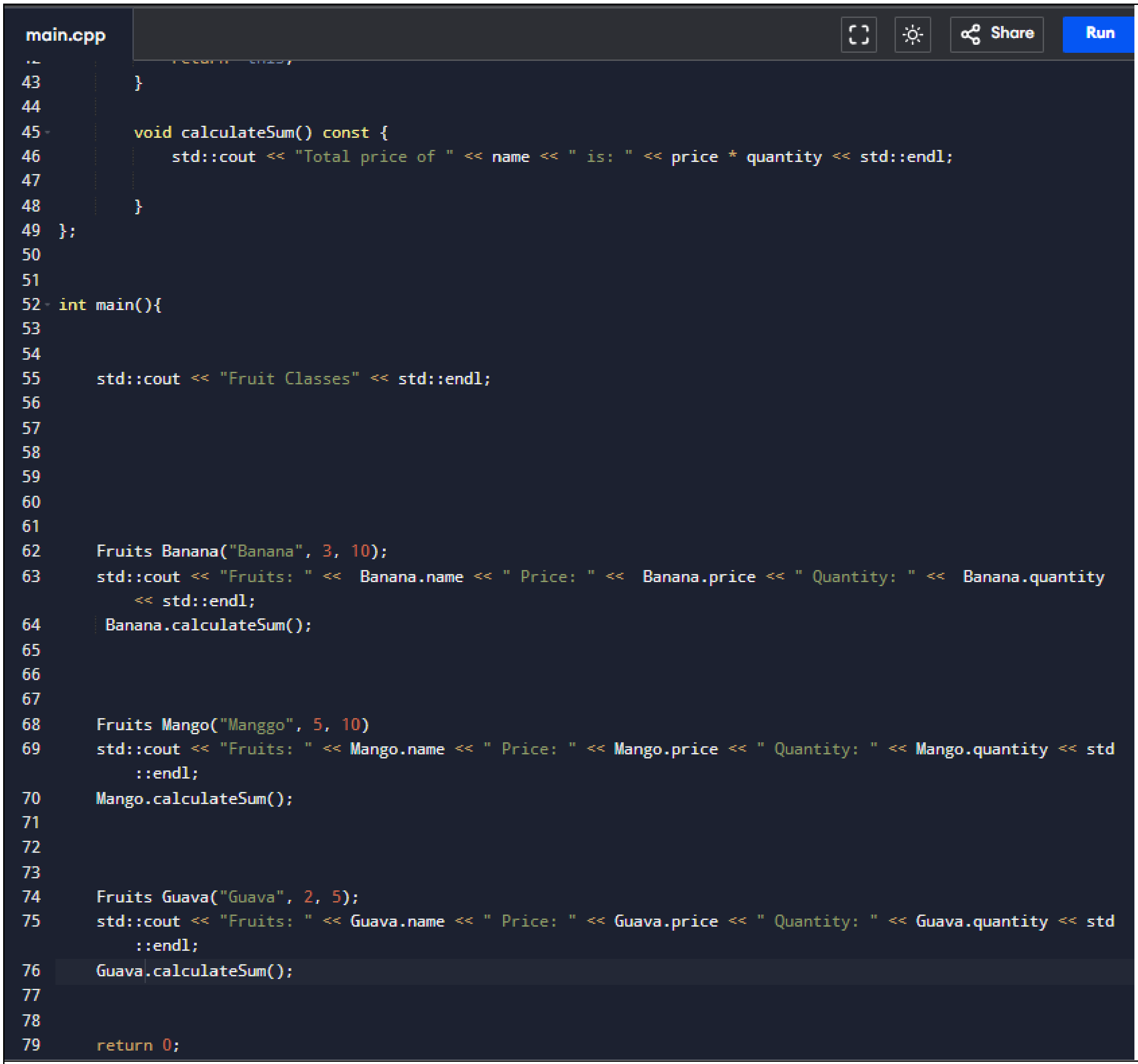
```cpp
            if (this != &copyVegetables){
                Vegetables::name = copyVegetables.name;
                Vegetables::price = copyVegetables.price;
                Vegetables::quantity = copyVegetables.quantity;
            }
            return *this;
        }

        void calculateSum() const {
            std::cout << "Total price of " << name << " is: " << price * quantity << std::endl;

        }
};


int main(){


    std::cout << "Vegetable Classes" << std::endl;






     Vegetables Carrots("Carrots", 3, 10);
     Vegetables Broccoli("Broccoli", 5, 10);
     Vegetables Squash("Squash", 2, 5);

    std::cout << " Vegetables: " << Carrots.name << " Price: " << Carrots.price << " Quantity: " << Carrots.quantity << std::endl;
    Carrots.calculateSum();
    std::cout << " Vegetables: " << Broccoli.name << " Price: " << Broccoli.price << " Quantity: " << Broccoli.quantity << std::endl;
    Broccoli.calculateSum();
    std::cout << " Vegetables: " << Squash.name << " Price: " << Squash.price << " Quantity: " << Squash.quantity << std::endl;
    Squash.calculateSum();


    return 0;
```



```cpp
#include <iostream>

class GroceryItem {
private:
    std::string name;
    int quantity;
    double price;

public:

    GroceryItem(std::string itemName, int itemQuantity, double itemPrice) {
        name = itemName;
        quantity = itemQuantity;
        price = itemPrice;
    }


    void displayDetails() const {
        std::cout << "Item: " << name << std::endl;
        std::cout << "Quantity: " << quantity << std::endl;
        std::cout << "Price per unit: Php" << price << std::endl;
        std::cout << "Total cost: Php" << quantity * price << std::endl;

    }
};

int main() {
    // Create an array of GroceryItem objects
    GroceryItem GroceryList[3] = {
        GroceryItem("Coffee", 10, 2.99),
        GroceryItem("Bread", 10, 3.50),
        GroceryItem("Milk", 10, 4.50)
    };

    // Display all item details
    for (int i = 0; i < 3; ++i) {
        GroceryList[i].displayDetails();
    }

    return 0;
```



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


class Fruits {
    public:


    //Attributes
    std::string name;
    int price;
    int quantity;

        // Constructor
        Fruits(std::string newName, int newPrice, int newQuantity){
            name = newName;
            price = newPrice;
            quantity = newQuantity;
        }


        // Destructor
        ~Fruits() {


        }
        // Copy Constructor
        Fruits(const Fruits &copyFruits){
            Fruits::name = copyFruits.name;
            Fruits::price = copyFruits.price;
            Fruits::quantity = copyFruits.quantity;
        }


        // Copy Assignment Operator
        Fruits &operator=(const Fruits &copyFruits){
            if (this != &copyFruits){
                Fruits::name = copyFruits.name;
                Fruits::price = copyFruits.price;
                Fruits::quantity = copyFruits.quantity;
```



```
            return *this;
        }

        void calculateSum() const {
            std::cout << "Total price of " << name << " is: " << price * quantity << std::endl;

        }
};


int main(){


    std::cout << "Fruit Classes" << std::endl;






    Fruits Banana("Banana", 3, 10);
    std::cout << "Fruits: " <<  Banana.name << " Price: " <<  Banana.price << " Quantity: " <<  Banana.quantity
        << std::endl;
     Banana.calculateSum();



    Fruits Mango("Manggo", 5, 10)
    std::cout << "Fruits: " << Mango.name << " Price: " << Mango.price << " Quantity: " << Mango.quantity << std
        ::endl;
    Mango.calculateSum();



    Fruits Guava("Guava", 2, 5);
    std::cout << "Fruits: " << Guava.name << " Price: " << Guava.price << " Quantity: " << Guava.quantity << std
        ::endl;
    Guava.calculateSum();


    return 0;
```

8. Conclusion:

To conclude, we were able to do some of the tasks. It was confusing and hard to it so what we did was watch some videos on how to do it and ask some help to my friends. The task were making a class together with the object and methods and how will we apply it in in C++. Overall, it was hard to do it and I hope we get to learn more of this to help myself in making a code or to ourselves.

9. Assessment Rubric