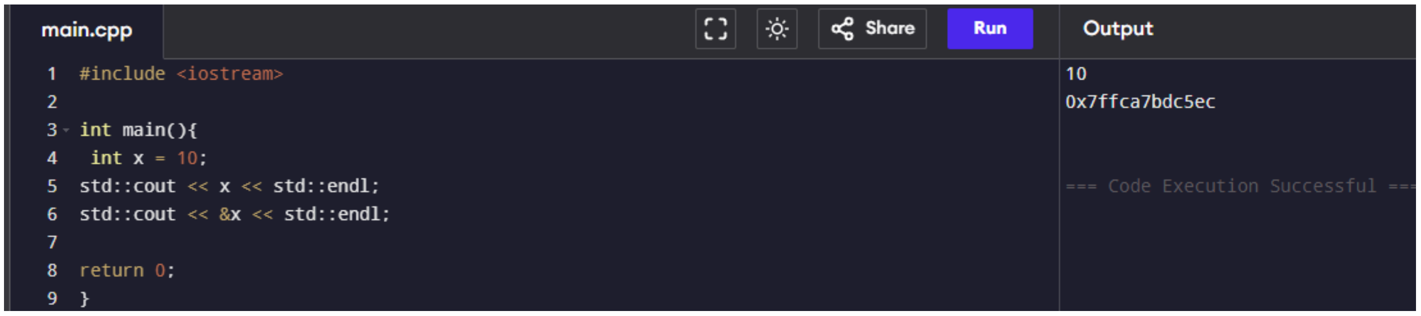
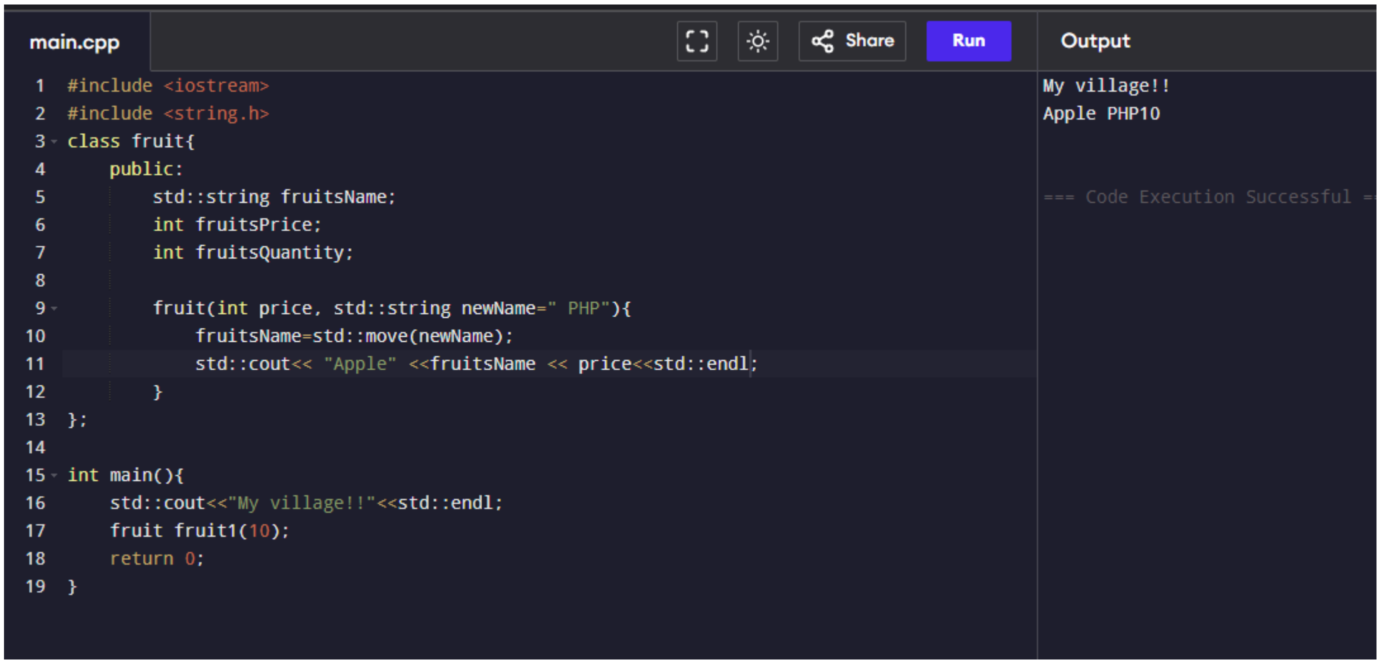
Activity No. 2	
Arrays, Pointers and Dynamic Memory Allocation	
Course Code: CPE010	**Program:** Computer Engineering
Course Title: Data Structures and Algorithms	**Date Performed: 7/31/25**
Section:CpE21S4	**Date Submitted: 7/31/25**
Name(s): Francis Nikko I. Andoy	**Instructor: Jimboy Quejado**

6. Output



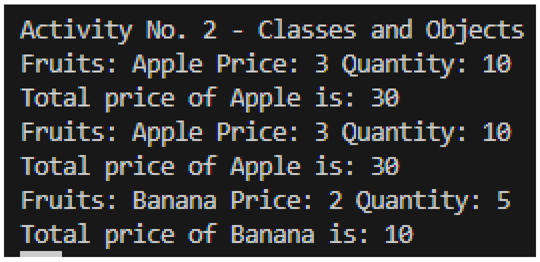
```
#include <iostream>

int main(){
 int x = 10;
std::cout << x << std::endl;
std::cout << &x << std::endl;

return 0;
}
```

```
10
0x7ffca7bdc5ec


=== Code Execution Successful ==
```

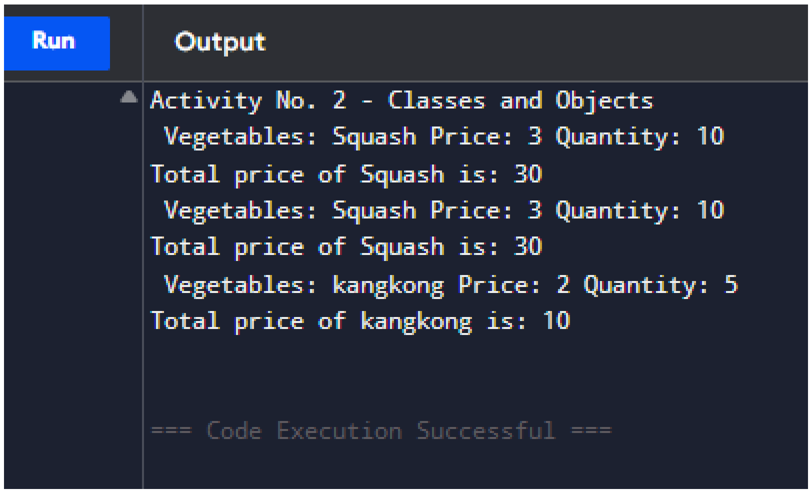


```
#include <iostream>
#include <string.h>
class fruit{
    public:
        std::string fruitsName;
        int fruitsPrice;
        int fruitsQuantity;

        fruit(int price, std::string newName=" PHP"){
            fruitsName=std::move(newName);
            std::cout<< "Apple" <<fruitsName << price<<std::endl;
        }
};

int main(){
    std::cout<<"My village!!"<<std::endl;
    fruit fruit1(10);
    return 0;
}
```

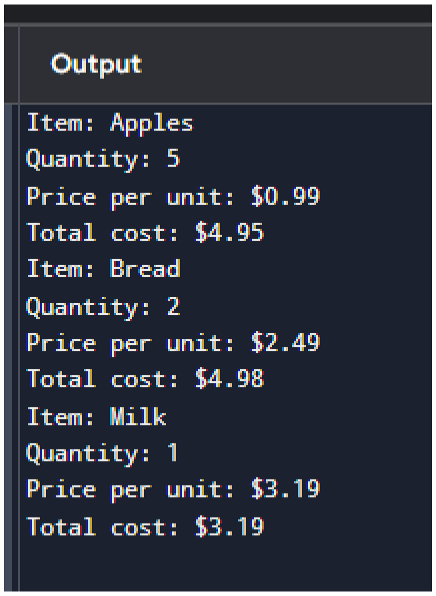
```
My village!!
Apple PHP10


=== Code Execution Successful =
```



```
Activity No. 2 - Classes and Objects
Fruits: Apple Price: 3 Quantity: 10
Total price of Apple is: 30
Fruits: Apple Price: 3 Quantity: 10
Total price of Apple is: 30
Fruits: Banana Price: 2 Quantity: 5
Total price of Banana is: 10
```



```
Activity No. 2 - Classes and Objects
 Vegetables: Squash Price: 3 Quantity: 10
Total price of Squash is: 30
 Vegetables: Squash Price: 3 Quantity: 10
Total price of Squash is: 30
 Vegetables: kangkong Price: 2 Quantity: 5
Total price of kangkong is: 10



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


```
Output
Item: Apples
Quantity: 5
Price per unit: $0.99
Total cost: $4.95
Item: Bread
Quantity: 2
Price per unit: $2.49
Total cost: $4.98
Item: Milk
Quantity: 1
Price per unit: $3.19
Total cost: $3.19
```

7. Supplementary Activity

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


class Fruit {
    public:

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

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

        // Destructor
        ~Fruit() {

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

        // Copy Assignment Operator
        Fruit &operator=(const Fruit &copyFruit){
            if (this != &copyFruit){
                Fruit::name = copyFruit.name;
                Fruit::price = copyFruit.price;
                Fruit::quantity = copyFruit.quantity;
            }
            return *this;
        }

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

        }
};

int main(){

    std::cout << "Activity No. 2 - Classes and Objects" << std::endl;




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


    Fruit orange(apple);
    std::cout << "Fruits: " << orange.name << " Price: " << orange.price << "
Quantity: " << orange.quantity << std::endl;
    orange.calculateSum();


    Fruit banana("Banana", 2, 5);
    std::cout << "Fruits: " << banana.name << " Price: " << banana.price << "
Quantity: " << banana.quantity << std::endl;
    banana.calculateSum();
    return 0;
}

#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;
            }
            return *this;
        }

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

        }
};


int main(){


    std::cout << "Activity No. 2 - Classes and Objects" << std::endl;







     Vegetables squash("Squash", 3, 10);
     Vegetables carrots(squash);
     Vegetables kangkong("kangkong", 2, 5);

    std::cout << " Vegetables: " << squash.name << " Price: " << squash.price << "
Quantity: " << squash.quantity << std::endl;
    squash.calculateSum();
    std::cout << " Vegetables: " << carrots.name << " Price: " << carrots.price << "
Quantity: " << carrots.quantity << std::endl;
    carrots.calculateSum();
    std::cout << " Vegetables: " << kangkong.name << " Price: " << kangkong.price <<
" Quantity: " << kangkong.quantity << std::endl;
    kangkong.calculateSum();


    return 0;
}




#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: $" << price << std::endl;
        std::cout << "Total cost: $" << quantity * price << std::endl;

    }
};

int main() {
    // Create an array of GroceryItem objects
    GroceryItem GroceryList[3] = {
        GroceryItem("Apples", 5, 0.99),
        GroceryItem("Bread", 2, 2.49),
        GroceryItem("Milk", 1, 3.19)
    };

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

    return 0;
}
```

8. Conclusion:

In conclusion, I learned how to use classes and objects, and also understand the basics of OOP, although I'm still confused about how member functions work and how they can help us. I'm having a hard time learning the concept behind this lesson. The access specifiers, which the private and public that we use on classes, these

two are also confusing and difficult to understand, especially the private since it requires more deeper understanding of how OOP works. In here, I don’t know if we can use the pointer, since I myself don’t even understand how the pointer and reference really work and how these two can help us on our code.
9. Assessment Rubric