UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 4

Arrays

Submitted by:
MAMANAO, KURT MARWIN C,

Instructor:
Engr. Maria Rizette H. Sayo

AUGUST, 9, 2025

I. Objectives

Introduction

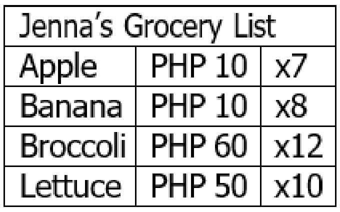
Array, in general, refers to an orderly arrangement of data elements. Array is a type of data structure that stores data elements in adjacent locations. Array is considered as linear data structure that stores elements of same data types. Hence, it is also called as a linear homogenous data structure.

This laboratory activity aims to implement the principles and techniques in:

- Writing algorithms using Array data structure
- Solve programming problems using dynamic memory allocation, arrays and pointers

II. Methods

Jenna's Grocery



Jenna's Grocery List		
Apple	PHP 10	x7
Banana	PHP 10	x8
Broccoli	PHP 60	x12
Lettuce	PHP 50	x10

Jenna wants to buy the following fruits and vegetables for her daily consumption. However, she needs to distinguish between fruit and vegetable, as well as calculate the sum of prices that she has to pay in total.

Problem 1: Create a class for the fruit and the vegetable classes. Each class must have a constructor, deconstructor, copy constructor and copy assignment operator. They must also have all relevant attributes (such as name, in the problem price and quantity) and functions (such as calculate sum) as presented description above.

Problem 2: Create an array GroceryList in the driver code that will contain all items in Jenna's Grocery List. You must then access each saved instance and display all details about the items.

Problem 3: Create a function TotalSum that will calculate the sum of all objects listed in Jenna's Grocery List.

Problem 4: Delete the Lettuce from Jenna's GroceryList list and de-allocate the memory assigned.

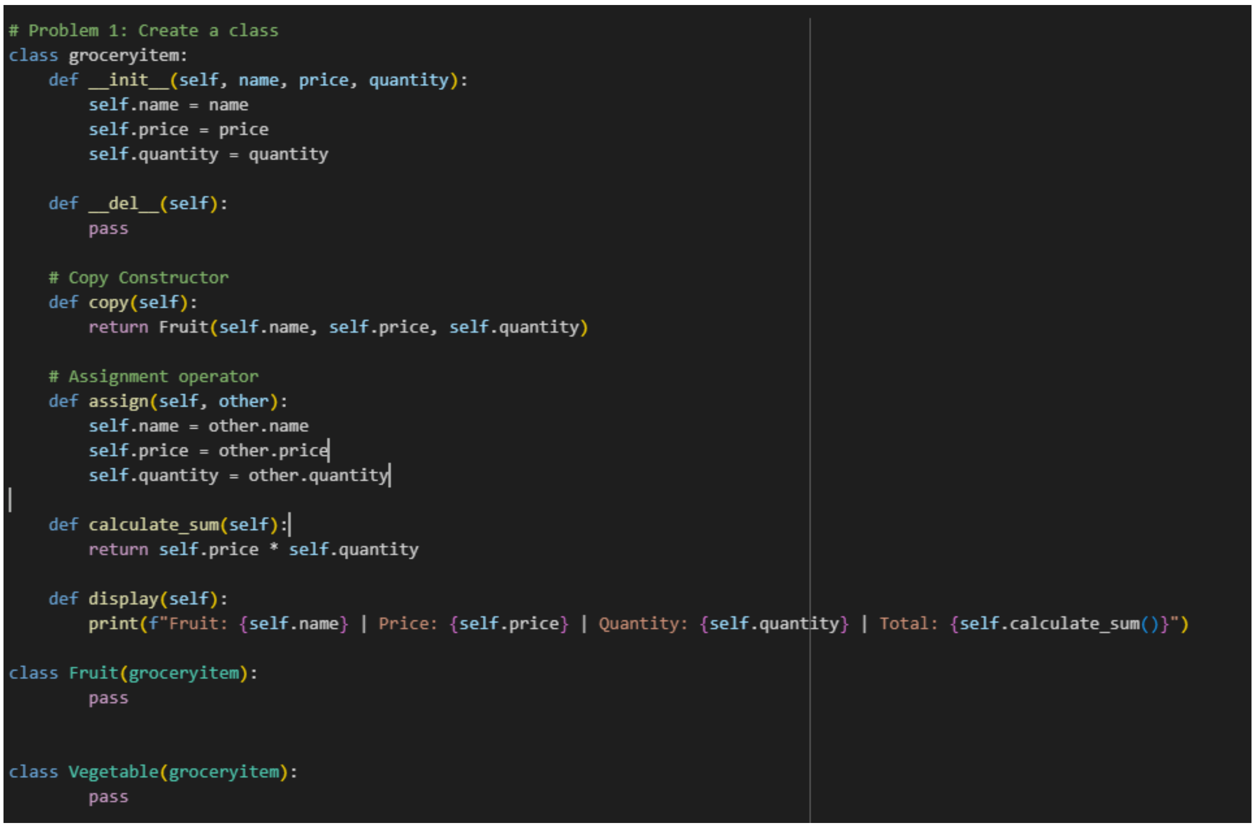
III. Results

Algorithm for Jenna's Grocery Program

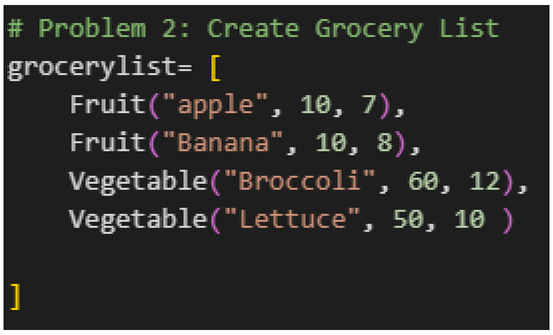
1. **Start**
2. Define a **class GroceryItem** with:
    - Attributes: name, price, quantity.
    - Methods:
        - __init__() → initializes name, price, and quantity.
        - __del__() → destructor (runs when object is deleted).
        - copy() → creates a new object with the same values.
        - assign(other) → copies values from another object.
        - calculate_sum() → returns price × quantity.
        - display() → prints item details.
3. Create two subclasses:
    - **Fruit** (inherits from GroceryItem).
    - **Vegetable** (inherits from GroceryItem).
4. Create a list named **grocerylist** and add the following items:
    - Fruit("Apple", 10, 7)
    - Fruit("Banana", 10, 8)
    - Vegetable("Broccoli", 60, 12)
    - Vegetable("Lettuce", 50, 10)
5. Print the heading "=== JENNA'S GROCERY LIST ===".
6. For each item in grocerylist, call the **display()** method.
7. Define a function **TotalSum(grocerylist)** that:
    - Calculates the sum of all items using calculate_sum().
    - Prints the total.
8. Call **TotalSum(grocerylist)** to display the total amount.
9. Print "Removing Lettuce from the list".
10. Update grocerylist by removing the item whose name is "Lettuce".
11. Print "UPDATED GROCERY LIST".
12. For each item in the updated grocerylist, call the **display()** method.
13. Call **TotalSum(grocerylist)** again to show the new total.
14. **End**

Source code

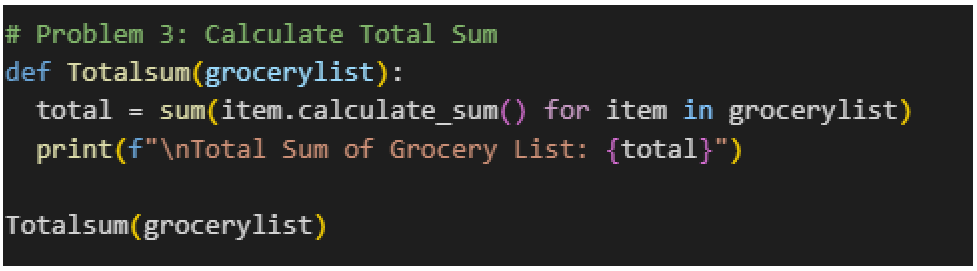
.



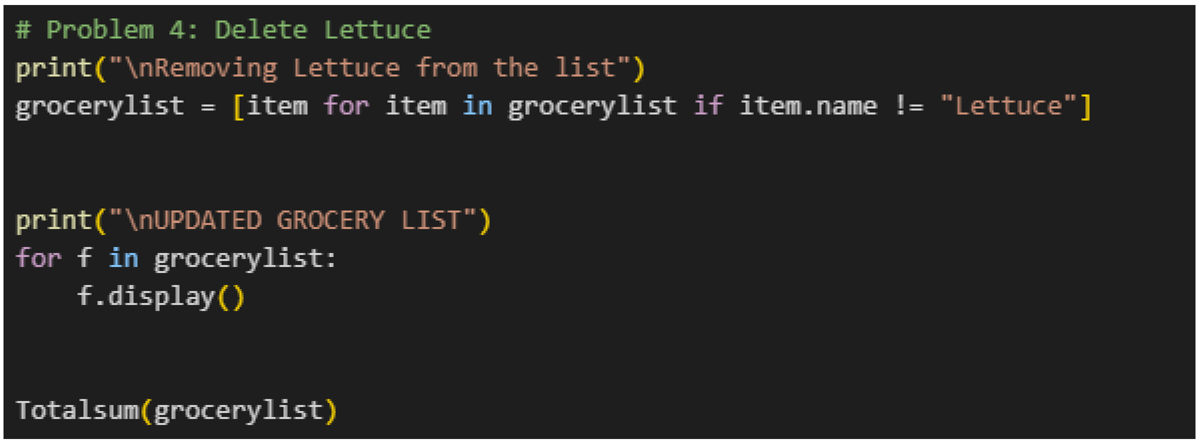
```python
# Problem 1: Create a class
class groceryitem:
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
        self.quantity = quantity

    def __del__(self):
        pass

    # Copy Constructor
    def copy(self):
        return Fruit(self.name, self.price, self.quantity)

    # Assignment operator
    def assign(self, other):
        self.name = other.name
        self.price = other.price
        self.quantity = other.quantity

    def calculate_sum(self):
        return self.price * self.quantity

    def display(self):
        print(f"Fruit: {self.name} | Price: {self.price} | Quantity: {self.quantity} | Total: {self.calculate_sum()}")

class Fruit(groceryitem):
        pass


class Vegetable(groceryitem):
        pass
```



```python
# Problem 2: Create Grocery List
grocerylist= [
    Fruit("apple", 10, 7),
    Fruit("Banana", 10, 8),
    Vegetable("Broccoli", 60, 12),
    Vegetable("Lettuce", 50, 10 )

]
```

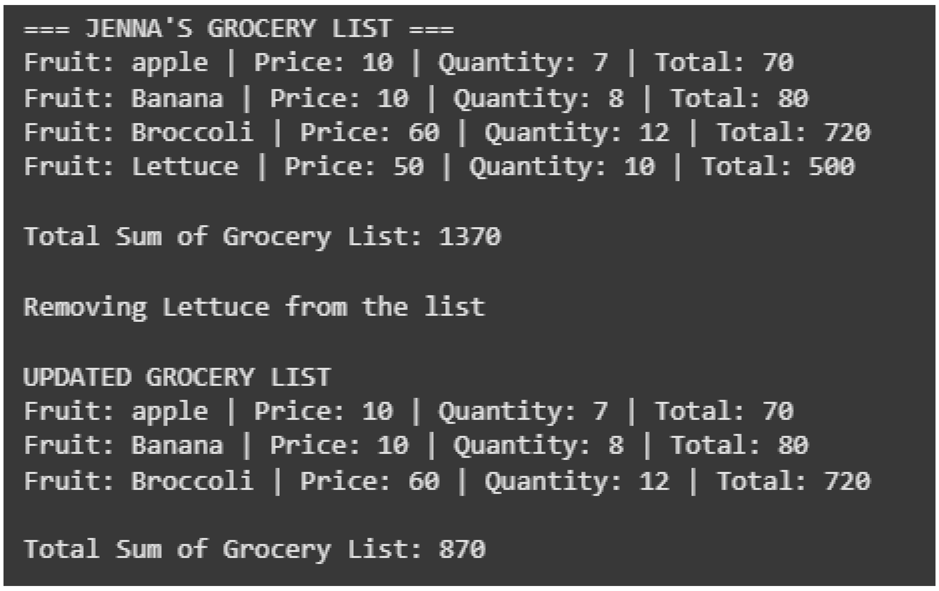


```python
# Problem 3: Calculate Total Sum
def Totalsum(grocerylist):
  total = sum(item.calculate_sum() for item in grocerylist)
  print(f"\nTotal Sum of Grocery List: {total}")

Totalsum(grocerylist)
```



```python
# Problem 4: Delete Lettuce
print("\nRemoving Lettuce from the list")
grocerylist = [item for item in grocerylist if item.name != "Lettuce"]


print("\nUPDATED GROCERY LIST")
for f in grocerylist:
    f.display()


Totalsum(grocerylist)
```

OUTPUI



```
=== JENNA'S GROCERY LIST ===
Fruit: apple | Price: 10 | Quantity: 7 | Total: 70
Fruit: Banana | Price: 10 | Quantity: 8 | Total: 80
Fruit: Broccoli | Price: 60 | Quantity: 12 | Total: 720
Fruit: Lettuce | Price: 50 | Quantity: 10 | Total: 500

Total Sum of Grocery List: 1370

Removing Lettuce from the list

UPDATED GROCERY LIST
Fruit: apple | Price: 10 | Quantity: 7 | Total: 70
Fruit: Banana | Price: 10 | Quantity: 8 | Total: 80
Fruit: Broccoli | Price: 60 | Quantity: 12 | Total: 720

Total Sum of Grocery List: 870
```

IV. Conclusion

This lab activity helped me understand arrays and object-oriented programming better. By making classes for fruits and vegetables, I learned how to use constructors, destructors, copy, and assignment methods in a simple program. I also practiced calculating totals and removing items from a list, which showed me how these concepts work in real-life situations like managing groceries. Overall, this activity improved my knowledge of arrays and algorithms.

References

[1]GeeksforGeeks, "Arrays in Data Structures," [Online]. Available: https://www.geeksforgeeks.org/array-data-structure/.