UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 4

Arrays

Submitted by:
Villanueva, Bryan O.

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, price and quantity) and functions (such as calculate sum) as presented in the problem 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

Source Code:

```
import copy

class Fruit:
    # Constructor
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
        self.quantity = quantity
        print(f"[Constructor] Created Fruit: {self.name}")

    # Copy Constructor
    def copy(self):
        print(f"[Copy Constructor] Copying Fruit: {self.name}")
        return copy.deepcopy(self)

    # Copy Assignment Operator
    def assign(self, other):
        print(f"[Copy Assignment] Assigning {other.name} to {self.name}")
        self.name = other.name
        self.price = other.price
        self.quantity = other.quantity

    # Calculate total
    def calculate_sum(self):
        return self.price * self.quantity

    # Destructor
    def __del__(self):
        print(f"[Destructor] Deleted Fruit: {self.name}")

class Vegetable:
    # Constructor
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
```

```
        self.quantity = quantity
        print(f"[Constructor] Created Vegetable: {self.name}")

    # Copy Constructor
    def copy(self):
        print(f"[Copy Constructor] Copying Vegetable: {self.name}")
        return copy.deepcopy(self)

    # Copy Assignment
    def assign(self, other):
        print(f"[Copy Assignment] Assigning {other.name} to {self.name}")
        self.name = other.name
        self.price = other.price
        self.quantity = other.quantity

    # Calculate total
    def calculate_sum(self):
        return self.price * self.quantity

    # Destructor
    def __del__(self):
        print(f"[Destructor] Deleted Vegetable: {self.name}")

# Problem 3 – Total sum function
def TotalSum(grocery_list):
    total = 0
    for item in grocery_list:
        if item is not None:
            total += item.calculate_sum()
    return total

# DRIVER CODE
def main():

    # Problem 1 – Create objects
    apple = Fruit("Apple", 10, 3)
    banana = Fruit("Banana", 7, 5)
    lettuce = Vegetable("Lettuce", 25, 1)
```

```
    carrot = Vegetable("Carrot", 8, 4)

    # Problem 2 – Array (Python list)
    GroceryList = [apple, banana, lettuce, carrot]

    print("\n=== Jenna’s Grocery List ===")
    for item in GroceryList:
        print(f"Name: {item.name}, Price: {item.price}, Qty: {item.quantity}, Total: {item.calculate_sum()}")

    # Problem 3 – Total sum
    print("\nTotal Cost:", TotalSum(GroceryList))

    # Problem 4 – Delete Lettuce
    print("\n=== Deleting Lettuce ===")
    for i in range(len(GroceryList)):
        if GroceryList[i].name == "Lettuce":
            del GroceryList[i]   # deletes object and shifts array
            break

    print("\n=== Updated Grocery List ===")
    for item in GroceryList:
        print(f"Name: {item.name}, Total: {item.calculate_sum()}")

    print("\nNew Total Cost:", TotalSum(GroceryList))

main()
```

```
[Constructor] Created Fruit: Apple
[Constructor] Created Fruit: Banana
[Constructor] Created Vegetable: Lettuce
[Constructor] Created Vegetable: Carrot

=== Jenna’s Grocery List ===
Name: Apple, Price: 10, Qty: 3, Total: 30
Name: Banana, Price: 7, Qty: 5, Total: 35
Name: Lettuce, Price: 25, Qty: 1, Total: 25
Name: Carrot, Price: 8, Qty: 4, Total: 32

Total Cost: 122

=== Deleting Lettuce ===

=== Updated Grocery List ===
Name: Apple, Total: 30
Name: Banana, Total: 35
Name: Carrot, Total: 32

New Total Cost: 97
[Destructor] Deleted Vegetable: Lettuce
[Destructor] Deleted Fruit: Banana
[Destructor] Deleted Fruit: Apple
[Destructor] Deleted Vegetable: Carrot
```

Figure 1 Outputs

IV. Conclusion

In this laboratory activity, I was able to apply the concepts of arrays, classes, and dynamic memory handling in Python. By creating the Fruit and Vegetable classes, I learned how to use constructors, copy constructors, copy assignment, and destructors in an object-oriented way. Using an array to store all the grocery items helped me practice accessing objects through indexing and displaying their details. I also practiced computing the total price of all items using a separate function, which improved my understanding of how functions work with arrays. Lastly, deleting the Lettuce item from the list showed how memory management and element removal work in a data structure.