UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT

Data Structure and Algorithm

Laboratory Activity No. 4

Arrays

Submitted by:
Ampong, J-kevin L.

Instructor:
Engr. Maria Rizette H. Sayo

August, 9, 2025

I. Objectives

Introduction

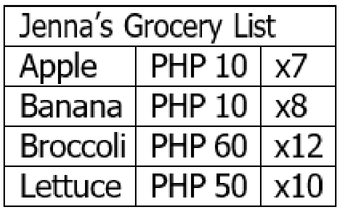
Array, in general, refers to an orderly arrangement of data elements. Array is a type of data structure that stores data elements in adjacent locations. Array is considered as linear data structure that stores elements of same data types. Hence, it is also called as a linear homogenous data structure.

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

<u>Jenna's Grocery</u>



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.

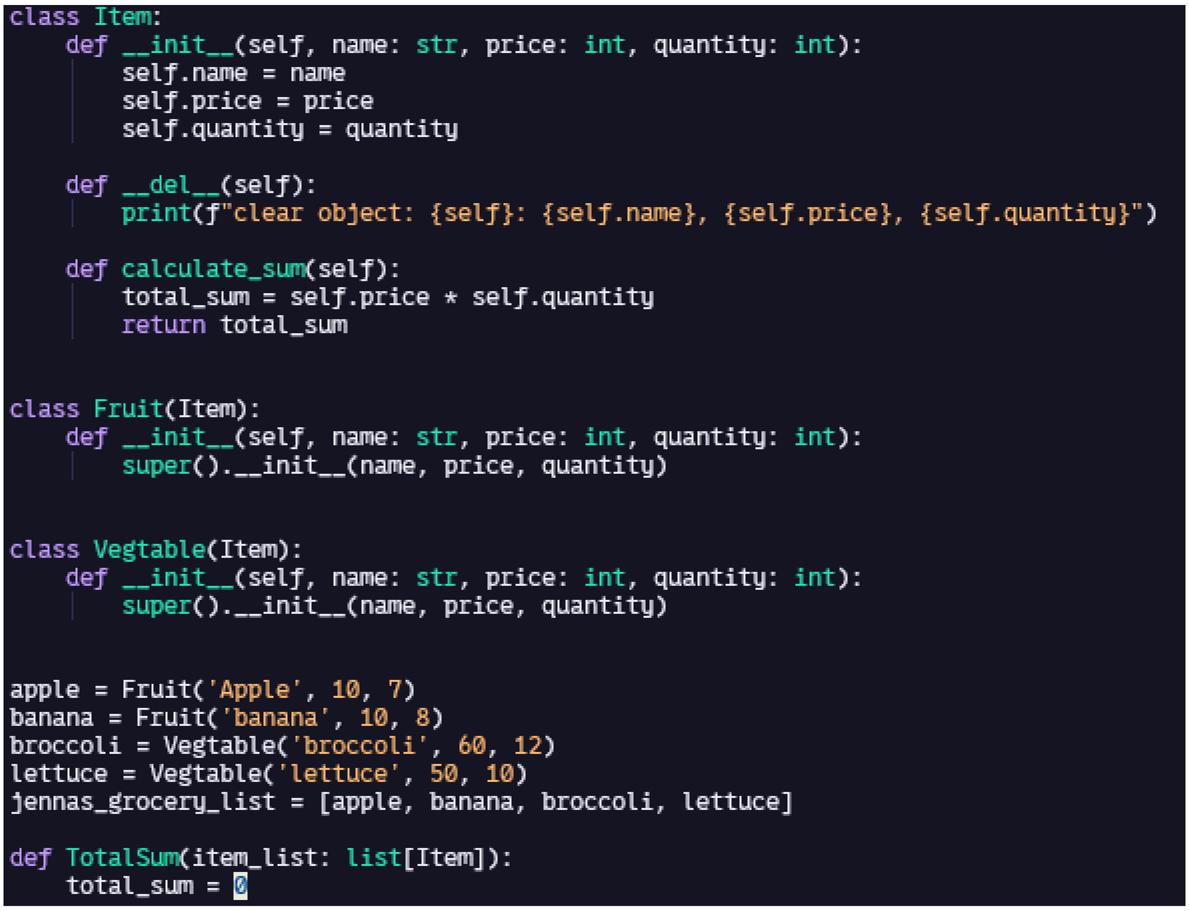
Problem 1: Create a class for the fruit and the vegetable classes. Each class must have a constructor, deconstructor, copy constructor and copy assignment operator. They must also have all relevant attributes (such as name, price and quantity) and functions (such as calculate sum) as presented in the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

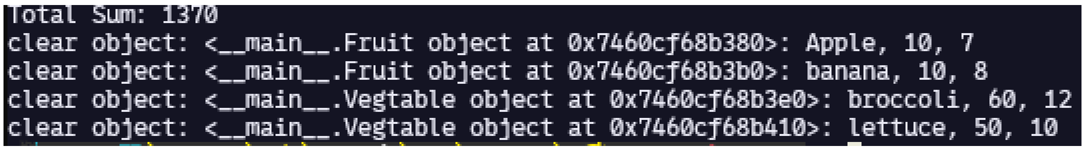
Problem 4: Delete the Lettuce from Jenna's GroceryList list and de-allocate the memory assigned.

III. Results



```
class Item:
    def __init__(self, name: str, price: int, quantity: int):
        self.name = name
        self.price = price
        self.quantity = quantity

    def __del__(self):
        print(f"clear object: {self}: {self.name}, {self.price}, {self.quantity}")

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


class Fruit(Item):
    def __init__(self, name: str, price: int, quantity: int):
        super().__init__(name, price, quantity)


class Vegtable(Item):
    def __init__(self, name: str, price: int, quantity: int):
        super().__init__(name, price, quantity)


apple = Fruit('Apple', 10, 7)
banana = Fruit('banana', 10, 8)
broccoli = Vegtable('broccoli', 60, 12)
lettuce = Vegtable('lettuce', 50, 10)
jennas_grocery_list = [apple, banana, broccoli, lettuce]

def TotalSum(item_list: list[Item]):
    total_sum = 0
```

Output:



```
Total Sum: 1370
clear object: <__main__.Fruit object at 0x7460cf68b380>: Apple, 10, 7
clear object: <__main__.Fruit object at 0x7460cf68b3b0>: banana, 10, 8
clear object: <__main__.Vegtable object at 0x7460cf68b3e0>: broccoli, 60, 12
clear object: <__main__.Vegtable object at 0x7460cf68b410>: lettuce, 50, 10
```

https://github.com/AmpongJkevin2/CPE-201L-DSA-2-A/blob/main/Act4/lab_4.ipynb

IV. Conclusion

This laboratory activity teaches me the fundamentals of constructor and deconstructor in a class, this gives some knowledge on how objects work in detail like cleaning memory after using object as well manually freeing using del keyword.

What I learned is that objects can be manually clean, but this is done automatically by the garbage collector since it's part of simplicity of python core system design as a result no need to worry in memory leaks in another programming language such as C, C++ etc.

References

[1] Co Arthur O.. “University of Caloocan City Computer Engineering Department Honor Code,” UCC-CpE Departmental Policies, 2020.