



UNIVERSITY OF CALOOCAN CITY  
COMPUTER ENGINEERING DEPARTMENT



Data Structure and Algorithm

Laboratory Activity No. 4

---

# Arrays

---

*Submitted by:*  
Disomnong, Jalilah M.

*Instructor:*  
Engr. Maria Rizette H. Sayo

08-09-2025

# 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

```
class Fruit:
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
        self.quantity = quantity

    def get_name(self):
        return self.name
    def get_price(self):
        return self.price
    def get_quantity(self):
        return self.quantity

class Vegetables:
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
        self.quantity = quantity

    def get_name(self):
        return self.name
    def get_price(self):
        return self.price
    def get_quantity(self):
        return self.quantity

def sum_groceries(groceries):
    total = 0
    for groceries in groceries:
        total += groceries.get_price() * groceries.get_quantity()
    return total
```

Figure 1 Problem 1 link: <https://colab.research.google.com/drive/1nmd1rvmLy2r4q4z1r6Vu2O-13Fiuq048#scrollTo=YCgQfHG3xjnp>

In figure 1, the provided source code defines a class called Fruit and Vegetables and function named sum\_groceries. The Fruit class and Vegetables class is designed to represent an individual grocery item, characterized by three attributes: name, price, and quantity. These attributes are initialized through the constructor method ( \_\_init\_\_ ) when a new instance of the class is created. The class also includes three getter methods — get\_name(), get\_price(), and get\_quantity() — which allow access to the respective attributes of a grocery item.

The function sum\_groceries takes a list of Grocery objects as an argument. Inside the function, a variable total is initialized to zero, which will accumulate the total cost of all grocery items. The function then iterates over each grocery item in the list, multiplying the price of the item by its quantity, and adds this value to the total. After completing the loop, the function returns the computed total cost of all groceries combined.

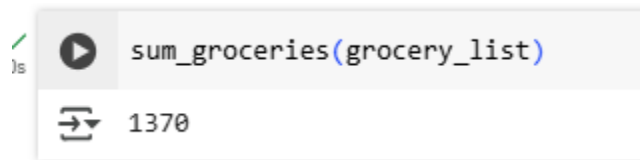
```
[5] apple = Fruit('Apple',10,7)
    banana = Fruit('Banana',10,8)
    broccoli = Vegetables('Broccoli',60,12)
    lettuce = Vegetables('Lettuce',50,10)

    grocery_list = [apple, banana, broccoli, lettuce]
```

Figure 2 Problem 2 link: <https://colab.research.google.com/drive/1nmd1rvmLy2r4q4z1r6Vu2O-13Fiuq048#scrollTo=YCgQfHG3xjnp>

In figure 2, the code creates two objects of the Fruit class and two objects of the Vegetable class each representing a grocery item with specific attributes: name, price, and quantity. For instance, apple is an object with the name "Apple," a price of 10, and a quantity of

7. Similarly, banana, broccoli, and lettuce are created with their respective prices and quantities. After creating these objects, it was group into a list called `grocery_list`. This list stores multiple objects, allowing them to be handled collectively in future operations.

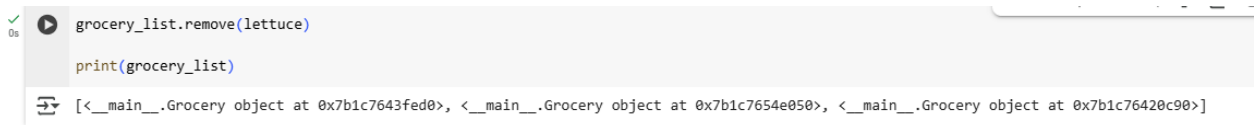


```
sum_groceries(grocery_list)
```

1370

Figure 3 Problem 3 link: <https://colab.research.google.com/drive/1nmd1rvmLy2r4q4z1r6Vu2O-13Fiuq048#scrollTo=YCgQfHG3xjnp>

In figure 3, The list passed to functions of `sum_groceries` to calculate the total value of all the grocery items combined.



```
grocery_list.remove(lettuce)
print(grocery_list)
```

[<\_\_main\_\_.Grocery object at 0x7b1c7643fed0>, <\_\_main\_\_.Grocery object at 0x7b1c7654e050>, <\_\_main\_\_.Grocery object at 0x7b1c76420c90>]

Figure 4 Problem 4 link: <https://colab.research.google.com/drive/1nmd1rvmLy2r4q4z1r6Vu2O-13Fiuq048#scrollTo=YCgQfHG3xjnp>

Figure 4, the `remove` method is used on the `grocery_list` to delete the `lettuce` object from the list. This means that after this operation, the list will no longer contain the `lettuce` grocery item.

## IV. Conclusion

In this laboratory activity, we explored the concept of arrays to organize and store multiple data items of the same type. The activity we performed was to use arrays to manage a list of groceries, specifically fruits and vegetables, by creating classes to represent each item with important details like name, price, and quantity. This activity make us to see how arrays can hold many objects together, making it easier to perform operations on the whole group.

## References

[1] Google. (2024). Google Colaboratory. Retrieved April 18, 2024, from <https://colab.research.google.com/>