



UNIVERSITY OF CALOOCAN CITY
COMPUTER ENGINEERING DEPARTMENT



Data Structure and Algorithm

Laboratory Activity No. 5

Implementation of Arrays

Submitted by:
Palmes, Lewis Clark L.

Instructor:
Engr. Maria Rizette H. Sayo

August 16, 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
- Writing a python program that can implement Array data structure

II. Methods

- Write a Python program to create an array of 10 integers and display the array items. Access individual elements through indexes and compute for the sum.
- Write a Python program to append a new item to the end of the array. Original array: numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
- Write a Python program to insert a new item before the second element in an existing array. Original array: numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
- Write a Python program to reverse the order of the items in the array. Original array: numbers = [5, 4, 3, 2, 1]

Write a Python program to get the length of the array. Original array: numbers = [5, 4, 3, 2, 1]

III. Results

```
numbers = [1,2,3,4,5,6,7,8,9,10]
print("Original Array:", numbers)
print("Sum of all elements:", sum(numbers))

while True:
    print("\nTo exit input 10")
    choice = input("Enter the index to call (0-9): ")

    if choice == '10':
        print("Exiting the program.")
        break

    if choice.isdigit() and int(choice) in range(0, 10):
        index = int(choice)
        print("Element at index", index, "is:", numbers[index])
    else:
        print("Invalid input, try again.")
```

Figure 1 and 2 : Screenshot of program

```
Original Array: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Sum of all elements: 55

To exit input 10
Enter the index to call (0-9): 0
Element at index 0 is: 1

To exit input 10
Enter the index to call (0-9): 1
Element at index 1 is: 2

To exit input 10
Enter the index to call (0-9): 2
Element at index 2 is: 3

To exit input 10
Enter the index to call (0-9): 3
Element at index 3 is: 4

To exit input 10
Enter the index to call (0-9): 4
Element at index 4 is: 5

To exit input 10
Enter the index to call (0-9): 5
Element at index 5 is: 6

To exit input 10
Enter the index to call (0-9): 6
Element at index 6 is: 7

To exit input 10
Enter the index to call (0-9): 7
Element at index 7 is: 8

To exit input 10
Enter the index to call (0-9): 8
Element at index 8 is: 9

To exit input 10
Enter the index to call (0-9): 9
Element at index 9 is: 10

To exit input 10
Enter the index to call (0-9): 10
Exiting the program.
```

The program stores numbers 1–10 in an array, prints the array and their total sum, then repeatedly asks the user for an index (0–9). If the index is valid, it shows the element at that position. If the user enters 10, the program exits.

```

numbers = [1,2,3,4,5,6,7,8,9,10]
print("Original Array", numbers)
numbers.append(int(input(f"Enter a number to add: ")))
print("Updated Array", numbers)

```

Original Array [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Enter a number to add: 24
Updated Array [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 24]

Figure 3 Screenshot of program

The program creates an array of numbers 1–10 and displays it. It then asks the user to input a number, adds that number to the end of the array using append(), and finally prints the updated array.

```

[ ] numbers = [1,2,3,4,5,6,7,8,9,10]
print("Original Array", numbers)

numbers.insert(1,(int(input(f"Enter the number you want to insert: "))))
print("Updated Array", numbers)

```

Original Array [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
Enter the number you want to insert: 24
Updated Array [1, 24, 2, 3, 4, 5, 6, 7, 8, 9, 10]

Figure 4 Screenshot of program

The program creates an array of numbers 1–10 and displays it. It then asks the user to input a number, adds that number before the second elemets of the array using insert(), and finally prints the updated array.

```

[ ] numbers = [5, 4, 3, 2, 1]
print("Original Array", numbers)

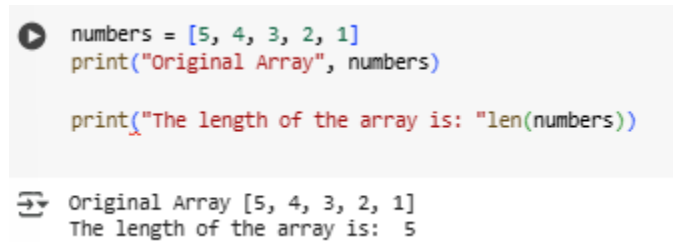
numbers.reverse()
print("Reversed Array", numbers)

```

Original Array [5, 4, 3, 2, 1]
Reversed Array [1, 2, 3, 4, 5]

Figure 5 Screenshot of program

The program creates an array of numbers 1–10 and displays it. It then reverses the numbers inside the array using reverse() function then prints it.



```
numbers = [5, 4, 3, 2, 1]
print("Original Array", numbers)

print("The length of the array is: " + str(len(numbers)))
```

Original Array [5, 4, 3, 2, 1]
The length of the array is: 5

Figure 3 Screenshot of program

The program creates an array of numbers 1–10 and displays it. It then prints the length of the array using len() function.

IV. Conclusion

This activity helped us understand how arrays work and how they can be used to store and manage data. By practicing different operations such as accessing elements, finding the sum, adding new items, inserting, reversing, and checking the length, we were able to see how arrays make handling data easier and more organized. Overall, the activity gave us a clearer picture of the usefulness of arrays in programming.

References

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

- [2] *Google Colab*. (n.d.). https://colab.research.google.com/drive/1R6thIPLfzd_Cj-OjEYSEwAn6Y1ohzeCA#scrollTo=ulJ7vhu7Nxvk

- [3] Lewis-Clark-Palmes. (n.d.). *CPE-201L-DSA-2-A/LAB_5.ipynb at main · Lewis-Clark-Palmes/CPE-201L-DSA-2-A*. GitHub. https://github.com/Lewis-Clark-Palmes/CPE-201L-DSA-2-A/blob/main/LAB_5.ipynb