Data Structure and Algorithm

Laboratory Activity No. 5

Implementation of Arrays

|  |  |
| --- | --- |
| *Submitted by:* | *Instructor:* |
| Ramos, Jan Lawrence M. | Engr. Maria Rizette H. Sayo |

August 23, 2025

# 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

# 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]

# Results

Present the visualized procedures done. Also present the results with corresponding data visualizations such as graphs, charts, tables, or image . Please provide insights, commentaries, or explanations regarding the data. If an explanation requires the support of literature such as academic journals, books, magazines, reports, or web articles please cite and reference them using the IEEE format.

Please take note of the styles on the style ribbon as these would serve as the style format of this laboratory report. The body style is Times New Roman size 12, line spacing: 1.5. Body text should be in Justified alignment, while captions should be center-aligned. Images should be readable and include captions. Please refer to the sample below:

# A screenshot of a computer program AI-generated content may be incorrect.

# Figure 1: screenshot of problem 1, problem 2, and problem 3.

# A screenshot of a computer AI-generated content may be incorrect.

# Figure 2: screenshot of the output of problem 1, problem 2, and problem 3.

# A screen shot of a computer program AI-generated content may be incorrect.

# Figure 1: screenshot of problem 4 and problem 5

# 

# Figure 1: screenshot of problem 4 and problem 5

# The programs successfully demonstrated fundamental array operations in Python:

# **Array Creation and Sum Calculation**: The program created an array of 10 integers (1 through 10) and correctly calculated the sum as 55, which aligns with the mathematical formula for the sum of the first n natural numbers: n(n+1)/2 = 10×11/2 = 55 .

# **Appending Operation**: The append() method successfully added the integer 11 to the end of the array, increasing the array length from 10 to 11 elements. This operation has a time complexity of O(1) as it simply adds an element to the end of the list .

# **Insertion Operation**: The insert() method placed the value 1.5 at index position 1 (before the second element). This operation requires shifting all subsequent elements, resulting in a time complexity of O(n) in the worst case .

# **Array Reversal**: The reverse() method successfully changed the order of elements from [5, 4, 3, 2, 1] to [1, 2, 3, 4, 5]. This operation has a time complexity of O(n) as it needs to swap n/2 elements .

# **Length Calculation**: The len() function correctly returned 5 as the length of the array [5, 4, 3, 2, 1]. This operation has a time complexity of O(1) as Python lists maintain their length as an attribute.

# These operations demonstrate Python's efficient handling of array manipulations, which is crucial for data processing tasks in scientific computing and data analysis.

# Conclusion

This laboratory exercise successfully demonstrated fundamental array manipulation operations in Python, confirming that the language's built-in list methods provide efficient and intuitive tools for data handling. The implementation of programs for creation, summation, appending, insertion, reversal, and length calculation highlighted key performance characteristics: appending and checking length are constant-time operations (O(1)), while insertion and reversal require linear time (O(n)) due to element shifting. These findings underscore the importance of selecting appropriate array operations to optimize efficiency in data-intensive applications, establishing these foundational skills as critical for scientific computing, data analysis, and software engineering where effective sequential data processing is paramount.

**References**

[1] "Academic Honor Code," CMU Policy Library, 2023. [Online].

[2] G. Strang, "Introduction to Linear Algebra," 5th ed., Wellesley-Cambridge Press, 2016.

[3] M. L. Hetland, "Beginning Python: From Novice to Professional," 3rd ed., Apress, 2023.

[4] T. H. Cormen et al., "Introduction to Algorithms," 4th ed., MIT Press, 2022.

[5] W. McKinney, "Python for Data Analysis," 3rd ed., O'Reilly Media, 2022.