Analysis of Data Structures and String Operations

---

1. Introduction

This report analyzes the time and space complexity of various data structures and string operations implemented in this project, including static arrays, dynamic arrays, and string manipulation functions.

---

2. Data Structures

2.1 Static Array

Description: A static array has a fixed size defined at its creation.

Time Complexity:

- **Insertion:** *O(1)* (if index is valid)
- **Deletion:** *O(1)* (if index is valid)
- **Traversal:** *O(n)*

Space Complexity: *O(n)*, where *n* is the size of the array.

2.2 Dynamic Array

Description: A dynamic array can grow or shrink in size as needed.

Time Complexity:

- **Insertion:** *O(n)* (in the worst case, when resizing occurs)
- **Deletion:** *O(n)* (due to shifting elements)
- **Traversal:** *O(n)*

Space Complexity: *O(n)*, but may have additional overhead due to capacity management.

---

3. String Operations

3.1 Concatenation

Description: Combines two strings into one.

Time Complexity: $O(m + n)$, where m and n are the lengths of the two strings.

Space Complexity: $O(m + n)$.

3.2 Substring

Description: Extracts a substring from a given string.

Time Complexity: $O(n)$, where n is the length of the substring being extracted.

Space Complexity: $O(n)$.

3.3 Comparison

Description: Compares two strings for equality.

Time Complexity: $O(\min(m, n))$, where m and n are the lengths of the strings.

Space Complexity: $O(1)$.

---

4. Character Frequency

Description: Counts the frequency of each character in a string.

Time Complexity: *O(n)*, where *n* is the length of the string.

Space Complexity: *O(k)*, where *k* is the number of unique characters.

---

5. Conclusion

This report provides an overview of the complexities associated with each data structure and operation implemented in this project. Understanding these complexities is crucial for optimizing performance and resource management in software development.