CAM2003C - Data Structures and Algorithms with C and C++

Lab-1: Introduction to Data Structures, Algorithm Analysis & Arrays

Objective:

- To familiarize students with algorithm writing, analysis, and implementation using C/C++.
- To perform asymptotic and a priori analysis.
- To implement basic array operations and solve problems using arrays.

Tasks:

Solve questions 1-4 in the lab. Questions 5 & 6 are take home assignments.

Programming Environment:

Use **Code::Block** software for writing and testing your codes

Deliverables:

Submit the following and also store these solutions to your respective Z:/ drive.

- Pseudo-code for all algorithm writing questions.
- Asymptotic analysis and a priori analysis report.
- C/C++ source code files for array programs.

Part A: Algorithm Writing & Analysis

1. **Write algorithms (pseudo-code) for the following problems:**
   a) Find the maximum and minimum element in a list of n integers.
   b) Count the number of odd and even numbers in an array of size n.
   c) Reverse a given array of integers of size n.
2. **A Priori Analysis**: For each algorithm above, determine:
   - The time complexity (Big O notation).
   - The space complexity.
   - Discuss whether the algorithm is optimal or can be improved.

Part B: Asymptotic Analysis

- **Write the asymptotic analysis neatly for each snippet.**
- **Mention:**

- Best-case, Worst-case, and Average-case complexities (where applicable).
- The total number of operations step by step.

3. **Given the following code snippet, determine its time complexity and justify your answer:**

a)

```
 for (int i = 0; i < n; i++) {
  for (int j = 0; j < n; j++) {
    printf("*");
  }
}
```

b)

```
void func(int n) {
  if (n == 1)
    return;
  func(n/2);
  func(n/2);
}
```

c)

```
int i = 1;
while (i < n) {
  printf("%d ", i);
  i = i * 2;
}
```

d)

```
for (int i = 0; i < n; i++) {
  for (int j = 0; j < n; j++) {
    for (int k = 0; k < n; k++) {
      printf("*");
    }
  }
}
```

e)

```
int fib(int n) {
    if (n <= 1)
        return n;
    return fib(n - 1) + fib(n - 2);
}
```

f)

```
printf("Hello World");
```

g)

```
for (int i = 0; i < n; i++) {
    printf("*");
}
for (int j = 0; j < n; j++) {
    for (int k = 0; k < n; k++) {
        printf("#");
    }
}
```

Part C: Array Operations in C/C++

4. **Write a C/C++ program to perform the following operations on arrays:**

    a) Traverse an array.
    b) Find the number of element (Length) of an array
    c) Search an element from the array.
    d) Find the largest element from the array.
    e) Find the smallest element from the array
    f) Insert an element at a given position in the array.
    g) Delete an element from a given position in the array.
    h) Search for an element using linear search.

Part D: Application-Based Problems

5. **Find the “Leader Elements” in an array:**
   An element is a leader if it is greater than all the elements to its right side. The rightmost element is always a leader.

Sample Input 1:

Enter the size of array: 6

Enter array elements: 16 17 4 3 5 2

Sample Output 1:

Leader elements are: 17 5 2

Sample Input 2:

Enter the size of array: 5

Enter array elements: 1 2 3 4 0

Sample Output 2:

Leader elements are: 4 0

Explanation:

For **Input 1**,

- Starting from the end: 2 (leader),
- 5 > 2 ⇒ 5 (leader),
- 3 < 5, skip.
- 4 < 5, skip.
- 17 > 5 ⇒ 17 (leader).
- 16 < 17, skip.

Hence leaders = 17 5 2 (printed in order of occurrence).

Part-E: Competitive Programming

6. **Solve the following problems at Neetcode platform. (Create your profile on https://leetcode.com/)**

a) Two Sum - LeetCode
   https://leetcode.com/problems/two-sum/description/

b) Best Time to Buy and Sell Stock – LeetCode
   https://leetcode.com/problems/best-time-to-buy-and-sell-stock/description/