**National University of Computer and Emerging Sciences**



Laboratory Manual

for

Data Structures Lab

|  |  |
| --- | --- |
| Lab Instructor(s) | Anosha Khan |
| Section | BCS-3G |
| Semester | Fall 2024 |

**Department of Computer Science**

FAST-NU, Lahore, Pakistan

**Objectives:**

In this lab, students will practice:

1. Recursion

**Question 1**

Write a recursive function *int findGCD(int a, int b)* that calculates the greatest common divisor (GCD) of two positive integers a and b using the Euclidean algorithm. The GCD is the largest number that can divide both a and b without leaving a remainder.

**Requirements:**

1. Use recursion to implement the Euclidean algorithm:

* If b is 0, the GCD is a.
* Otherwise, recursively call *findGCD(b, a % b).*

1. The function should handle cases where a is less than b.
2. Demonstrate the function with different pairs of values for a and b.

**Question 2**

Write a recursive function *int binarySearchInRange(int arr[], int low, int high, int target)* that performs a binary search on a sorted array to find a target value. In addition, the function should be able to search for the target only within a specified range of indices *(low to high),* not necessarily the entire array.

**Requirements:**

1. Implement the recursive binary search that returns the index of the target value if it is found within the given range [low, high].
2. If the target is not found, the function should return -1.
3. Demonstrate the function by calling it with different ranges to find the target value.

**Question 3**

Write a C++ program that generates all possible subsequences of a given string using recursion. Your program should implement a function that makes **two recursive calls** to explore both the inclusion and exclusion of each character in the string.

**Requirements:**

1. Define a recursive function void *generateSubsequences(string str, string current, int index)* that:
   * Takes a string str, a string current representing the current subsequence being built, and an integer index for tracking the position in the original string.
   * Makes **two recursive calls**:
     + One call should include the character at the current index in the subsequence.
     + The other call should exclude the character at the current index.
   * Print each subsequence when the end of the string is reached (i.e., when index equals the length of str).
2. In the main function:
   * Prompt the user to enter a string.
   * Call the generateSubsequences function and display all subsequences.

Output:

Enter a string:

abc Subsequences: "abc" "ab" "ac" "a" "bc" "b" "c" ""

**Question 4**

Write a C++ program that performs two key operations on a singly linked list:

1. **Calculate the Sum of Elements**: Implement a recursive function *int sumOfList(Node\* head)* that calculates and returns the sum of all elements in the singly linked list. Assume the singly linked list is defined using the following Node structure:

*struct Node { int data; Node\* next; };*

**2.Add an Element at the End of the List**: Create a recursive function *void addElementAtEnd(Node\*& head, int data)* that adds a new element at the end of the singly linked list. The function should create a new node with the given data and link it appropriately.

**Requirements:**

* In the *main function*, create a simple singly linked list with a few initial nodes.
* Call the *sumOfList* function to calculate and display the sum of the elements in the list.
* Use the *addElementAtEnd* function to add a new element to the end of the list and then display the updated list.