Data Analysis And Algorithm

University of Engineering and Technology, Lahore (New Campus)

Submitted to:

Mam Drakhshan Bokhat

Submitted by:

SOHAIB IKRAM (CS-621)

SHABEER JAN BUTT (CS-623)

TALHA MUSHTAQ (CS-636)

Department of Computer Science

Approach Documentation

Purpose:

The purpose of this code is to demonstrate the application of mathematical theorems - Master theorem and Muster theorem - to analyze and classify functions in the context of algorithmic complexity.

Code Structure:

1. Header and Namespace Declaration:

Includes necessary libraries such as `<iostream>` and `<cmath>`.
The `using namespace std;` statement is used to simplify code by allowing direct access to elements of the `std` namespace.

2. Function Definitions:

The code defines three functions:
`DivMasterTheorem`, `Div_ExtendedMasterTheorem` and `DecMusterTheorem`, each corresponding to a specific theorem.
These functions take parameters related to the coefficients and exponents of the given function, then classify it based on the theorem's conditions.

3. Function `isconstant`:

A utility function to check if a character represents a constant.

4. Main Function (`main`):

Displays a menu for the user to choose between two types of functions: Dividing Function and Decreasing Function.
Reads user input for coefficients and the function itself.
Calls the appropriate theorem function based on the user's choice and input.

Theorems Applied:

1. Master Theorem:

Used to analyze the time complexity of divide-and-conquer algorithms.
Classifies functions of the form `T(n) = aT(n/b) + f(n)`.

2. Extended Master Theorem:

An extension of the Master theorem for analyzing functions with more complex forms, including logarithmic factors.
Handles functions like `T(n) = aT(n/b) + f(n)` with additional logarithmic terms.

3. Muster Theorem:

A theorem used for functions where the recursion decreases by a constant factor each time.
Analyzes functions of the form `T(n) = aT(n-b) + f(n)` with different conditions based on the values of `a` and `b`.

User Interaction:

1. Menu Selection:

The user is prompted to select between Dividing Function and Decreasing Function.
Input validation ensures the user selects a valid option.

2. Input Handling:

User inputs coefficients `a` and `b` along with the function `f(n)` as per the chosen theorem.
Additional validation checks for the presence of logarithmic terms in the function.

3. Output:

The program outputs the classification of the function based on the chosen theorem.

The output includes the time complexity expression and the theorem used for classification.

Conclusion:

This code provides a practical demonstration of how mathematical theorems can be applied to analyze and categorize functions commonly encountered in algorithm analysis. It facilitates understanding the time complexity of algorithms by providing a systematic approach to classify functions based on their growth rates.