Assignment 5

Course: CS_413 - Analysis of Algorithms
Assignment: 5
Date: 04/10/2025
Group Member(s): 1) Aryan Jigneshbhai Bhagat (sl5310)
2) Moksha Kumudbhai Shah (bp4199)

Aim: The goal of this assignment is to design, analyze, and implement a divide and conquer algorithm to count the number of odd elements in an array of n integers. The solution should run in O(n) time.

Problem Statement: Given an array of n integers, design a divide-and-conquer algorithm to count how many of those integers are odd.

The program should:

- Divide the array into two subarrays of (approximately) equal sizes.
- Recursively determine the number of odd elements in each subarray.
- Combine the results by summing those two counts.

A formal proof or clear derivation must be provided to show that the total running time of the algorithm is O(n).

Solution:

1) **Base Case:**
   If there is only one element in the portion of the array being considered: Return 1 if it is odd; otherwise return 0.

2) **Recursive Step:**
   If there are more than one element in the portion of the array:
   a) Split the array portion into a left half and a right half.
   b) Recursively count the number of odd elements in the left half.
   c) Recursively count the number of odd elements in the right half.
   d) Combine the results by adding these two counts.

Time Complexity Analysis:

Let T(n) be the time complexity for an input array of size n. Using the divide and conquer approach described:

1. We split the array into two subproblems of size n/2.
2. We solve each subproblem recursively.
3. We combine the results in constant time O(1) (just adding two counts).

Hence, the **recurrence relation** is:

$$T(n) = 2*T(n/2) + O(1)$$

By the **Master Theorem** (or standard recurrence tree analysis), this simplifies to:

$$T(n)=O(n).$$

Thus, the algorithm is **linear** in n.

Worst Case Time Complexity is also **O(n).**

C++ Code Implementation:

```
// Submitted by : Aryan Jigneshbhai Bhagat - NetID: sl5310, & Moksha
Kumudbhai Shah - NetID: bp4199

// CS_411 - Assignment 5 - C++ program to count the number of odd
elements using divide and conquer

#include <bits/stdc++.h>
using namespace std;

// Function to count the number of odd elements in an array using
divide-and-conquer
int countOdd(int arr[], int left, int right) {
    // Base case: if the array has one element
    if (left == right)
        return (arr[left] % 2 != 0) ? 1 : 0;
```

```cpp
    int mid = left + (right - left) / 2;
    // Recursively count odd elements in the left and right halves
    int leftCount = countOdd(arr, left, mid);
    int rightCount = countOdd(arr, mid + 1, right);

    // return the total count of odd elements
    return leftCount + rightCount;
}

int main() {
    // Test cases

    // Test-1: {1, 12, 34, 5, 7}
    // Expected output: 5 (1, 5, 7 are odd)
    int test1[] = {1, 12, 34, 5, 7};
    int n1 = sizeof(test1) / sizeof(test1[0]);

    // Test-2: {7}
    // Expected output: 1 (7 is odd)
    int test2[] = {7};
    int n2 = sizeof(test2) / sizeof(test2[0]);

    // Count the number of odd elements in the array
    int oddCount = countOdd(test1, 0, n1 - 1);
    cout << "#Test-1: Number of odd elements: " << oddCount << endl;

    oddCount = countOdd(test2, 0, n2 - 1);
    cout << "#Test-2: Number of odd elements: " << oddCount << endl;

    return 0;
}
```