**Assignment -4**

**Name:** Bhavin Patil

**Roll No.:** 66

**Class:** TY CS-D

**Assignment based on finding majority elements.**

1. **Boyer’s Moore Algorithms**

**Code:**

#include <bits/stdc++.h>

using namespace std;

int majorityElement(int arr[], int n)

{

    int count = 0, majorityElement = -1;

    for (int i = 0; i < n; i++)

    {

        if (count == 0)

        {

            majorityElement = arr[i];

            count = 1;

        }

        else if (majorityElement == arr[i])

        {

            count++;

        }

        else

        {

            count--;

        }

    }

    return majorityElement;

}

int main()

{

    cout << "Enter Size of Array: " << endl;

    int n;

    cin >> n;

    cout << "Enter Elements in the Array : " << endl;

    int arr[n];

    for (int i = 0; i < n; i++)

    {

        cin >> arr[i];

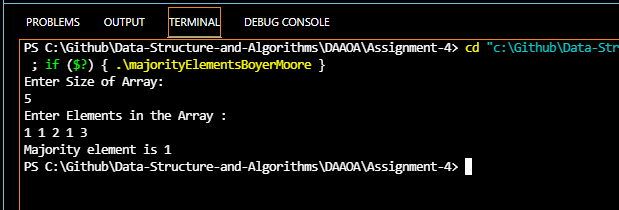
    }

    cout << "Majority element is " << majorityElement(arr, n) << endl;

    return 0;

}

**Output:**

****

1. **Using Sort Function**

**Code:**

#include <bits/stdc++.h>

using namespace std;

int majorityElement(int arr[], int n)

{

    sort(arr, arr + n);

    int count = 1, maxCount = 0, majorityElement = -1;

    for (int i = 1; i < n; i++)

    {

        if (arr[i] == arr[i - 1])

        {

            count++;

        }

        else

        {

            if (count > maxCount)

            {

                maxCount = count;

                majorityElement = arr[i - 1];

            }

            count = 1;

        }

    }

    if (count > maxCount)

    {

        majorityElement = arr[n - 1];

    }

    return majorityElement;

}

int main()

{

    cout << "Enter Size of Array: " << endl;

    int n;

    cin >> n;

    cout << "Enter Elements in the Array : " << endl;

    int arr[n];

    for (int i = 0; i < n; i++)

    {

        cin >> arr[i];

    }

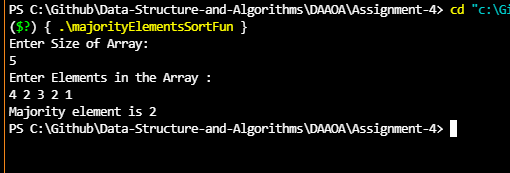
    cout << "Majority element is " << majorityElement(arr, n) << endl;

    return 0;

}

// TC - O(n log n)

**Output:**

****

1. **Using Hashmap**

**Code:**

#include <bits/stdc++.h>

using namespace std;

int majorityElement(int arr[], int n)

{

    unordered\_map<int, int> count;

    for (int i = 0; i < n; i++)

    {

        count[arr[i]]++;

        if (count[arr[i]] > n / 2)

        {

            return arr[i];

        }

    }

    return -1;

}

int main()

{

    cout << "Enter Size of Array: " << endl;

    int n;

    cin >> n;

    cout << "Enter Elements in the Array : " << endl;

    int arr[n];

    for (int i = 0; i < n; i++)

    {

        cin >> arr[i];

    }

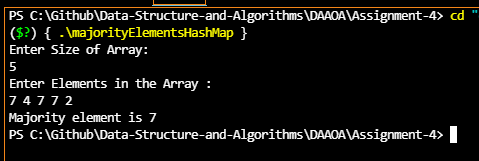
    cout << "Majority element is " << majorityElement(arr, n) << endl;

    return 0;

}

// TC- O(n)

**Output:**

****

1. **Using Divide & Conquer with Recursion**

**Code:**

#include <bits/stdc++.h>

using namespace std;

int majority\_count(vector<int> &arr, int i, int j, int maj)

{

    int cc = 0;

    for (int k = i; k <= j; k++)

    {

        if (arr[k] == maj)

        {

            cc++;

        }

    }

    return cc;

}

int majority\_fun(vector<int> &arr, int i, int j)

{

    if (i == j)

    {

        return arr[i];

    }

    int mid = i + (j - i) / 2;

    int left = majority\_fun(arr, i, mid);

    int right = majority\_fun(arr, mid + 1, j);

    if (left == right)

    {

        return left;

    }

    int left\_count = majority\_count(arr, i, j, left);

    int right\_count = majority\_count(arr, i, j, right);

    if (left\_count > right\_count)

    {

        return left;

    }

    else

    {

        return right;

    }

}

int main()

{

    int n;

    cout << "Enter size of Array: " << endl;

    cin >> n;

    vector<int> arr(n);

    int check = 0;

    cout << "Enter Elements: " << endl;

    for (int i = 0; i < n; i++)

    {

        cin >> arr[i];

    }

    int element = majority\_fun(arr, 0, n - 1);

    for (int i = 0; i < n; i++)

    {

        if (arr[i] == element)

        {

            check++;

        }

    }

    if (check > (n / 2))

    {

        cout << "Majority Element is: " << element;

    }

    else

    {

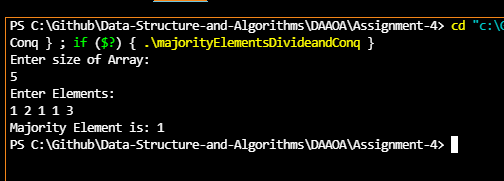
        cout << "Majority Element does not exist" << endl;

    }

    return 0;

}

**Output:**

****