**DIVIDE AND CONQUER**

PROBLEM 2:

2-MAJORITY ELEMENT

AIM:

Given an array nums of size n, return the majority element.

The majority element is the element that appears more than ⌊n / 2⌋ times. You may assume that the majority element always exists in the array.

CODE:

#include <stdio.h>

int countInRange(int nums[], int l, int r, int num) {

int count = 0;

for (int i = l; i <= r; i++) {

if (nums[i] == num) {

count++;

}

}

return count;

}

int majorityElementRec(int nums[], int l, int r) {

if (l == r) {

return nums[l];

}

int mid = l + (r - l) / 2;

int leftMajority = majorityElementRec(nums, l, mid);

int rightMajority = majorityElementRec(nums, mid + 1, r);

if (leftMajority == rightMajority) {

return leftMajority;

}

int leftCount = countInRange(nums, l, r, leftMajority);

int rightCount = countInRange(nums, l, r, rightMajority);

return leftCount > rightCount ? leftMajority : rightMajority;

}

int majorityElement(int nums[], int size) {

return majorityElementRec(nums, 0, size - 1);

}

int main() {

int n;

scanf("%d", &n);

int nums[n];

for (int i = 0; i < n; i++) {

scanf("%d", &nums[i]);

}

int result = majorityElement(nums, n);

printf("%d\n", result);

return 0;

}:

INPUT AND OUTPUT:

