Algorithm: majorityElements

Input: An array of integers nums.

Output: A list of integers representing the majority elements.

* Initialize an empty list result to store majority elements.
* Check if the array nums is null or empty. If true, return the empty result list.
* Initialize two candidates (candidate1 and candidate2) and their corresponding counts (count1 and count2) to 0 and 1, respectively.
* Iterate through each element num in the array nums:

1. If num is equal to candidate1, increment count1.
2. If num is equal to candidate2, increment count2.
3. If count1 is 0, update candidate1 to the current num, and set count1 to 1.
4. If count2 is 0, update candidate2 to the current num, and set count2 to 1.
5. If none of the above conditions is met, decrement both count1 and count2.

* After the first pass, reset counts (count1 and count2) and re-count occurrences of candidates (candidate1 and candidate2) in the array nums to verify their majority status.
* If the count of candidate1 is greater than n / 3, where n is the length of the array, add candidate1 to the result list.
* If the count of candidate2 is greater than n / 3, add candidate2 to the result list.
* Return the result list.

Algorithm: main

Input: None.

Output: Display the majority elements in the entered array.

* Create a Scanner object to take input from the user.
* Prompt the user to enter the size of the array.
* Create an array of integers with the specified size.
* Prompt the user to enter the elements of the array.
* Call the majorityElements method with the array.
* Display the majority elements obtained from the result list.