Write a C program to determine whether a majority element exists in a given array of integers. A majority element is defined as an element that appears more than ⌊n/2⌋ times in the array. Extend the problem with additional complexities such as edge case handling, dynamic inputs, and efficient processing.

Detailed Requirements:

1. Input Constraints:
   - The program should accept an array of integers of size NNN, where NNN is entered by the user.
   - The size NNN must be between 10 and 200, inclusive. Validate the input and prompt the user until it is valid.
   - Allow both positive and negative integers in the array.
2. Majority Element Definition:
   - A majority element is one that appears more than ⌊n/2⌋ times in the array. For example:
     - If N=7, an element must appear more than ⌊7/2⌋=3 times to be considered a majority element.
3. Algorithm Complexity:
   - Implement the solution using a two-step process:
     - Identify a candidate element for the majority using a voting algorithm (e.g., Moore's Voting Algorithm).
     - Verify if the candidate is indeed a majority element by counting its occurrences.
4. Additional Features:
   - If a majority element exists:
     - Display the element and its count.
     - Display all indices where the majority element appears.
     - If no majority element exists:
       - Display "No majority element exists in the array."

        - Handle arrays with multiple elements appearing the same number of times (e.g., ties) appropriately.

4. Output Requirements:
    - Display the original array.
    - Show the candidate element (if any) and the results of the verification step.
    - Display the majority threshold ⌊n/2⌋.
5. Performance Considerations:
    - Use a single pass to find the candidate element and another single pass for verification.
    - Ensure the program works efficiently for arrays close to the maximum size of 200.
6. Edge Case Handling:
    - Arrays where all elements are the same (e.g., [5, 5, 5, 5, 5]).
    - Arrays where no majority element exists (e.g., [1, 2, 3, 4, 5, 6, 7]).
    - Arrays with negative numbers and zeros (e.g., [-1, -1, 0, 0, 0, -1, -1]).
    - Arrays with ties but no majority (e.g., [1, 1, 2, 2, 3, 3, 4]).
7. Program Constraints:
    - Use decision-making statements (if, else if, else) to handle verification and output.
    - Use functions for input validation, majority element identification, and verification.

Example Execution:

Input 1:

- Array Size (N): 9
- Array Elements: [3, 3, 4, 2, 3, 3, 3, 5, 3]

Processing:

- Majority Threshold: ⌊9/2⌋=4

- Candidate Element (Using Voting): 333
- Count of Candidate: 666
- Indices of Candidate: [0, 1, 4, 5, 6, 8]

Output:

- Original Array: [3, 3, 4, 2, 3, 3, 3, 5, 3]
- Majority Threshold: 4
- Majority Element: 3
- Count: 6
- Indices: [0, 1, 4, 5, 6, 8]

Input 2:

- Array Size (N): 7
- Array Elements: [1, 2, 3, 4, 5, 6, 7]

Processing:

- Majority Threshold: $\lfloor 7/2 \rfloor=3$
- Candidate Element: None
- Verification: No element appears more than 3 times.

Output:

- Original Array: [1, 2, 3, 4, 5, 6, 7]
- Majority Threshold: 3
- Majority Element: No majority element exists in the array.

```
#include <stdio.h>

int main()
{
   int N;
```

```
// Array size validation
do
{
    printf("Enter the array size: ");
    scanf("%d", &N);
    if(N < 10 || N >200)
        printf("Invalid array size\n");
} while(N < 10 || N > 200);

// Input of array elements
int a[N], i;
printf("Enter the elements: ");
for(i = 0; i < N; i++)
    scanf("%d", &a[i]);

// Display of the original array
printf("Original array: [");
for(i = 0; i < N; i++)
    printf("%d%s", a[i], (i == N - 1) ? "]" : ", ");

// Majority threshold calculation
int t = N / 2;
printf("\nMajority threshold: %d\n", t);

// Majority candidate identification and verification
int max_count = 0, ele = -1, j;
for(i = 0; i < N; i++)
```

```
{
    int count = 0;
    for(j = 0; j < N; j++)
    {
        if(a[i] == a[j])
            count++;
    }
    if(count > max_count)
    {
        max_count = count;
        ele = a[i];
    }
}

if(max_count > t)
{
    printf("Majority element: %d\n", ele);
    printf("Count: %d\n", max_count);
    printf("Indices: ");
    for(i = 0; i < N; i++)
    {
        if(a[i] == ele)
            printf("%d ",i);
    }
    printf("\n");
}
else
```

```
        printf("No majority element exists\n");

    return 0;
}
```

O/P:

```
Enter the array size: 10
Enter the elements: 3 3 4 2 3 3 3 5 3 3
Original array: [3, 3, 4, 2, 3, 3, 3, 5, 3, 3]
Majority threshold: 5
Majority element: 3
Count: 7
Indices: 0 1 4 5 6 8 9
```