```c
#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/wait.h>


// Function to sort the array

void sortArray(int arr[], int n) {

    for (int i = 0; i < n - 1; i++) {

        for (int j = 0; j < n - i - 1; j++) {

            if (arr[j] > arr[j + 1]) {

                int temp = arr[j];

                arr[j] = arr[j + 1];

                arr[j + 1] = temp;

            }

        }

    }

}


// Function to reverse the array

void reverseArray(int arr[], int n) {

    int start = 0;

    int end = n - 1;

    while (start < end) {

        int temp = arr[start];

        arr[start] = arr[end];

        arr[end] = temp;

        start++;

        end--;

    }

}
```

```
int main() {
    int n;
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);
    int arr[n];

    printf("Enter the elements of the array: ");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    // Forking a child process
    pid_t pid = fork();

    if (pid < 0) {
        // Error occurred
        fprintf(stderr, "Fork failed\n");
        return 1;
    } else if (pid == 0) {
        // Child process
        char *args[n + 2]; // Extra space for program name and NULL terminator
        args[0] = "./display_reverse"; // Name of the program to execute

        // Converting array elements to strings
        for (int i = 0; i < n; i++) {
            args[i + 1] = (char *)malloc(10 * sizeof(char)); // Assuming the max length of an integer is 10
            sprintf(args[i + 1], "%d", arr[i]);
        }
        args[n + 1] = NULL; // Terminating the argument list

        // Executing the display_reverse program
        execve(args[0], args, NULL);

    } else {
        // Parent process
        wait(NULL); // Wait for the child to finish

        // Sort the array
        sortArray(arr, n);

        // Displaying the sorted array
        printf("Sorted Array: ");
        for (int i = 0; i < n; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");

        // Reverse the sorted array
        reverseArray(arr, n);

        // Displaying the array in reverse order
        printf("Array in Reverse Order: ");
        for (int i = 0; i < n; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");
    }

    return 0;
}
```