```c
#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/wait.h>


// Function to sort an array (using Bubble Sort)

void bubbleSort(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;

            }

        }

    }

}


int main() {

    int n;

    printf("Enter the number of integers to be sorted: ");

    scanf("%d", &n);


    int *arr = (int *)malloc(n * sizeof(int));

    if (arr == NULL) {

        printf("Memory allocation failed\n");

        return 1;

    }


    printf("Enter %d integers:\n", n);

    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");
        free(arr);
        return 1;
    } else if (pid == 0) {
        // Child process

        // Demonstrate orphan state by delaying child execution
        sleep(2);

        // Sort the integers using Bubble Sort
        bubbleSort(arr, n);
        printf("Child Process Sorted the Integers using Bubble Sort:\n");
        for (int i = 0; i < n; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");

        free(arr); // Free memory allocated for the array
        return 0; // Child process terminates
    } else {
        // Parent process

        // Demonstrate zombie state by delaying parent execution
        sleep(5);

        // Wait for the child process to finish
        wait(NULL);

        // Sort the integers using Selection Sort
        for (int i = 0; i < n - 1; i++) {
            int min_index = i;
            for (int j = i + 1; j < n; j++) {
                if (arr[j] < arr[min_index]) {
                    min_index = j;
                }
            }
            // Swap the elements
            int temp = arr[i];
            arr[i] = arr[min_index];
            arr[min_index] = temp;
        }
        printf("Parent Process Sorted the Integers using Selection Sort:\n");
        for (int i = 0; i < n; i++) {
            printf("%d ", arr[i]);
        }
        printf("\n");

        free(arr); // Free memory allocated for the array
    }

    return 0;
}
```