```c
#include <stdio.h>
#include <sys/types.h>
#include <unistd.h>
#include <stdlib.h>
#include <sys/wait.h>

void bubble_sort_ascending(int arr[], int n) {
    int i, j, temp;
    for (i = 0; i < n; i++) {
        for (j = 0; j < n - 1; j++) {
            if (arr[j] > arr[j + 1]) {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
    printf("\nAscending Order:\n");
    for (i = 0; i < n; i++)
        printf("%d\t", arr[i]);
    printf("\n\n");
}

void bubble_sort_descending(int arr[], int n) {
    int i, j, temp;
    for (i = 0; i < n; i++) {
        for (j = 0; j < n - 1; j++) {
            if (arr[j] < arr[j + 1]) {
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
        }
    }
    printf("\nDescending Order:\n");
    for (i = 0; i < n; i++)
        printf("%d\t", arr[i]);
    printf("\n\n");
}

void fork_example() {
    int arr[25], n, i;
    printf("Enter the number of values in the array: ");
    scanf("%d", &n);
    printf("Enter the array elements:\n");
    for (i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    pid_t pid = fork();

    if (pid == 0) { // Child process
        sleep(1);
        printf("\nChild process:\n");
        printf("Child process ID = %d\n", getpid());
        bubble_sort_descending(arr, n);
        printf("Parent process ID = %d\n", getppid());
    } else if (pid > 0) { // Parent process
        printf("\nParent process:\n");
        printf("Parent process ID = %d\n", getpid());
```

```
        bubble_sort_ascending(arr, n);

        // Wait for child to complete
        wait(NULL);

        printf("\nSorting completed.\n");
    } else { // Fork failed
        printf("Fork failed!\n");
        exit(1);
    }
}

int main() {
    fork_example();
    return 0;
}
```