#include <stdio.h>

#include <sys/types.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/wait.h>

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

int i, j, temp;

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

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

if (arr[i] > arr[j]) {

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

printf("Ascending order arrangement: ");

for (i = 0; i < n; i++) {

printf("%d ", arr[i]);

}

printf("\n");

}

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

int i, j, temp;

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

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

if (arr[i] < arr[j]) {

temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

printf("Descending order arrangement: ");

for (i = 0; i < n; i++) {

printf("%d ", arr[i]);

}

printf("\n");

}

int main() {

int n, i;

pid\_t pid;

printf("Enter number of elements: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid array size.\n");

exit(1);

}

int arr[n];

printf("Enter array elements:\n");

for (i = 0; i < n; i++) {

printf("a[%d]: ", i);

scanf("%d", &arr[i]);

}

printf("\n");

pid = fork();

if (pid < 0) {

perror("Fork error");

exit(1);

} else if (pid == 0) {

// Child process

printf("Child process ID: %ld\n", (long)getpid());

printf("\nChild process sorting in ascending order...\n");

asc(arr, n);

exit(0);

} else {

// Parent process

printf("Parent process ID: %ld\n", (long)getpid());

printf("\nParent process sorting in descending order...\n");

desc(arr, n);

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

printf("\nParent process displaying process list:\n");

system("ps -elf");

exit(0);

}

}