import java.util.Arrays;

import java.util.Scanner;

public class ProcessControlDemo {

    public static void main(String[] args) throws InterruptedException {

        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of integers to sort: ");

        int n = scanner.nextInt();

        int[] numbers = new int[n];

        System.out.println("Enter the integers:");

        for (int i = 0; i < n; i++) {

            numbers[i] = scanner.nextInt();

        }

        Thread childThread = new Thread(new ChildProcess(numbers));

        childThread.start();

        System.out.println("Parent process is sorting the array...");

        Arrays.sort(numbers);

        System.out.println("Parent sorted array: " + Arrays.toString(numbers));

        System.out.println("Parent process is delaying before waiting for the child process (simulating zombie state)...");

        Thread.sleep(5000);

        childThread.join();

        System.out.println("Parent process waited for the child process to finish (Zombie state resolved).");

        System.out.println("Parent process is terminating, leaving the child process as an orphan...");

    }

}

class ChildProcess implements Runnable {

    private int[] numbers;

    public ChildProcess(int[] numbers) {

        this.numbers = Arrays.copyOf(numbers, numbers.length);

    }

    @Override

    public void run() {

        try {

            Thread.sleep(3000);

            System.out.println("Child process sorting the array...");

            Arrays.sort(numbers);

            System.out.println("Child sorted array: " + Arrays.toString(numbers));

            System.out.println("Child process has finished but is waiting for parent (simulated zombie state).");

            Thread.sleep(5000);

            System.out.println("Child process finished after parent terminated (Orphan state).");

        } catch (InterruptedException e) {

            e.printStackTrace();

        }

    }

}