Name : Mu In Nasif
Roll : 001910501036
Class : BCSE II
Sem : 3
Session : 2020-2021

Data Structures and Algorithms Assignment Set 2

Question 8:
Find whether an array is sorted or not, and the sorting order.

Solution Approach:

We can use the following steps to identify the “sortedness” of an array (we assume elements of the array can be compared)
1. An array of length 0, 1 and 2 is sorted by default.
2. For an input array A of length >= 2, for each possible consecutive pair A[i], A[i+1] we can find the order in which they are arranged (ascending, descending or equal).
3. An array is sorted if all such consecutive pairs are either ascending or equal, or descending or equal, but not ascending or descending. (Example: <3, 3, 3, 5, 6> is sorted even though some consecutive pairs are equal, but <3. 3. 3. 5. 6. 4> isn’t because some pairs are ascending and one pair is descending.)
4. If all consecutive pairs are equal or ascending, the array is in ascending order. If all consecutive pairs are equal or descending, the array is descending order. And obviously if all consecutive pairs are equal, all the elements are same (the array can said to be in both ascending in descending order).

Pseudocode:
We assume a state type sortedness which has the values ASCENDING, DESCENDING, EQUAL and UNSORTED to indicate possible outputs of the procedure.

```
check_2state(a, b) {
        //Return a sortedness value for an array
        if a < b: return ASCENDING
        elif a == b: return EQUAL
        else return DESCENDING
}

check_sorted(A) {
        //Returns state of type sortedness
        if length of A is either 0 or 1: return EQUAL //Because we have nothing to compare to
        overall_state = EQUAL //Initially we assume all elements are equal.
        for i = 1 to (length of A)-1 inclusive { //Assuming 1-based indexes.
                current_state = check_2state(A[i], A[i+1]) //State of current pair
                if overall_state was EQUAL: overall_state = current_state
                        //We can change from EQUAL to EQUAL, ASCENDING or
                        DESCENGING
                elif overall_state does not equal current_state and current state is not EQUAL:
                return UNSORTED //If we change from DESCENDING to ASCENDING or vice
                versa, input is unsorted
        }
        return overall_state
}
```

Code: a2q8_sortedness.h is a C header containing implementations for the algorithms discussed above. a2q8_test.c is a testing driver program for the header. Both conform to ANSI C89