1) (10 pts) DSN (Recursive Coding)

Write a recursive function that returns 1 if an array of size n is in sorted order from smallest to largest with all values less than or equal to max, and 0 otherwise. Note: If array a stores 3, 6, 7, 7, 12, then `isSorted(a, 12, 5)` should return 1 but `isSorted(a, 11, 5)` should return 0. If array b stores 3, 4, 9, 8, then `isSorted(b, 20, 4)` should return 0, since 9 is bigger than 8 but appears before it.

```
int isSorted(int* array, int max, int n) {
    if (n == 0) return 1;                          // 3 pts
    if (array[n-1] > max) return 0;                // 3 pts
    return isSorted(array, array[n-1], n-1);       // 4 pts
}
```

Grading Notes: Lots of ways to solve this problem. Map their solution to the cases laid out here:

0/1 size case: 3 pts
Checking last element too big: 3 pts
Recursive call if last pair (or first pair) is valid: 4 pts

The points are mapped so that many items could be off by a tiny bit, so adjust the points as necessary. If there are two small errors in two different places each worth 2 pts, just take off 1 point for one of the two errors and let the other one go.

Note: If max was not a parameter to the function and we wanted to simply write a recursive function that determined if the array specified by the parameters was sorted or not, the following function would suffice:

```
int isSortedAlt(int* array, int n) {
    if (n < 2) return 1;
    if (array[n-1] < array[n-2]) return 0;
    return isSortedAlt(array, n-1);
}
```