Insertion Sort

ZPRAC-16-17-Lab7

[40 Points]
Insertion Sort

ANNOUNCEMENT: Up to 20% marks will be allotted for good programming practice. These include
- Comments for non trivial code
- Indentation: align your code properly
- Use of functions

Write a C program to sort the given list of costs in increasing order according to the algorithm described below.

Algorithm : Insertion Sort
==========

Intuition :

Let us first try to intuitively see how this algorithm works.
Imagine that you are playing a card game. You're holding the cards in your hand, and these cards are sorted. The dealer hands you exactly one new card. You have to put it into the correct place so that the cards you're holding are still sorted. The new card could be smaller than some of the cards you're already holding, and so you go down the line, comparing the new card against each card in your hand, until you find the place to put it. You insert the new card in the right place, and once again, your hand holds fully sorted cards. Then the dealer gives you another card, and you repeat the same procedure. Then another card, and another card, and so on, until the dealer stops giving you cards.
This is exactly how the given algorithm sorts these numbers.

Logic :

At each point during the execution of this algorithm, you split your complete list into a sorted sublist and an unsorted sublist. And gradually over the iterations, you keep adding members from the unsorted sublist to the sorted one until the entire list becomes sorted.

For example, say you have a list of numbers
1, 7, 2, 8, 9, 3

1. Initially the the sorted sublist is empty and unsorted sublist is : 1, 7, 2, 8, 9, 3 (which is the complete list)
2. Now you pick the first element from the unsorted sublist(which is 1 in this case) and place it appropriately in the sorted sublist. Overall after this step,
Sorted sublist : 1
Unsorted sublist : 7, 2, 8, 9, 3
We can also see the concatenated list which is simply joining the two sublists into a single list.
Concatenated list : 1, 7, 2, 8, 9, 3

3. Again you pick the first element from the unsorted sublist(which is 7 now) and again place it appropriately in the sorted sublist. Overall after this step,
Sorted sublist : 1, 7
Unsorted sublist : 2, 8, 9, 3
Concatenated list : 1, 7, 2, 8, 9, 3

4. Repeat the same procedure as step 3. and after this step,
Sorted sublist : 1, 2, 7
Unsorted sublist : 8, 9, 3
Concatenated list : 1, 2, 7, 8, 9, 3

5. Continue this way after 3 more steps,
Sorted sublist : 1, 2, 3, 7, 8, 9
Unsorted sublist :
Concatenated list : 1, 2, 3, 7, 8, 9

So overall your entire list becomes sorted in this fashion. Your code should execute in similar manner **printing the concatenated list at each step** .

IMPORTANT NOTE : You are encouraged to not use a separate array for the sorted sub-list. It will be more efficient to sort the array "in-place" i.e. use the same array space for the sorted sub-list. Some marks will be deducted if you use a second array.

Format :
========
You are given the number of items n followed by a list of n integers denoting the costs. It is given that the costs are between 0 and 19, both inclusive. The number of items can be at most 10000. Overall,

1 <= n <= 10000
0 <= Value of each cost <= 19

Example :
=========

Input:
6
1 7 2 8 9 3
Output:
1,7,2,8,9,3,
1,7,2,8,9,3,
1,2,7,8,9,3,
1,2,7,8,9,3,
1,2,7,8,9,3,
1,2,3,7,8,9,