Lists Assignment

Write a Python program that allows a user to play a game that involves reversing slices of numbers in a list. Specifically, the program should first generate a list consisting of 10 random integers between 1 and 100 (it's okay if a number occurs more than once in the array). Next, it should display the list and prompt the user to enter a number, which the program inputs. I'll call this number `n`, but your program can use any variable name you like. The program then displays to the user a list that in effect reverses the first `n` numbers of the original list and leaves the remaining numbers as they were. This process of inputting a number and reversing that many elements in the list continues until the user succeeds in putting the list in ascending numeric order, when the game ends.

Here's an example of playing the game:

```
Current list of numbers: [44, 90, 30, 87, 93, 49, 74, 7, 3, 82]
How many numbers would you like to reverse? 5
Current list of numbers: [93, 87, 30, 90, 44, 49, 74, 7, 3, 82]
How many numbers would you like to reverse? 10
Current list of numbers: [82, 3, 7, 74, 49, 44, 90, 30, 87, 93]
How many numbers would you like to reverse? 7
Current list of numbers: [90, 44, 49, 74, 7, 3, 82, 30, 87, 93]
How many numbers would you like to reverse? 9
Current list of numbers: [87, 30, 82, 3, 7, 74, 49, 44, 90, 93]
How many numbers would you like to reverse? 8
Current list of numbers: [44, 49, 74, 7, 3, 82, 30, 87, 90, 93]
How many numbers would you like to reverse? 6
Current list of numbers: [82, 3, 7, 74, 49, 44, 30, 87, 90, 93]
How many numbers would you like to reverse? 7
Current list of numbers: [30, 44, 49, 74, 7, 3, 82, 87, 90, 93]
How many numbers would you like to reverse? 4
Current list of numbers: [74, 49, 44, 30, 7, 3, 82, 87, 90, 93]
How many numbers would you like to reverse? 6
Final sorted list: [3, 7, 30, 44, 49, 74, 82, 87, 90, 93]
>>>
```

Your program's output should be formatted as shown. In particular, each list of numbers that is output should be preceded on the same line by a short description.

This example illustrates a strategy for winning the game that always succeeds, although it does not necessarily minimize the number of moves made by the user.

Your program must not use Python features that we have not yet covered, and the program must use the list functions and slicing that we have covered so as to avoid a brute-force solution. For instance, when testing whether or not the user has succeeded in putting the numbers in ascending order, the user's list should **not** be tested in a brute-force way by comparing the first element to the second, the second element to the third, and so on, ensuring that each is no larger than the next. Instead, the user's list should be compared with a list that has been sorted using a call to the `sort()` method. Similarly, the `reverse()` method should be used when a slice of a list needs to be reversed.

Hint: POGIL Activity 18 shows how to using slicing to easily make a copy of a list. Making list copies will probably be very useful in your program.

Grading

0: Program does not run, uses Python features not yet covered, or uses a brute-force approach rather than using appropriate list methods
5: Program runs and conforms perfectly with specifications above
4: Program runs and is mostly correct but deviates from the above specification in one way
3: Program runs and is mostly correct but deviates from the above specification in two ways
0: Program does not run or deviates from the above specification in three or more ways