Lab 10 Report: Insertion Sort

JJ Shepherd

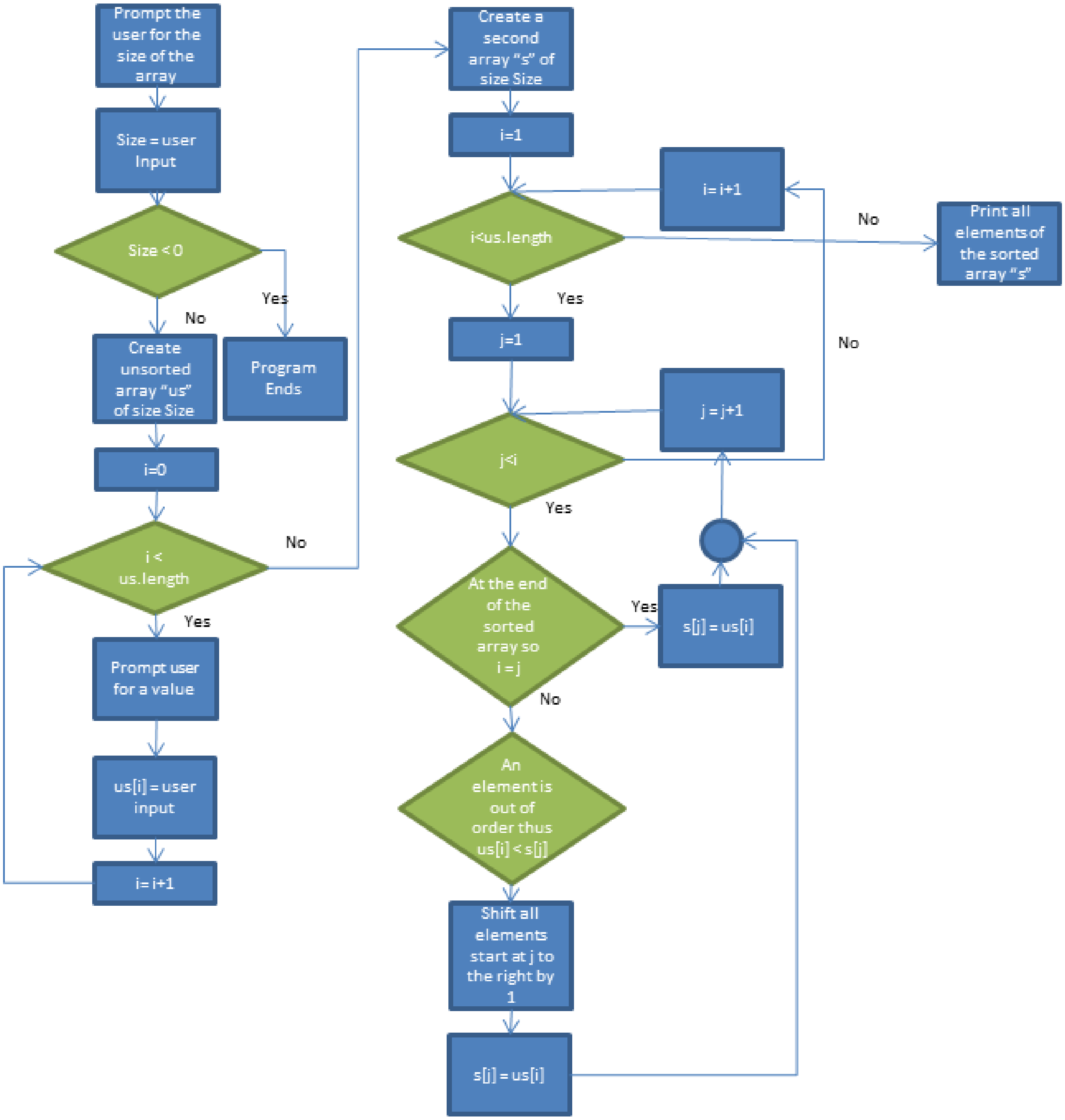
Problem

We had to write a program in which a user populated an array of integers and then it was sorted using insertion sort. Finally, the program printed out the original and sorted array to the console.

Solution

In order to create the solution, we had to collect data from the user. As the problem requires us to create an array, we had to first ask the user for the array size. We choose to do this as arrays have to be given a size before they are constructed in memory. Next, we validated the size given was correct, and if it was not (such as a negative value) then the program immediately stopped. However, if the size is valid, we then constructed the array of that size. Next, we asked the user to enter values, and those values populated the array.

Once the array had been populated, we began the insertion sort algorithm. First, we created a new array that was the same size as the other. Next, we began adding values to the new array one-by-one. The first element is added to the first position of the new array. After that, each new value to be added to the new array was the compared against the other values found in it. Once we found the correct position for that value, we inserted it. In the event, that the value had to be inserted somewhere in the middle of the array, we first shifted the values one index over before inserting. Finally, we printed both arrays.


Prompt the user for the size of the array
Size = user Input
Size < 0
Yes
No
Program Ends
Create unsorted array "us" of size Size
i=0
i < us.length
No
Yes
Prompt user for a value
us[i] = user input
i = i+1
Create a second array "s" of size Size
i=1
i<us.length
i = i+1
No
Print all elements of the sorted array "s"
Yes
j=1
No
j = j+1
j<i
Yes
At the end of the sorted array so i = j
Yes
s[j] = us[i]
No
An element is out of order thus us[i] < s[j]
Shift all elements start at j to the right by 1
s[j] = us[i]

Implementation Problems Encountered

Using the incorrect index for unsorted array and the sorted array was a major problem. It was difficult to keep track of one to the other until I went back and renamed the arrays and the variables using a more descriptive but much longer identifier.

Index out of bounds exceptions came up as I accidentally was using <= the array's length for loop's Boolean expression instead of <. I've made a note that the last valid index in an array is the length-1.

Another index out of bound exception arose when I forgot to check to make sure the entered size of the array was non-negative. That was fixed by putting an if-statement that halted the program when that occurred.

Lab Report Questions

1. The flow chart is found in the Solution section.
2. When checking for elements that are out of order, we may simply switch the sign from "<" to ">", and that will sort in descending order.
3. Insertion sort demonstration using the given array:

Index	0	1	2	3	4
Value	5	4	3	2	1

Create New, Empty Array

Index	0	1	2	3	4
Value					

Insert 5

Index	0	1	2	3	4
Value	5				

Insert 4

Index	0	1	2	3	4
Value	4	5			

Insert 3

Index	0	1	2	3	4
Value	3	4	5		

Insert 2

Index	0	1	2	3	4
Value	2	3	4	5	

Insert 1

Index	0	1	2	3	4
Value	1	2	3	4	5

4. Insertion sort can be done using only one array. However, to get the desired results the first change is when the array is populated it must immediately be printed out.