```
/*----------------------------------------------------------------------
Programming Assignment 2 Submission
Stacks -- Use two stacks to display numbers in descending order
          by sorting the numbers from stack 1 into stack 2 in
          ascending order.
Created by Hyo Lee
Student ID: 002292770
01/19/2016
-----------------------------------------------------------------------*/
```

Part a)

Problem:

Design an algorithm to sort a group of numbers in ascending order using two stacks, and display the numbers in descending order separated by commas.

Solution description:

Stack 1 will contain our initial unsorted numbers. We can initially move the top value in stack 1 to stack 2 since stack 2 is initially empty the first element in stack 2 will be the top. We can then compare the top value of stack 1 with the top value of stack 2. If the top value of stack 1 is larger than the top value of stack 2, we can simply “move” the top element of stack 1 onto stack 2, however, if the top value of stack 2 is larger than the top value of stack 1, we cannot move the top element of stack 2 back to stack 1 without removing the top of stack 1 first. In order to “remove” the top element from stack 1, we can store the top value in a temp variable, then remove the top element from stack 1. We can then use the temp variable instead of the top value of stack 1 to compare to the top value of stack 2. Now if the top value of stack 2 is greater than the temp value, we can move each top element from stack 2 back to stack 1 until the temp variable becomes larger than the top value in stack 2. The temp variable can then be moved on top of stack 2. If we continue these loops for each element in both stacks, stack 1 should become empty and stack 2 should contain all the initial numbers sorted in ascending order.

Solution steps:

Step 1: Loop through stack 1 while it is not empty.

Step 2: While step 1 condition remains true, store the top value of stack 1 in a temp variable, then pop off the top element in stack 1 (now stored in the temp variable).

Step 3: Continue looping through stack 1 and possibly loop through stack 2.

Step 4: Loop through stack 2 while stack 2 is not empty, and while the top value of stack 2 is greater than the temp variable.

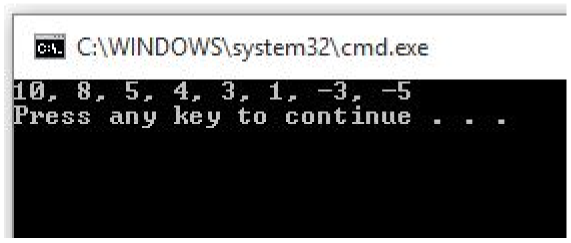
Step 5: While step 4 condition remains true, push the top value of stack 2 onto the top of stack 1, then pop off the top element in stack 2.

Step 6: While step 1 condition remains true and step 4 condition remains false, push the temp value onto the top of stack 2.

Step 7: All that remains is to display the contents of stack 2. Loop through stack 2 and print out the top value while popping off the top element until the stack is empty.

Sample input 1: [1, 5, 3, -3, 4, 8, 10, -5]

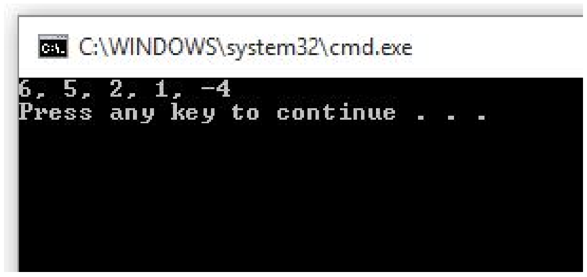
Screenshot shows the program correctly displays: 10, 8, 5, 4, 3, 1, -3, -5



```
10, 8, 5, 4, 3, 1, -3, -5
Press any key to continue . . .
```

Sample input 2: [1, 5, -4, 6, 2]

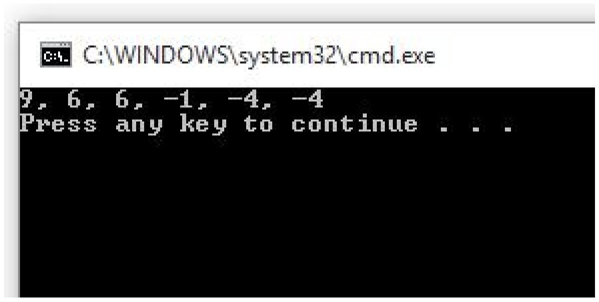
Screenshot shows the program correctly displays: 6, 5, 2, 1, -4



```
6, 5, 2, 1, -4
Press any key to continue . . .
```

Sample input 3: [-1, -4, -4, 6, 6, 9]

Screenshot shows the program correctly displays: 9, 6, 6, -1, -4, -4



```
9, 6, 6, -1, -4, -4
Press any key to continue . . .
```