Due Date: Monday, April 6 at 7:00 PM

Turnin:

Submit all files to the cs352s15assg5 turnin directory. Recall that this command looks like

```
turnin cs352s15assg5 <list of files>
```

You may list the files individually, or, if they are the only .c files in your directory, you may just use the ‘*.c’ as the second argument.

You may check what files have been submitted by typing "turnin -ls cs352s15assg5".

Your scripts will be put through a series of test cases and compared to ours using diff. This means that things like spelling, punctuation and whitespace are very important if you would like to pass all the test cases without needing to appeal. Additionally, your filenames will need to match exactly in order for the script to pick it up. If you have any questions about the output of your scripts or the format of the filenames, please stop by our office hours, post on Piazza or email us.

Remember, no late work will be accepted.

The following rules apply to all programs:

1. **Do not use anything covered after chapter 17 in the King textbook.**
2. Each program should be a single C file.
3. Do not #include anything beyond <string.h>, <stdio.h> and <stdlib.h>
1. **postfix arithmetic (postfix2.c)**
   Please write a program that will evaluate a postfix notation arithmetic expression(s) on integer values. Do everything required of the postfix problem in Assignment #3, with the following changes:

   1. Do not use arrays. Use dynamically allocated structures like a linked list to implement the evaluation stack.
   2. Do not assume any maximum depth of the evaluation stack.

2. **Ascending numbers (ascending2.c)**
   Please write a program that will find the longest ascending sequence of integers in a sequence of values read from stdin. Do everything required of the ascending program in Assignment #3, with the following changes:

   1. Do not use arrays. Use dynamically allocated structures like a linked list to implement the evaluation stack.
   2. Assume input numbers are non-negative, so they can include zero.
   3. Correct execution should exit with status 0 after printing the results; incorrect input should result in immediate exit with status 1;
   4. Do not assume any maximum length of the input.

3. **Sorted Linked List (sorted.c)**
   Please write a program that will read in a list of non-negative integer and keep them in sorted order in a linked list.

   1. Your program will read an arbitrarily long list of non-negative integers from stdin, separated by whitespace.
   2. For each number, do the following
      a. If the number is greater than zero, insert it into a linked list in such a way that the numbers are in ascending sorted order.
      b. If the number is zero, do only the following two things:
         i. print out the entire list (including duplicates) in ascending order, with one number per line and no extra whitespace.
         ii. After printing out the list, remove all duplicates from the list.
   3. When the program reaches the end of input, print out the length of the list (including duplicates) on a line, followed by a newline character.
   4. On any error condition, exit with status 1.
   5. On normal completion, exit with status 0.
   6. Do not use arrays. This should be done with a dynamically allocated linked list.