Practice Exercise 2: SLL Functions

To learn how to manipulate linked list, you will need to practice. Use the project associated with the lecture slides for singly-linked list and create new member functions (see examples below). Do **not** add any additional member variables, but do create new testing cases.

These are some ideas discussed in class:

- Member function **insertBack**
  - **Parameter:** An integer to store in the new node Traverse the list and insert a new node at the end of the list.

- Function **deleteFirstNode**
  - Delete the first node in the list.
  - You should print the list to check that the list is still connected to the object of the AnyList class.

- Member function **deleteLastNode**
  - Traverse the list and delete the last node in the list.
  - Make sure that the new last node (the one before the node you just deleted) is pointing to NULL.

- Member function **search**
  - **Parameter:** An integer storing the data to be searched.
  - Searches data in the list.
  - Traverse the list using a WHILE loop; stop when you find the data you were looking for.
  - If you do not find the data, output an error message.

- Member function **min (or max)**
  - Traverse the list to find the minimum (or maximum) value and return it.

- Member function **replaceData**
  - **Parameter:** An interger that stores the to be replaced and an integer that stores the data that will replace it.
  - Using a WHILE loop, traverse the list and replace the data when found; make sure you stop once you have found the data.

- Member function **commonEnds**
  - http://codingbat.com/prob/p191991
  - CodingBat is a great source for ideas. Check Array-1, Array-2, and Array-3 problems and modify them to fit linked list. Use their testing cases to verify that your function is working as expected.