This assignment has the same functionality, and same inputs and outputs as assignment 1. It will also be run from the command line with

cat input.txt | java -jar A2.jar > output.txt

# Implementation Details

* Create a class called A2. It will have the main() method and do the bulk of the processing. Be sure to set it up to instantiate an A2 object.
* Input is all from standard input. Create a Scanner object and use that for all input:

Scanner inp = new Scanner(System.in);

* All output should be to standard output. Use System.out.print(); and System.out.println(); for all output.
* You have a Food class from assignment 1 to represent a food. You should not need to alter it.
* You should create a linked list class called SLL. It will have member data:
  + Node<T> head, tail;
  + int size;
* You can use the Node class given out in lab as a starting point. If you create methods in SLL that match the ones in ArrayList, you can use much of your existing code with little editing. Note that your SLL should allow you to add Foods in an order specified by you (based on the Comparable and Comparator interfaces).
* Your Node class uses a generic type for the data that it stores. However, it should not just be any old class. You want to be able to order your list, so you need a type that implements Comparable. That is done like this: public class Node<T extends Comparable<T>> and public class SLL<T extends Comparable<T>> This will restrict T to be only classes that implement Comparable. The easiest way to do this assignment is to simply replace the ArrayList<Food> of Foods with a SLL< Food> object.
* Initially you should only worry about building a list in alphabetical ordering.
* You will need to create different sorted lists to display them. Use these lists to print out your results exactly as in Assignment 1. You will not be able to use the Collections.sort method, as your SLL will not implement the Collections interface.

# Testing and example files.

* Use the test files provided in Assignment 1. Use these to determine if your program is running correctly.
* You should use the diff command to compare your output to the samples.

**diff (cat sampleout.txt) (cat myout.txt)** If diff gives no output, your program is working correctly.

* You should also devise a test plan and create a series of test files to fully exercise your program. I will be evaluating the program against files you have not seen yet that will be meant to test your code.
* This is a fairly short assignment. You really only need four classes, A2. Food, Node and SLL. Focus on writing compact, efficient code.

# Documentation and Coding Standards

* Your program should follow all of the coding rules and guidelines outlined in the document provided.
* In particular, include Javadoc style comments for all classes and substantial methods.
* Look at GoodDocs.java for an example

# What to Hand In

* a single file, A2.jar. Submit this to the Blackboard drop box provided. The jar should be executable and contain:
* All of your .class and .java files.
* A class called A2 which has a main() method.

# Grading

A detailed grading rubric will be provided. I will run your program with the command line given above on three text files and compare your output to the specifications.

# Outcomes

Once you have completed this assignment you will have:

* Implemented an ordered linked list;
* Used the Java Comparable interface;
* Used a Comparator object;
* Used standard input and standard output re-direction.