Salesperson FAQ

Do I need to follow the prescribed API?
Yes, you may however add any private helper methods you'd like.

What should my program do if the tour has 0 points?
Panic. When done, the `size` method should return 0; the `distance` method should return 0.0; the `show` method should write nothing to standard output; the `draw` method should draw nothing to standard draw.

How long should my programs take to execute?
It should take less than a minute for the 13,509 city example (substantially less if you have a fast computer). If your code takes much longer, try to discover why (think Big-O, analysis of algorithms).

How can I produce an animation of each heuristic in action?
It's easy (and instructive): just redraw the tour after each insertion. It could take a while on a big input file, so you might want to modify it so that it redraws only after every 20 insertions or so.

Omg! What is the file `Tour$Node.class`?!
When you declare a nested class like Node, the Java compiler uses the `$` symbol to mark its name.

Why do I get a NullPointerException?
You are attempting to access a field (or method) using a variable that stores a `null` reference. Linked list algorithms are difficult to debug, there's no way around this. If you have a problem, test with small inputs (a tour with only a handful of points), and log what's happening to the console.

When should I create a linked list node with the keyword `new`?
To create a tour with *N* points, you should use `new` exactly *N* times with Node, once per invocation of whichever insertion method you are using.

Can I use Java's built in LinkedList class?
No, you lazy! One of the main goals of this assignment is to gain experience writing and using linked structures. The Java libraries can only take you so far, and you will eventually discover applications which cannot be solved without devising your own data structures.

What is the difference between the nearest neighbor and smallest increase heuristics?
The lab doc has pretty good descriptions of these methodologies, but put another way:

- *Nearest neighbor:* simply scans the tour and inserts the supplied point after its nearest neighbor.
- *Smallest increase:* insert the current point into the tour at the location (index) which results in the smallest overall tour distance.