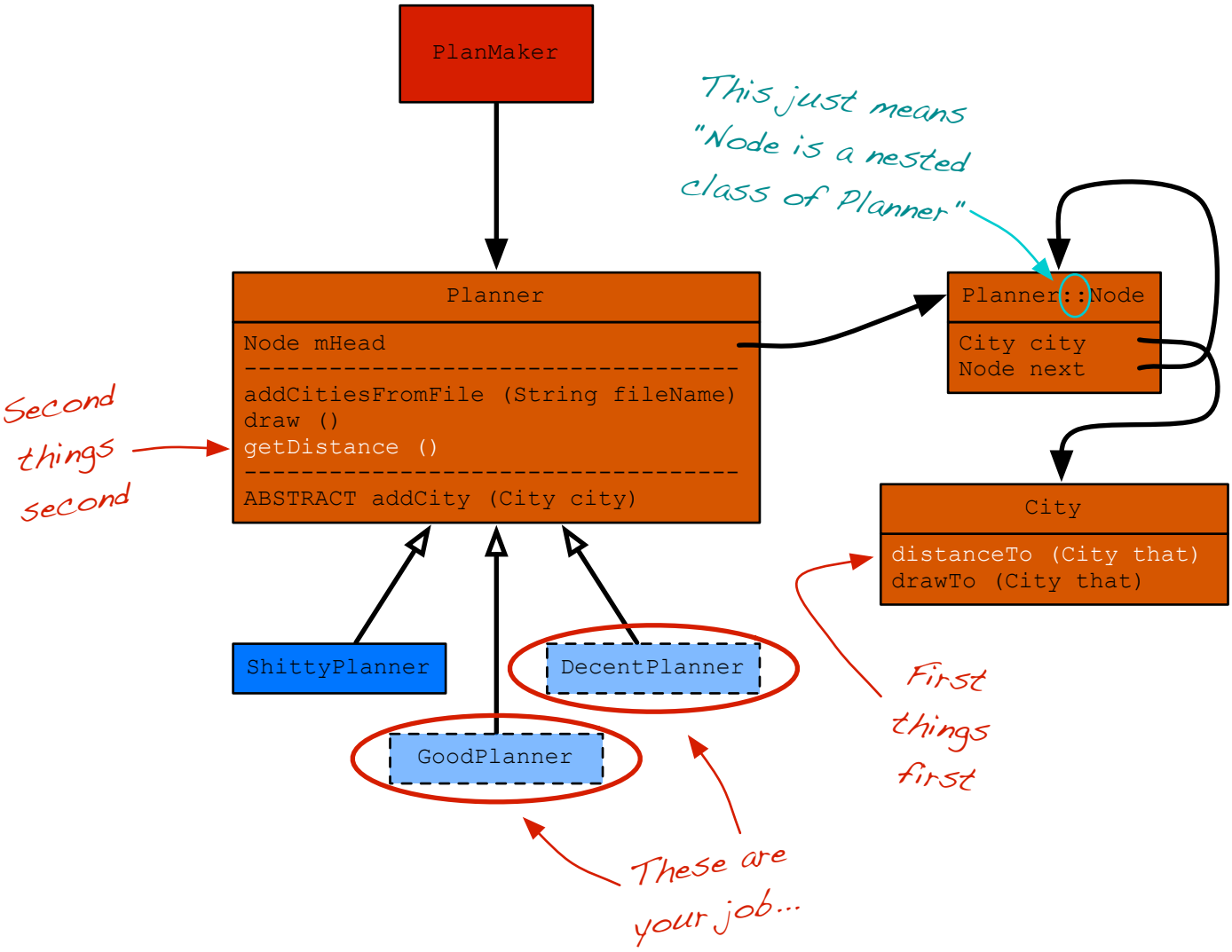
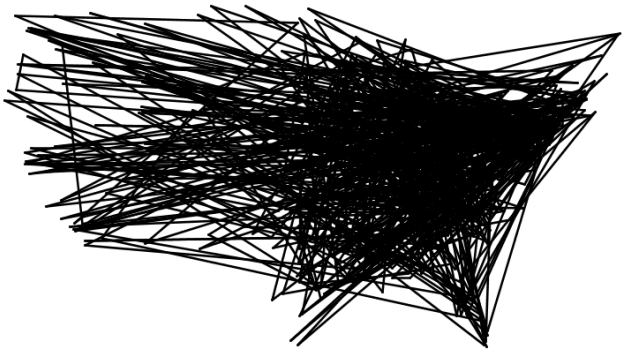
The Traveling Penman



It's been two years. You thought they were finished with you. You thought wrong. It turns out *Pen-Fifteen*'s vast network of anonymous agents is difficult to maintain. And logistics are starting to get in the way of The Grand Plan. They need your help.

To maintain utmost secrecy, *Pen-Fifteen* uses only one disseminating agent per country. These agents must visit thousands of cities to hand-deliver assignments to individual agents such as yourself. Thousands of cities. In alphabetical order. Which looks something like this:



It's up to you to devise a better method.
Preferably before next operation stage.
Which is in a week.

To: Agent X
From: Pen-Fifteen
Subject: Long-Term Planning

As before, start at the beginning: PlanMaker.java. The main() method works much like in the previous assignment, but go through it quickly to make sure you understand each line. You will have to modify it slightly to add your own Planners.

Now familiarize yourself with the *Planner* class. It is essentially a linked list of cities which leaves the actual order of insertion up to the individual subclasses. The only subclass that exists right now is the *ShittyPlanner* class, but you will be writing two more yourself: the creatively named *DecentPlanner* and *GoodPlanner*.

Once you're ready, you must first fix a few pending issue we had with the code. We don't actually know how far agents have to travel because we couldn't figure out how to calculate distance, but we hear good things about a certain "Pythagorean Formula." You should use this to write the City::distanceTo() method, and then use that to write the Planner::getDistance() method. Hint: if the latter is confusing, Planner::draw() has a surprisingly similar structure.

Now for your real job. Create a new class *DecentPlanner* that uses the following algorithm to add cities to the list:

1. To insert City X, loop through the list to find the Node storing the nearest City.
2. Insert City X into the list following the Node you just found.

But wait, there's an even better algorithm! Try this in a new class *GoodPlanner*.

1. To insert City X, loop through each neighboring Cities Y and Z.
2. Calculate the distance that would be added by visiting X in between Y and Z.
3. Insert City X in between the two cities which had the minimum distance added.

That's it! Simple, easy, downright trivial!
It must be ready by next a week.
There is no "or else."

Grading

Algorithms

Distance method	(5)
DecentPlanner	(5)
GoodPlanner	(5)

Code Quality

Neatness, consistency, readability	(5)

Extra Credit

Create your own Planner algorithm that outperforms GoodPlanner	(5)

Super Duper IMPORTANT Things

Talk through the following with your partner and ask me about about any you don't understand. These aren't graded because that would be annoying for all involved, but not being able to answer any of them indicates some conceptual misunderstanding which is in your (and your family and grade's) best interest to resolve.

1. If ShittyPlanner truly is visiting cities alphabetically, in what order must they appear in the file? Hint: it's not alphabetical! This is a little tricky, look at the code and try drawing out the linked list for the first few insertions.

2. Where would the StdDraw class fit into the class diagram on the first page?

3. Right now, addCity() is O(n) for both DecentPlanner and GoodPlanner. If you were to use an array instead of a linked list, how would the efficiency change?