By Hsuan-Chu Tang

Design

The program consists of 5 classes. The FileIO class handles the map file input, while the Bulldozer class represents the behavior of a bulldozer, and the Site class stores the state of the site. The Simulator class takes in user input and decides what method to invoke from the Bulldozer and Site class to reflect the result of the command. Finally, there is the Main class. The main method inside Main creates an instance of the Simulator for each new simulation and calls on the simulator's methods that pieces together the program.

There is no direct interaction between the classes Bulldozer and Site. According to the user input, the Site prepares a valid route, which is passed to Bulldozer through the Simulator. The Bulldozer processes it, updates its internal state, and returns back a *cleared* route, which is passed back to the Site through the Simulator for the map to be updated.

Approach

After making a list of the desired behavior and constraints of the program, I observed three main things that can be abstracted: the bulldozer, the site, and the simulator itself. I decided to start with the most simple class, the Bulldozer, which has only very simple behaviors such as turn and advance. Implementation was done following TDD philosophy. Next I worked on the Site class, but soon found by failing the test that I have to add an FileIO class for the Site to have a map to base on. After the basic components were completed, I could start to try to assemble them into the Simulator class.

During implementation, I first tried to let Site determine what types of terrain the Bulldozer will pass. However when I started to write the Simulator class I found this approach difficult to decouple the Site and Bulldozer class, and conceptually it does not feel right that the Bulldozer is not the one that observes which terrain it is on or passing through. Therefore in the end I developed a solution which the Site checks whether there is a protected tree in the user proposed route or if the bulldozer will run off site and returns a validated route. The Bulldozer, or any other type of vehicle that may be added in the future, does not care about whether the route is valid or not. It simply drives the route and updates its fuel consumption and other states according to the type of terrain it crosses.