Matt Stevenson

Vapor Trail MTB Race Assistant

I was a little ambitious in the outset of this project, which is why the data classes (the first section of code) attributes are a little overbuilt. However, I was indeed able to execute the “get to safety” function to spit out the various routes off a specific section of trail. Likewise, the “track your position” function and concurrent ability to update the difficulty of the trail using an @property were small victories. The prompts are pretty straight forward, and the error handling is good enough to guide you when you break it.

Please try to break it. I could not find any error cases that were not handled, though I did raise one exception that will terminate the program. I worked hard to try not to raise exceptions, but could not find a way around setting the over-time exception into the classes Rider and Safety, which meant that they would not spit the user back to the top of the prompt, but instead exit.

I had fun doing this project, though would have done a few things differently knowing what I do now about classes, exceptions, and feeding data from input.

The architecture I used to tie the data classes together (each section has an end, start and bail trail that refer to other instances of classes) was nifty, but I would want to rethink this if I were going to make this database more searchable. However, instead of having to use weakrefs, gc, etc., I just put a sequential list (sections) to map out the flow of the race. Likewise, it is hard to model a map with incomplete information using just a few classes. There are just too many variables, and I would not want to make this too complex if this were a format for modeling any type of ultra-endurance race/back-country trail. Designing in sufficient flexibility is surely a skill that needs to be developed with practice.

One frustrating thing was that I was unable to get parent classes to work for methods other than \_\_init\_\_ . Having the ability to do so would have made my code much more succinct and elegant, and I could have even put some of the scripting (the last ¼ of the code) into some classes, as the prompts were repetitive for error handling, etc. Similarly, I had intended for the Rider class to be a parent of class Safety, as many of the calculations were similar, but slightly different enough to separate them. However, if I were to rewrite it, I would make a parent class for both with similar calcs and init methods.

I think the main thing I conceptualized is how the complexity for a more rigorous mapping program gets intense quickly. In order to do route setting using GPS routes, there must be a ton of processing power to handle all the nodes along a route and count up the length. Kudos to Google Maps.