Trevor Krenz
Lab 9

1. [].class returns Array
   3/2 returns 1
   3.0/2.0 returns 1.5
   [].nil? returns false
   def h; “Hello world”; end returns nil
2. Having everything as an object. It makes it easier for code to work with since you don’t have to worry over special cases. For example, in Java I would have to worry about treating primitive types and objects differently, but not in Ruby.
3. I think the advantages of BDD are worth the drawbacks. BDD is centered on the user, which determines the success of the product. Additionally, BDD has a goal in mind, while TDD does not necessarily have one besides testing driving all development.
4. I think it helps developers, generally. It enhances communication to the client about what the developers are doing, and can give an indication of what is working and what is not. The feature descriptions describe the requirements better than just tests, so having the feature descriptions help fix requirements errors, which are the most expensive to fix.
5. Having more verbose, complex English helps communicate ideas to those who aren’t programmers, and communicates the point directly rather than getting lost in abstractions by the language. The drawback is that it can be ambiguous since natural language can be interpreted in many ways.