Devin Banks

Lab 9

Q1: [].class => Array

3/2 => 1

3.0/2.0 => 1.5

[].nil? => false

def h; “Hello world”; end => nil

Q2: My personal favorite feature of Ruby is that everything is an object. This makes it really easy to reuse code because everything could be interpreted in similar ways.

Q3: The advantages of BDD are definitely worth the drawbacks. Even though BDD adds another layer of complication and possible bugs,

Q4: I think BDD in certain cases can be incredibly useful in the development process. It can be used most easily in the cases where it may be difficult to produce formal test cases for a system. In this case, the scenarios may be much easier to write. However, in other cases, using BDD can hinder developers. If the system truly does not require the amount of depth or type of testing used in BDD, simple unit tests may be sufficient and easier to write.

Q5: Using more verbose, complex English can help in feature definitions by more easily clarifying what the system needs to do in order to pass tests. By clarifying the tests as much as possible, it can help to more easily write the code necessary to pass the tests. The main drawback is that using verbose, complex English can make the system seem even muddier than it was to begin with. Using complex English in vague and superfluous ways can make it incredibly difficult to decipher what actually needs to happen.