In the second project, I did not have as many struggles as I thought I was going to after looking at the spec. The most notable issue I came across was recognizing no input or the “empty string” for the destination so that I could write an error message. A quick google search revealed I could test for an empty string simply by saying, “if (destination == "")” meaning two quotation marks would notate a lack of any input. The spec however noted that the program could recognize a space as a valid input, so my program does not deal with spaces as a destination. I also struggled at first with the layout of the program due to its seemingly complex nature, but this issue resolved itself very easily when I started to write the first few lines of code which quickly turned into the first if statement and finally the rest of the outputs. The last thing I almost forgot to do was put in the lines of code on page 32 of the textbook that sets all output values to a precision of 2 places after the decimal that I only caught after extensive testing.

Test Data

* negative age (verifies my “The age must not be negative” works)
* Not y or s for student (verifies my “You must enter y or n” works)
* No destination (verifies my “You must enter a destination” works)
* Negative zones crossed (verifies my “The number of zone boundaries crossed must not be negative” works)
* Various combination to make sure the first erroneous input is the error message displayed (confirms that no matter what improper inputs I apply in any order will write the error message for the first improper input and not any of the following)
* The following will make sure student status does not affect a minor as well as if the program can correctly differentiate between a short trip discount and a standard fare trip for a minor
  + Minor, student, 1 boundary
  + Minor, student, 5 boundaries
  + Minor, non-student, 5 boundaries
  + Minor, non-student, 1 boundary
  + Minor, student, 0 boundaries
  + Minor, non-student, 0 boundaries
* The following will test if the program correctly identifies an 18-year-old as an adult, if it can recognize a short trip discount for adults, and if it can correctly apply that discount only when the adult is a student
  + 18 y/o, student, 1 boundary
  + 18 y/o student, 0 boundaries
  + 18 y/o non-student, 1 boundary
  + 18 y/o non-student, 5 boundaries
  + 18 y/o student, 5 boundaries
* The following will test if the program correctly identifies a 65-year-old as a senior, if the 0 zone discount is recognized, if that discount is applied correctly over the student discount only in the case of 0 zones (which is $0.20 more), and applies the standard senior fare if no other discount is present
  + 65 y/o non-student, 0 boundaries
  + 65 y/o non-student, 1 boundary
  + 65 y/o non-student, 12 boundaries
  + 65 y/o student, 0 boundaries
  + 65 y/o student, 1 boundary
  + 65 y/o student, 12 boundaries
* The following will check a few random cases that should not trigger any discounts (there are many cases above that also should not trigger any discount)
  + 25 y/o non-student, 12 boundaries
  + 75 y/o non-student, 12 boundaries