Matthew Staehely

CSC 143 Offenback

1. I came up with my design by using the matrix test file we were given to get a good abstraction of just what I expected my algorithm to do. Once I got a handle on how it would examine each element of a single row, I could construct a recursive method which would examine one row at a time while cross checking to make sure that a particular value was one which had not already been selected at a previous node. Thanks to the benefits of local scope, the String I used to track selections never went back ‘up’ the stack, which helped keep track of which choices were ‘unmade’. I didn’t turn to outside material beyond the book and class slides, as I felt that would have given away too much information and defeated the purpose of the assignment.
2. I tested my code in two ways- first, the method which made matrices needed to be checked to make sure the matrix it produced matched what was expected. The commas made things a little difficult, and the method will only work for comma-delineated text files. A little too specific, but hopefully this is to spec. I tested the path method by passing valid and invalid matrices to it, and seeing where exceptions were or were not caught and attempting to troubleshoot those. One way I am concerned my program may not be to spec: I only thought of this after submission but I realized that if there is a negative integer in the matrix and my recursive method does not get to that particular element, then it won’t detect that an exception should be thrown- for instance, on the 7x7 matrix, if one of the values on the last row is negative but it never needs to get to the last row to complete its path, then it isn’t going to see the negative value there. How should this be handled? One way would be to have the method check the entire matrix for negative values in any element, but since this is a recursive function that seems like it would be defeating the purpose of using recursion in the first place. Another thing which came to mind would be to have a boolean for having checked the matrix, and then using a helper method, having the recursion catch the exception thrown by that helper and throwing it again. The boolean would act as a gatekeeper for the method- once it’s been set to ‘true’, the helper method would never be called again. Seems a little clunky, though.
3. I think this was a great project. I am not one of those people to whom recursion comes naturally and have had to struggle a good deal with the Practice-it exercises, and have spent a lot of time looking at other peoples’ code to try and figure out how it is recursion does what it does, as well as working with more of the problems available to us. I hope we get to work with recursion more because my mastery of it is not where I would like it to be.