For simplicity sakes the combinational test were done to cover possible inputs and only focused on output flow not so much as content. Also, let me note that the difference between a properly read string file and an invalid string file is that both tm's are loaded but the one with an invalid string file will have an empty string file list.

<r1,c1,-> The combinational test here tested the exit command and it was performed on a vm with the appropriate results. The TM was loaded properly. On the command prompt I entered x which gave us the expected action which in this case was a complete exit of the program. Since the input is of no importance there is no reason to include it in the combinational test.

<r1,c2,i1> This is an impossible scenario considering if the command has a prompt generated than it is impossible for input to not be needed.

<r1,c2,i2> This combinational test was produced with the insert command. After being prompted for a command I entered ‘I’ and was then prompted for string to insert to string list. Here I entered ab and then pressed enter. One thing that was not included in the combinational test was the confirmation from some commands such as the Insert Command. Just as invalid input on prompt generating commands produces an error display so do valid inputs on prompt commands. In this case the message read, “ab successfully entered into the input string list”. I was then returned to the command prompt. Everything for the insert command performed as expected in the actions model.

<r2,c2,i3> This combinational test was produced with the insert command. I entered an ‘I’ in the command prompt. Then I was prompted for input upon which I entered a string of digits, specifically “1234”. The proper error message was displayed and then I was returned to the command prompt.

<r2,c3,i1> This combinational test was produced with the View Command. The view command was activated by pressing V on the command prompt. This was after the TM was successfully loaded. Immediately after the Definition of the TM was displayed. This is the correct output for the View Command.

Q: {s0, s1, s2, s3, s4}

Sigma: {a, b}

Gamma: {a, b, X, Y, -}

Delta:

(s0, a) = (s1, X, R)

(s0, Y) = (s3, Y, R)

(s1, a) = (s1, a, R)

(s1, b) = (s2, Y, L)

(s1, Y) = (s1, Y, R)

(s2, a) = (s2, a, L)

(s2, X) = (s0, X, R)

(s2, Y) = (s2, Y, L)

(s3, Y) = (s3, Y, R)

(s3, -) = (s4, -, R)

q0 = s0

B = -

F = {s4}

<r1,c3,i2> This command was deemed impossible because either input is not needed so either valid or invalid input is an impossible scenario.

<r1,c3,i3> This command was deemed impossible because either input is not needed so either valid or invalid input is an impossible scenario.

\*\*\*\*\* All the Combinational Models for the Invalid Input String has no effect on our test exempt for the fact that the list of input strings is empty or not. Again our Test Models will not determine this exception.

<r2,-.-> To make sure that the input string remained empty when an inproper string file was loaded I took our initial proper string file and left it empty. Then I ran the program with the improper string file. To test if it was empty, I immediately ran the List Command on the TM and was displayed the expected message, “No strings in the input string list”. Therefore, there is no other reason to test the remaining commands under r2.

<r3, -, -> This model was tested by going back into the .def and deleting some characters in the Keywords. TAPE\_ALPHABET was changed to TAPE\_PHABET. After trying to execute tm I was displayed an error message, which was expected. The program did not exit though completely. It required me to press enter again to completely exit the program.