Software Test Plan

1.0 Test Plan Identifier

Turing Machine (TM) in C# 1.0.

1.11 Team 2 (Lansdon Page, Jason Wong, Ryan Wilson, Jason Stidham)

2.0 Test Items / Fault Model

Our fault model includes the following items believed to be high risk in the operation of the Turing Machine in C# 1.0

2.11 When executing the TM the TM definition file is read into the application it must be received by the parsing algorithm in a specific order of keywords to be a valid TM. See 4.2.0.4d of the Requirements Documentation for the order of the keywords. Loading the definition file is critical to the operation of the Turing Machine.

2.12 The Parsing Algorithm checks for validity of each data set related to keywords. Need to check that the algorithm only accepts valid input for each category, as referenced in the Requirements Document section 4.2.0. Parsing the definition properly is critical to the operation of the turring machine and due to the nature of parsing, it's a high risk area.

2.13 Sections of TM that requires users input. User input is a critical area to the functioning of the application and must function properly. Below outlines all scenarios in which a user interfaces the application, commands that require additional input to function will be labeled sub-commands:

- Help (h)
- Show (w)
- View (v)
- List (l)
- Insert (i)
    - Sub-command (string)
- Delete (d)
    - Sub-command (integer)
- Set (e)
    - Sub-command (integer)
- Truncate (t)
    - Sub-command (integer)
- Run (r)
    - Sub-command (integer)
- Quit (q)
- Exit (x)

3.0 Features to be Tested

3.11 The method of reading a text upon program execution.
3.12 Ensure that each section of the text file is being handled properly by the parsing algorithm.
3.13 Test user controlled actions, all menu commands and their subcommands.

4.0 Approach

For test 3.11, reading and importing data from the TM definition text file, we will use a test driver that will check the validity of all keywords discovered defined by the Requirements Document section 4.2.0.4d. The number of tests i.e. different definition files will be used to ensure that errors are discovered at each keyword if an error in formatting is discovered. The focus of these tests will be the loading of a file and the integration of all the classes that participate in extracting portions of the file. This chain of events and logic must be tested to make sure there are no errors in how the separate components interact.

For these tests we will use a valid file, as well as several invalid files:

- Valid File
- No States
- No Input Alphabet
- No Tape Alphabet
- No Transition Function
- No Initial State
- No Blank Char
- No Final States

For test 3.12, parsing and data handling we will use a test driver that will check that data is being handled correctly by the associated class objects. Sample input will be defined for each test using both valid and invalid values. The tests will stress the areas of parsing that should succeed as well as values that should cause errors according to the requirements documentation.

4.0 Approach (continued)

The following areas were identified as critical areas for testing:

4.2.0 Description	
4.2.0.5 - Any text before the keyword “States:” is considered a description of the tm.def file	a. Text before the keyword “States:” may not contain any keywords.
4.2.0.6 - The tm.def file uses white space as its delimiter.	a. White spaces include the following: i. Spaces ii. Tabs iii. Newlines

4.2.1 States	
4.2.1.2	There must be at least one state.
4.2.1.4 - The name of the state...	a. must be unique from other states.
	b. is case sensitive.
	c. must only use alphanumeric characters or the underscore character

4.2.2 Input Alphabet	
4.2.2.2	Each element consists of a single character
4.2.2.3	Duplicate characters aren’t allowed

4.2.4 Transition Function	
4.2.4.1 - Each transition function must contain 5 elements delimited, and in the following order:	a. States element b. Tape_Alphabet element c. States element d. Tape_Alphabet element
4.2.4.5	Transition functions must use characters from t Tape Alphabet

4.0 Approach (continued)

4.2.5 Initial_State	
4.2.5.1	There can only be one state.

4.2.6 Blank_Character	
4.2.6.1	Must be an element of the Tape Alphabet Γ

4.2.7 Final State:	
4.2.7.1	Must have one or more final states
4.2.7.2	The state must be an element of the set of states Q

For test 3.13 input commands, we will test using a black-box approach testing all menu functions. Combinations of input will be used to trigger error conditions as well as testing the successful processing of each command. Every command that has a sub-command option will also be tested for valid and invalid handling of input.

5.0 Test Cases

5.1: TM Definition File Validity

Unique Test Identifier	5.1.1 Parse definition - check parse correctly
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and loads corectly.
Component	Function Name (TestDefinitionValid())
Input Condition	Input = valid text document formatted to Requirements specification
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords in the correct order required for a valid TM application.

Unique Test Identifier	5.1.2 Parse definition - check no state
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and fails on no states being found.
Component	Function Name (TestDefinitionNoState())
Input Condition	Input = invalid text document with error in the STATES keyword
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords discovering error condition in the STATES keyword resulting in invalid message.

Unique Test Identifier	5.1.3 Parse definition - check no input
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and fails with INPUT_ALPHABET keyword
Component	Function Name (TestDefinitionNoInput())
Input Condition	Input = invalid text document with error in the INPUT_ALPHABET keyword
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords discovering error condition in the INPUT_ALPHABET keyword resulting in invalid message.

Unique Test Identifier	5.1.4 Parse definition - check no tape
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and fails with TAPE_ALPHABET keyword.
Component	Function Name (TestDefinitionNoTape())
Input Condition	Input = invalid text document with error in the TRANSITION_FUNCTION keyword
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords discovering error condition in the TAPE_ALPHABET keyword resulting in invalid message.

5.1: TM Definition File Validity (continued)

Unique Test Identifier	5.1.5 Parse definition - check no Transitions
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and fails with TRANSITION_FUNCTION keyword
Component	Function Name (TestDefinitionNoTransiton())
Input Condition	Input = invalid text document with error in the TRANSITION_FUNCTION keyword
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords discovering error condition in the TRANSITION_FUNCTION keyword resulting in invalid message.

Unique Test Identifier	5.1.6 Parse definition - check no initial
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and fails with INITIAL_STATE keyword
Component	Function Name (TestDefinitionNoInitial())
Input Condition	Input = invalid text document with error in the INITIAL_STATE keyword
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords discovering error condition in the INITIAL_STATE keyword resulting in invalid message.

Unique Test Identifier	5.1.7 Parse definition - check no blank
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and fails with BLANK_CHARACTER keyword
Component	Function Name (TestDefinitionNoBlank())
Input Condition	Input = invalid text document with error in the BLANK_CHARECTER keyword
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords discovering error condition in the BLANK_CHARECTER keyword resulting in invalid message.

Unique Test Identifier	5.1.8 Parse definition - check no final
Test Description	This test accomplishes verifying the TM definition file is able to be opened and read by the Application and fails with FINAL_STATES keyword.
Component	Function Name (TestDefinitionNoFinal())
Input Condition	Input = invalid text document with error in the FINAL_STATES keyword
Input State	Values for STATES, INPUT_ALPHABET, TRANSITION_FUNCTION, INITIAL_STATE, BLANK_CHARECTER, and FINAL_STATES contain no data.
Expected Result	TM application will open txt file and read keywords discovering error condition in the FINAL_STATES keyword resulting in invalid message.

5.2 TM Parsing Algorithm

Test ID	5.2.1 - ParseDefinition_CheckForDuplicateStates
Test Description	Verify Conformance of Requirement 4.2.1.4a of Requirements Document that State names must be unique. This is accomplished by feeding a definition file with two duplicate states to the load method of the States Class, and verifying a false boolean is returned.
Component	Type: Class method Name: States.load() bool States.load(List<string> definition_file)
Input Condition	List<string> invalid_definition = {“S1 S1”, “INPUT_ALPHABET:”} States test_state = new States()
Input State	Turing Machine program has just finished processing both the Turing Machine description and the keyword “STATES:”
Expected Results	A false boolean should be returned from the load class method of the States class.

Test ID	5.2.2 - ParseDefinition_CheckCaseSensitivityForStates
Test Description	Verify Conformance of Requirement 4.2.1.4b of Requirements Document that State names must be case sensitive. This is accomplished by feeding a definition file with two similarly named, but differingly cased states to the load method of the States class, and verifying that case is maintained by States class.
Component	Type: Class method Name: States.load() bool States.load(List<string> definition_file)
Input Condition	List<string> invalid_definition = {“state1 STATE1”, “INPUT_ALPHABET:”} States test_state = new States()
Input State	Turing Machine program has just finished processing both the Turing Machine description and the keyword “STATES:”
Expected Results	State class should contain two states, and maintain case sensitivity.

5.2 TM Parsing Algorithm (continued)

Test ID	5.2.3 - ParseDefinition_CheckForAtLeastOneState
Test Description	Verify Conformance of Requirement 4.2.1.2 of Requirements Document that there must be at least one state. This is accomplished by feeding a definition file with no states, and verifying that an exception was thrown.
Component	Type: Class method Name: States.load() bool States.load(List<string> definition_file)
Input Condition	List<string> invalid_definition = (“”, “INPUT_ALPHABET:”) States test_state = new States()
Input State	Turing Machine program has just finished processing both the Turing Machine description and the keyword “STATES:”
Expected Results	A false boolean should be returned from the load class method of the States class.

Test ID	5.2.4 - ParseDefinition_CheckForValidStateCharacters
Test Description	Verify Conformance of Requirement 4.2.1.4c of Requirements Document that only alphanumeric characters and the underscore character are allowed to be used in the naming of states. This is accomplished by feeding an invalid definition file using invalid characters, and verifying that an exception was thrown.
Component	Type: Class method Name: States.load() bool States.load(List<string> definition_file)
Input Condition	List<string> invalid_definition = (“$”, “INPUT_ALPHABET:”) States test_state = new States()
Input State	Turing Machine program has just finished processing both the Turing Machine description and the keyword “STATES:”
Expected Results	A false boolean should be returned from the load method of the States class.

5.2 TM Parsing Algorithm (continued)

Test ID	5.2.5 - ParseDefinition_CheckThatElementsAreLengthOne
Test Description	Verify Conformance of Requirement 4.2.2.2 of Requirements Document that input alphabet consists of elements of only length one. This is accomplished by feeding an invalid definition file that contains input alphabet elements of length greater than one.
Component	Type: Class Method Name: InputAlphabet.load() bool InputAlphabet.load(List<string> definition_file)
Input Condition	List<string> invalid_definition = ("ab", "TAPE_ALPHABET:") InputAlphabet test_inputalphabet = new InputAlphabet()
Input State	Turing Machine program has just finished processing both the Turing Machine description, the keyword "STATES:" and its elements, and the keyword "INPUT_ALPHABET".
Expected Results	A false boolean should be returned from the load class method of the InputAlphabet class.

Test ID	5.2.6 - ParseDefinition_CheckForDuplicateInputAlphabetCharacters
Test Description	Verify Conformance of Requirement 4.2.2.3 of Requirements Document that input alphabet consists of unique elements. This is accomplished by feeding an invalid definition file that contains duplicate input alphabet elements.
Component	Type: Class Method Name: InputAlphabet.load() bool InputAlphabet.load(List<string> definition_file)
Input Condition	List<string> invalid_definition = ("ab", "TAPE_ALPHABET:") InputAlphabet test_inputalphabet = new InputAlphabet()
Input State	Turing Machine program has just finished processing both the Turing Machine description, the keyword "STATES:" and its elements, and the keyword "INPUT_ALPHABET".
Expected Results	A false boolean should be returned from the load class method of the InputAlphabet class.

5.2 TM Parsing Algorithm (continued)

Test ID	5.2.7 - ParseDefinition_TransFunct_Valid
Test Description	Test that a transition function containing proper 5 components is parsed properly.
Component	transition_function
Input Condition	Definition input containing a transition function with 5 valid fields.
Input State	“TRANSITION_FUNCTION:” keyword has already been parsed.
Expected Results	all 5 fields are correctly parsed and stored in transition_function class.

Test ID	5.2.8 - ParseDefinition_TransFunct_InvalidFieldCount
Test Description	Test that a transition function NOT containing proper 5 components is produces an error.
Component	transition_function
Input Condition	Definition input containing a transition function without 5 valid fields.
Input State	“TRANSITION_FUNCTION:” keyword has already been parsed.
Expected Results	Should produce an error and return false.

Test ID	5.2.9 - ParseDefinition_TransFunct_InvalidChar
Test Description	Verify that accepted transitions have valid characters from the tape alphabet.
Component	transition_function
Input Condition	Definition containing a state that doesn’t exist in the tape alphabet
Input State	“TRANSITION_FUNCTION:” keyword has already been parsed.
Expected Results	Should have error

5.2 TM Parsing Algorithm (continued)

Test ID	5.2.10 - ParseDefinition_InitState_TooManyStates
Test Description	Validate only 1 initial state is accepted
Component	turing_machine
Input Condition	Give two initial states.
Input State	TRANSITION_FUNCTION: already parsed
Expected Results	Should have an error due to too many initial states.

Test ID	5.2.11 - ParseDefinition_BlankChar_NotInAlphabet
Test Description	Blank char must be member of tape_alphabet.
Component	turing_machine
Input Condition	blank char used that is not part of tape alphabet.
Input State	INITIAL_STATE: keyword has been parsed.
Expected Results	Should have an error.

Test ID	5.2.15 - ParseDefinition_FinalStates_NoStates
Test Description	Final states must have at least one final state.
Component	final_states
Input Condition	Pass 0 final states
Input State	INPUT_STATE: keyword has already been parsed.
Expected Results	Must have one or more final states

5.2 TM Parsing Algorithm (continued)

Test ID	5.2.16 - ParseDefinition_FinalStates_NotInStates
Test Description	Final State must be a member of states
Component	final_states
Input Condition	final_state defined that is not in states
Input State	FINAL_STATES: already parsed.
Expected Results	Should have error due to invalid final state.

5.3 User Input

Unique Test Identifier	5.3.1
Test Description	This test will be used to test the validity of TM Help command after entering ‘h’ or ‘H’ at the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document.
Expected Result	The first time ‘h’ is entered the TM command menu will be displayed and the Boolean value for help will be true for help messages enabled. Upon entering ‘H’ a second time the Boolean value will be false for help messages. See section 5.2.1 of system requirements for a representation.

Unique Test Identifier	5.3.2
Test Description	This test will be used to test the validity of TM Show command after entering ‘w’ or ‘W’ at the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document.
Expected Result	First test will be lower case ‘w’ is pressed the TM will display the current status. Pressing ‘W’ will result by displaying the TM current status. See section 5.2.2 of system requirements for a representation.

5.3 User Input (continued)

Unique Test Identifier	5.3.3
Test Description	This test will be used to test the validity of TM List command after entering ‘l’ or ‘L’ at the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document.
Expected Result	First test will be lower case ‘l’ is pressed the TM will display the list of input strings. Pressing ‘l’ will result by displaying the TM input strings. See section 5.2.4 of system requirements for a representation.

Unique Test Identifier	5.3.4
Test Description	This test will be used to test the validity of TM Insert command after entering ‘i’ or ‘I’ at the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document.
Expected Result	First test will be lower case ‘i’ is pressed the TM will prompt user to enter an input string. Pressing ‘I’ will result by displaying a prompt to enter input string. See section 5.2.5 of system requirements for a representation.

5.3 User Input (continued)

Unique Test Identifier	5.3.4b
Test Description	This test will be used to test the validity of TM Insert command by prompting user to enter an input string.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document. User has already pressed ‘I’ to execute the insert command.
Expected Result	User entering a valid string that is not currently in input string list will result in no prompt to the use and return to the main command input screen. See section 5.2.5 of system requirements for a representation.

Unique Test Identifier	5.3.5
Test Description	This test will be used to test the validity of TM Delete command after entering ‘d’ or ‘D’ at the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document.
Expected Result	First test will be lower case ‘d’ is pressed the TM will prompt user to enter the number for the string to delete. Pressing ‘D’ will result prompting user to enter the number for the string to delete. See section 5.2.6 of system requirements for a representation.

5.3 User Input (continued)

Unique Test Identifier	5.3.5b
Test Description	This test will be used to test the validity of TM Delete command after entering ‘d’ or ‘D’ at the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document. User has already pressed ‘d’ to execute the delete command.
Expected Result	User entering a number corresponding with a valid string that is currently in input string list will result in the removal of the string from the list and no prompt to the use and return to the main command input screen. See section 5.2.6 of system requirements for a representation.

Unique Test Identifier	5.3.6
Test Description	This test will be used to test the validity of TM Set command after entering ‘e’ or ‘E’ at the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document. User has already pressed ‘e’ to execute the Set command.
Expected Result	Used to set the maximum number of transitions to complete. User will be prompted for an integer of transitions. See section 5.2.7 of system requirements for a representation.

5.3 User Input (continued)

Unique Test Identifier	5.3.6b
Test Description	This test will be used to test the validity of TM Set command after ‘e’ has been invoked at the command line. User enters new value.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document. User has already pressed ‘e’ to execute the Set command.
Expected Result	Used to set the maximum number of transitions to complete. User will be prompted for an integer of transitions. Successful modification the variable Transitions. See section 5.2.7 of system requirements for a representation.

Unique Test Identifier	5.3.7
Test Description	This test will be used to test the validity of TM Truncate command after entering ‘t’ or ‘T’ on the command line.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document.
Expected Result	Used to set the maximum number of cells to display out. User will be prompted for an integer of cells to show. Successful modification of the variable number_of_cells. See section 5.2.8 of system requirements for a representation.

5.3 User Input (continued)

Unique Test Identifier	5.3.7b
Test Description	This test will be used to test the validity of TM Truncate command after ‘t’ has been invoked at the command line. User enters new value.
Component	Function Name (Enter section of Code this algorithm exists)
Input Condition	Input = TM has loaded a valid input file and is ready to accept user commands.
Input State	Values for Set, Truncate, and help are set to the default as referenced in section 5.1.2, Configuration Settings, of the Requirements Document. . User has already pressed ‘t’ to execute the Truncate command.
Expected Result	Used to set the maximum number of cells to display out. User will be prompted for an integer of cells to show. Successful modification of the variable number_of_cells. See section 5.2.8 of system requirements for a representation.

6.0 Test Summary

Below is a summary of results with references to any unresolved problem reports. See document entitled Defect Report attached this this Test Plan.

6.1 TM Definition File Validity Test Results

Unique Test Identifier	5.1.1 Parse definition - check parse correctly
Result	Test passed

Unique Test Identifier	5.1.2 Parse definition - check no state
Result	Test failed, loaded definition file with no state keyword

Unique Test Identifier	5.1.3 Parse definition - check no input
Result	Test failed, loaded definition file with no input character keyword

Unique Test Identifier	5.1.4 Parse definition - check no tape
Result	Test failed, loaded definition file with no tape character keyword

Unique Test Identifier	5.1.5 Parse definition - check no Transitions
Result	Test Failed, loaded definition file with noTransition keyword

Unique Test Identifier	5.1.6 Parse definition - check no initial
Result	Test passed

6.1 TM Definition File Validity Test Results (continued)

Unique Test Identifier	5.1.7 Parse definition - check no blank
Result	Test passed

Unique Test Identifier	5.1.8 Parse definition - check no final
Result	Test failed, loaded definition file with no final states keyword

6.2 TM Parsing Algorithm Validity Test Results

Unique Test Identifier	5.2.1 - ParseDefinition_CheckForDuplicateStates
Result	Test Failed - False was returned by method.

Unique Test Identifier	5.2.2 - ParseDefinition_CheckCaseSensitivityForStates
Result	Test Passed - Indicating that the State class does maintain case.

Unique Test Identifier	5.2.3 - ParseDefinition_CheckForAtLeastOneState
Result	Test Failed - False was returned by the method.

Unique Test Identifier	5.2.4 - ParseDefinition_CheckForValidStateCharacters
Result	Test Failed - False was returned by the method.

Unique Test Identifier	5.2.5 - ParseDefinition_CheckThatElementsAreLengthOne
Result	Test Failed - False was returned by the method.

Unique Test Identifier	5.2.6 - ParseDefinition_CheckForDuplicateInputAlphabetCharacters
Result	Test Failed - Class method failed to detect duplicate Input Alphabet characters.

Unique Test Identifier	5.2.10 - ParseDefinition_TransFunct_Valid
Result	PASS

6.2 TM Parsing Algorithm Validity Test Results (continued)

Unique Test Identifier	5.2.11 - ParseDefinition_TransFunct_InvalidFieldCount
Result	FAILED - Invalid transition accepted.

Unique Test Identifier	5.2.12 - ParseDefinition_TransFunct_InvalidChar
Result	FAILED - Invalid char accepted.

Unique Test Identifier	5.2.13 - ParseDefinition_InitState_TooManyStates
Result	PASS

Unique Test Identifier	5.2.14 - ParseDefinition_BlankChar_NotInAlphabet
Result	FAIL - Invalid blank char accepted.

Unique Test Identifier	5.2.15 - ParseDefinition_FinalStates_NoStates
Result	PASS

Unique Test Identifier	5.2.16 - ParseDefinition_FinalStates_NotInStates
Result	FAIL - Invalid final state accepted.

6.3 TM User Input Validity Test Results

Unique Test Identifier	5.3.1
Result	The first time ‘h’ is entered the TM command menu was displayed and the Boolean value for help was set to true for help messages enabled. Upon entering ‘H’ a second time the Boolean value was false for help messages.

Unique Test Identifier	5.3.2
Result	First test will be lower case ‘w’ is pressed the TM displayed the current status. Pressing ‘W’ resulted in displaying the TM current status.

Unique Test Identifier	5.3.3
Result	First test will be lower case ‘l’ is pressed the TM displayed the list of input strings. Pressing ‘I’ resulted in displaying the TM input strings.

Unique Test Identifier	5.3.4
Result	First test will be lower case ‘i’ is pressed the TM prompted user to enter an input string. Pressing ‘I’ resulted in displaying a prompt to enter input string.

Unique Test Identifier	5.3.4b
Result	Entering a valid string not currently in input string list resulted in no prompt to the use and returned to the main command input screen.

6.3 TM User Input Validity Test Results (continued)

Unique Test Identifier	5.3.5
Result	First test will be lower case ‘d’ is pressed the TM prompted user to enter the number for the string to delete. Pressing ‘D’ resulted in prompting user to enter the number for the string to delete.

Unique Test Identifier	5.3.5b
Result	Entering a number corresponding with a valid string that is currently in input string list resulted in the removal of the string from the list and no prompt to the use and returned to the main command input screen.

Unique Test Identifier	5.3.6
Result	Used to set the maximum number of transitions to complete. User was prompted for an integer of transitions.

Unique Test Identifier	5.3.6b
Result	Used to set the maximum number of transitions to complete. User was prompted for an integer of transitions. Successful modification of the variable Transitions.

Unique Test Identifier	5.3.7
Result	Used to set the maximum number of cells to display out. User was prompted for an integer of cells to show.

6.3 TM User Input Validity Test Results (continued)

Unique Test Identifier	5.3.7b
Result	Used to set the maximum number of cells to display out. User was prompted for an integer of cells to show. Successful modification of the variable number_of_cells after entered.