Turing Machine Testing Procedure

CPTS322 Software Engineering I

Christian Penick

April 29, 2024

### Parsing Turing Machine file.

To ensure the Turing Machine (TM) application could properly parse the TM definition file, I ran the application on a TM definition file that leveraged every freedom provided by the requirements specification. This includes inserting whitespaces arbitrary (except if whitespace would break token), using various combinations of uppercase/lowercase letters for keywords, placing components of TM on same line as keyword, and providing an empty description of the TM.

### Validating Turing Machine Definition

To ensure only valid definitions are accepted by the TM, I ran the application on a TM definition file that included mistakes for every component of the definition file. This including misspelling keywords, providing states with unacceptable characters, providing input alphabet elements with unacceptable characters, providing input alphabet elements not in the tape alphabet, providing tape alphabet elements with unacceptable characters, defining transition functions with states not in the set of states, defining transition functions with read/write/move direction tokens that are not a single character, defining transition functions with unacceptable move direction, providing unacceptable blank character, providing final state not in the set of states.

### Parsing and Validating Input String File

To ensure the TM application could properly parse the TM input string file, I ran the application on a TM input string file that attempted all configurations allowed in the requirements specification. This included inserting multiple newlines between strings, using a single newline between strings, using the empty string character within a string, and using various printable characters in strings.

### Validating Turing Machine Operation

To ensure the TM application operates correctly, I performed sequences of commands and permutations on those sequences such as *list, insert, view, run* and *run, insert, view, list*. I also ran the TM that accepts some number of As followed by the same number of Bs on various input strings such as AABB, AAB, BBA, AAAAAAABBBBBB.