Project 2 Readme Team Lagunowich

1	Team Name: lagunowich
2	Team members names and netids: Luke Lagunowich - llagunow
3	Overall project attempted, with sub-projects: Program 1: Tracing NTM Behavior
4	Overall success of the project: Completed and rigorously tested.
5	Approximately total time (in hours) to complete: 15 hours
6	Link to github repository: https://github.com/lukelagunowich/cse-30151-project-2-lagunowich
7	List of included files (if you have many files of a certain type, such as test files of different sizes, list just the folder): (Add more rows as necessary). Add more rows as necessary.

File/folder Name	File Contents and Use
Code Files	
code_traceTM_lagunowich.py	Contains functions used to trace NTM behavior. Includes “readTM” function to read a TM schema from a CSV file, “transition” function that takes an input of a list of states and outputs all possible transitions from those states given the specified transition function, “recurse” function recursively traces an NTM using the “transition” function until it either accepts, rejects, or reaches a maximum depth, “traceTM” reads a turing machine schema, calls the recursive function using a specified input string to start, and outputs the trace as well as other metrics, and “output” function to write the results of a trace to a specified file.
code_generate_outputs_lagunowich.py	Generates output files for various NTMs with various inputs.
Test Files	
check_traceTM_lagunowich.py	Compares output of “traceTM” function for three input strings, one that accepts, one that rejects, and one that reaches the max depth to hand-calculated result using assert statements. Verifies the functionality of the NTM trace. NTM used includes both left and right transitions.

<table>
<tr><td></td><td>
<table>
<tr><td colspan="2">Output Files</td></tr>
<tr><td>output_{machine}_{input}_lagunowich.txt</td><td>Files include name of machine, initial string specified, total transitions, average nondeterminism, whether or not the trace was accepted, rejected, or reached a maximum depth and in how many transitions, and the trace.</td></tr>
<tr><td colspan="2">Plots (as needed)</td></tr>
<tr><td>plot_NTM_statistics_lagunowich</td><td>Compares statistics listed above for various machines with various inputs.</td></tr>
</table>
</td></tr>
<tr><td>8</td><td>Programming languages used, and associated libraries: Python (csv library)</td></tr>
<tr><td>9</td><td>Key data structures (for each sub-project): NTM trace used a tree which was represented by a list of lists.</td></tr>
<tr><td>10</td><td>General operation of code (for each subproject): NTM trace code relied on recursion to trace the path of the provided NTM on the provided input. At each stage of the recursion, the code would check base cases (no more transitions, reject, accept, or maximum depth reached) before recursing on the next depth of the tree. For each current depth, the transition function would be called which iterated through each state in the depth and output all possible transitions given the transition function to the next depth. The tree data structure was recursively concatenated.</td></tr>
<tr><td>11</td><td>What test cases you used/added, why you used them, what did they tell you about the correctness of your code: Utilized a file which asserted the output of the “traceTM” function with hand-calculated statistics. The file includes a test case for an accepted input, a rejected input, and an input which triggers the maximum depth reached termination flag.</td></tr>
<tr><td>12</td><td>How you managed the code development: Practiced iterative development using GitHub, and attempted some aspects of OOP when developing function inheritance.</td></tr>
<tr><td>13</td><td>Detailed discussion of results: traceTM code correctly traces the path of a specified NTM with a given input. One of the statistics calculated is average nondeterminism. It is interesting to see how different inputs impact nondeterminism. It appears that the longer and less uniform an input is, the higher degree of average nondeterminism. Furthermore, I hypothesize that the more complex the NTM, the greater the degree of average nondeterminism.</td></tr>
<tr><td>14</td><td>How team was organized: I was the sole member of my team.</td></tr>
<tr><td>15</td><td>What you might do differently if you did the project again: If I did this project again, I would have more rigorously tested my NTM trace before attempting to develop different inputs and outputs. I caught bugs far too late in my development for my liking. If I had more diligently tested my function initially, it would have saved me time debugging later on. Also, I potentially could have found a more efficient way to develop test cases instead of comparing outputs to hand-calculations.</td></tr>
</table>

16	Any additional material: N/A