Project 2 Readme Team JT

Version 1 9/11/24

A single copy of this template should be filled out and submitted with each project submission, regardless of the number of students on the team. It should have the name readme_"teamname"

Also change the title of this template to "Project x Readme Team xxx"

<table>
<tr><td>1</td><td>Team Name:

JT</td></tr>
<tr><td>2</td><td>Team members names and netids:

Joshua Tighe, jtighe3</td></tr>
<tr><td>3</td><td>Overall project attempted, with sub-projects:

NTM Tracer</td></tr>
<tr><td>4</td><td>Overall success of the project:

Completed successfully</td></tr>
<tr><td>5</td><td>Approximately total time (in hours) to complete:

5</td></tr>
<tr><td>6</td><td>Link to github repository:

https://github.com/joshuatighe/Project2-TOC</td></tr>
<tr><td>7</td><td>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.
<table>
<tr><td>File/folder Name</td><td>File Contents and Use</td></tr>
<tr><td colspan="2">Code Files</td></tr>
<tr><td>traceTM_JT.py</td><td>NTM Tracer code
Use:
python traceTM_JT.py <filename> "<input_string>" <max_depth></td></tr>
<tr><td colspan="2">Test Files</td></tr>
<tr><td>input/aplus.csv</td><td>aplus: a+</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>input/composite.csv
input/w#w.csv</td><td>composite: provided
w#w: from book {w#w | w is an element of {0,1}*}</td></tr>
<tr><td colspan="2">Output Files</td></tr>
<tr><td>None</td><td>Prints to terminal</td></tr>
<tr><td colspan="2">Plots (as needed)</td></tr>
<tr><td>None</td><td>N/A</td></tr>
</table>
</td></tr>
<tr><td>8</td><td>Programming languages used, and associated libraries:

Python</td></tr>
<tr><td>9</td><td>Key data structures (for each sub-project):

Transitions of TM: List[Dict (current_state, current_char, next_state, write_char, direction)]
Config: List[left string, current state, right string]
Tree: List[List[Config]]</td></tr>
<tr><td>10</td><td>General operation of code (for each subproject)

NTM Tracer:
1. Parse command line arguments for filename, input_string, max_depth
2. Parse TM headers from CSV (ignored Q, epsilon, gamma as they were not needed)
3. Parse all transitions from CSV, add them to transitions list
4. Initialize tree with the starting config [[‘’, start_state, input_string]]
5. While the current depth is less than the max depth
a. Check if all configs at current depth are in rejected state
b. Loop through each config at current depth, find valid transitions
i. If no valid transitions, implicit reject
ii. Append all valid transitions’ configs to the next level of the tree
c. If any accepted transitions, exit while loop
d. Increase depth
6. Print results depending on:
a. Accepted config found
b. All configs rejected
c. Depth exceeded max depth (there may be a solution at a higher depth)
7. Print each level of tree</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.

w#w.csv - complex DTM from book to check edge cases
aplus.csv - basic NTM to verify code worked for NTMs
composite.csv - required screenshot for submission</td></tr>
</table>

12	How you managed the code development 1. First handled argument parsing, csv parsing, placeholder output (i.e. “tailends” of the code) 2. Then handled the main logic of actually generating the config tree 3. Struggled with out-of-bounds edge cases for transitions (i.e. aaaaq1, with no right string), was hard to wrap my head around these cases
13	Detailed discussion of results: There was three different types of results depending on the TM, input_string, and max_depth: 1. Accepted config found 2. All configs at a depth rejected 3. Depth exceeded max_depth before 1. or 2. could happen I printed the config tree and total_transitions for each regardless. For 1. I also printed how many instructions it took to get there (the depth), for 2. I printed the depth at which all configs were in reject, and for 3. I simply stated that depth had exceeded max_depth
14	How team was organized N/A, individual
15	What you might do differently if you did the project again - Use of classes to simplify data structions, e.g. a TuringMachine class like the one seen in Lax’s repository - Better argument parsing and handling - Use of functions for repetitive sections of the code
16	Any additional material: N/A