Project Readme Template

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: tfriedma</td></tr>
<tr><td>2</td><td>Team members names and netids: Ty Friedman: tfriedma</td></tr>
<tr><td>3</td><td>Overall project attempted, with sub-projects: hamiltonian path</td></tr>
<tr><td>4</td><td>Overall success of the project: very successful</td></tr>
<tr><td>5</td><td>Approximately total time (in hours) to complete: 8 hours (rough estimate)</td></tr>
<tr><td>6</td><td>Link to github repository: https://github.com/tyfriedman/hamiltonian_path</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>solution_tfriedma.py</td><td>This generates the graphs and tests them from the given input file specified on lines 25 and 26. Formatting of output can be changed on line 63. This file can be ran with ./solution_tfriedma.py</td></tr>
<tr><td>test_generator.py</td><td>This generates random graphs and places them in an output file specified in line 52. The number of graphs generated can be specified in the function generate_test_cases(), as explained in the comments of the file. This file can be ran with ./test_generator.py</td></tr>
<tr><td colspan="2">Test Files</td></tr>
<tr><td>testing_graphs_tfriedma.py</td><td>This file is the same as the testing file provided in a previous class announcement and is used to test if graphs are correctly classified as satisfiable or unsatisfiable. This file can be used as an input to solution_tfriedma.py by uncommenting line 25 in solution_tfriedma.py.</td></tr>
<tr><td>timing_graphs_tfried</td><td>This file was generated using test_generator.py and contains</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td>
<table>
<tr><td>ma.py</td><td>graphs with numbers of nodes from 3-12. For each size of graph, 15 were generated with a solution, and 3 were generated without a solution. This file can be used as an input to solution_tfriedma.py by uncommenting line 26 in solution_tfriedma.py.</td></tr>
<tr><td colspan="2">Output Files</td></tr>
<tr><td>output_tfriedma.txt</td><td>This is a .txt file copied from the terminal of the outputs of the timing and testing input files.</td></tr>
<tr><td>output_tfriedma.pdf</td><td>This is a .pdf version of the above file.</td></tr>
<tr><td colspan="2">Plots (as needed)</td></tr>
<tr><td>plots_testing_graphs_tfriedma.png</td><td>Plot generated from testing_graphs_tfriedma.py</td></tr>
<tr><td>plots_timing_graphs_tfriedma.png</td><td>Plot generated from timing_graphs_tfriedma.py</td></tr>
</table>
</td></tr>
<tr><td>8</td><td>Programming languages used, and associated libraries: Python - csv, time, itertools, random, math</td></tr>
<tr><td>9</td><td>Key data structures (for each sub-project): graphs (adjacency list representation), strings, csv files</td></tr>
<tr><td>10</td><td>General operation of code (for each subproject): Solution.py generates all possible solution paths and then checks each path to see if it is a valid solution. If a valid solution is found, then True is returned.</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.
I added the class-provided test cases that was used to check the accuracy of my code. I chose this file because it had known outputs. Additionally, I created a random graph generator to generate graphs that were used to time my solution. I created this because it made it easier to time a larger number of input graphs.</td></tr>
<tr><td>12</td><td>How you managed the code development:
I managed the code development on GitHub, working entirely on the master branch. If I were to have had group members, it is likely that I would have used different branches and merged them to the main branch after they were verified.</td></tr>
<tr><td>13</td><td>Detailed discussion of results:
My solution was very inefficient. I could only run a graph of 12 nodes that doesn’t have a solution, as that took about a minute per check. This is because there was no optimization in the way that I was checking solutions and because of the fact that I was generating and checking every single possible graph.</td></tr>
</table>

14	How team was organized: **I was the only group member, so I had complete control over everything.**
15	What you might do differently if you did the project again: **I would have liked to have explored multiple different optimized solutions to see how large of a graph I could find solutions to without the program running for too long.**
16	Any additional material: **N/A**