Deterministic Finite Automata (DFA) minimization

Description:

This Python software accepts a description of a deterministic finite automaton (DFA) and returns a minimized version of that DFA. The script initially invites the user to provide information on the DFA, such as its states, alphabet, beginning and end states, and transition functions. It then creates a DFA graph with the PyGraphviz module, visualizes it, and uses the minimization process to locate and merge comparable states. Finally, it outputs the minimized DFA, transition table, and visualizations. This way is called the equivalence theorem.

Input Format:

* The user is prompted to input:
  + Alphabet
  + Initial state o Final state(s) o State values
  + Transition functions

Output Format:

* The script outputs:
  + Initial state of the minimized DFA o Final state(s) of the minimized DFA o Transition table of the minimized DFA o Visualization of the minimized DFA graph

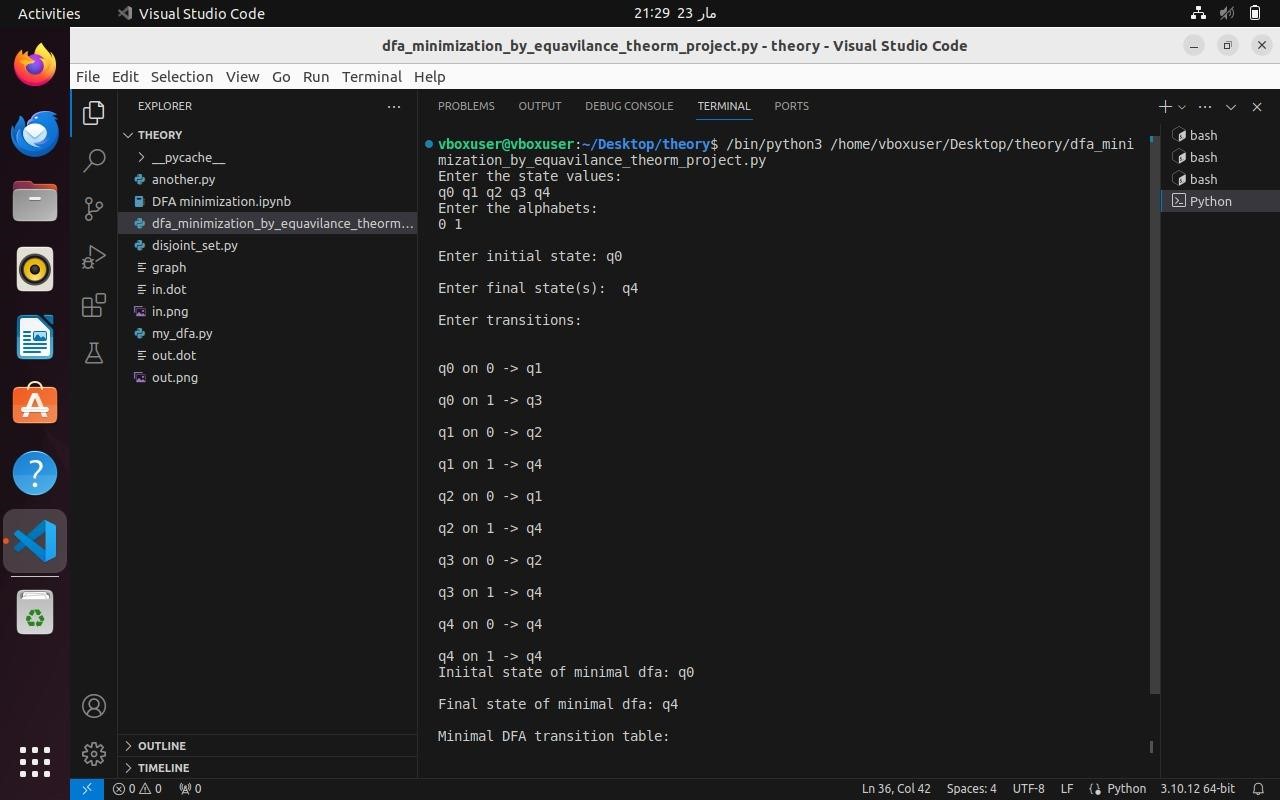
Inside Mechanism:

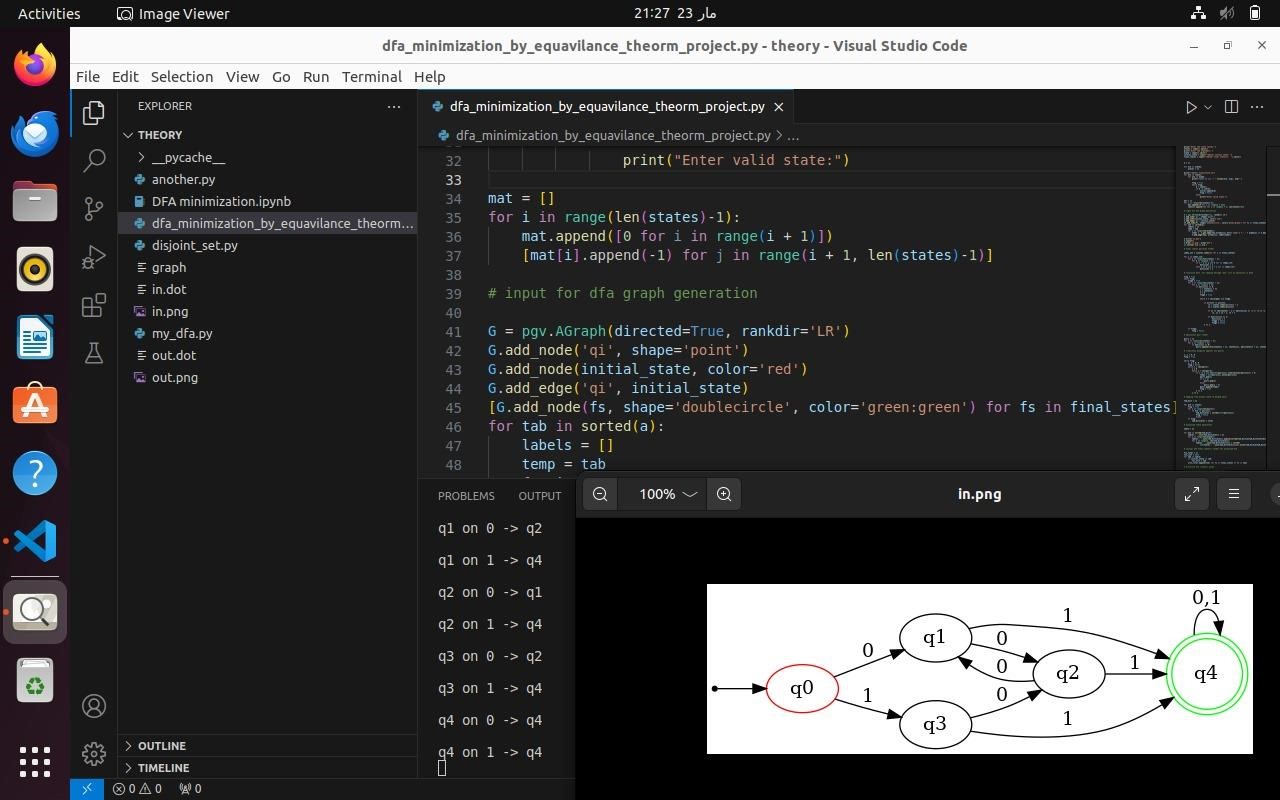
Contributed to developing the DFA minimization project, a Python tool that optimizes Deterministic Finite Automata (DFA) by reducing states while preserving language recognition. Using a table-filling method, the script systematically analyzes state transitions, identifying and merging equivalent state pairings to create a minimized DFA. It leverages PyGraphviz for graph visualization and the OS library for system commands, providing essential DFA information to enhance understanding of automata theory and algorithmic optimization.

The script is written in Python, a high-level programming language known for its simplicity and readability. It utilizes several tools and libraries:

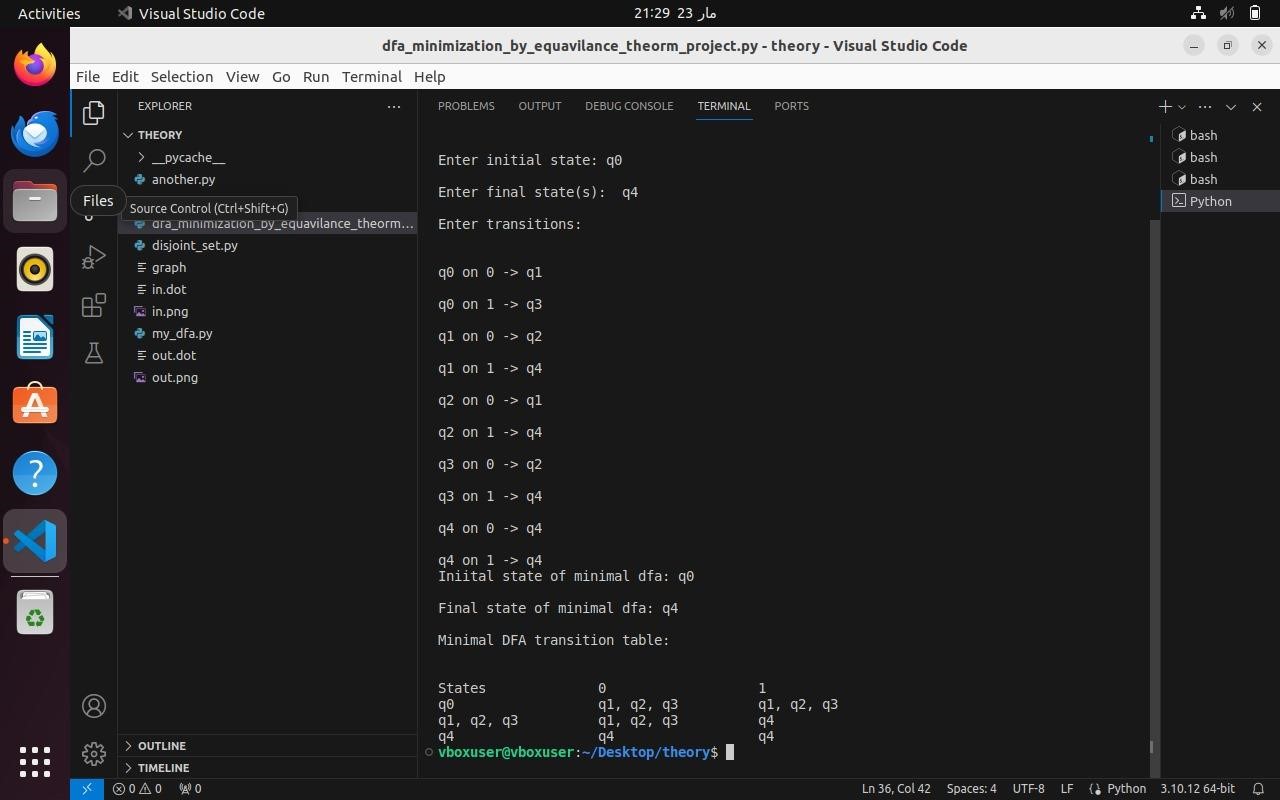
* PyGraphviz: This is a Python interface to the Graphviz graph layout and visualization package. It allows the script to create, manipulate, and visualize graphs.
* Graphviz: Graphviz is an open-source graph visualization software. It provides tools for generating graph layouts and rendering them into various formats, such as PNG images or DOT files.
* Operating System Commands: The script uses operating system commands via the OS module to execute commands like opening an image file for visualization.

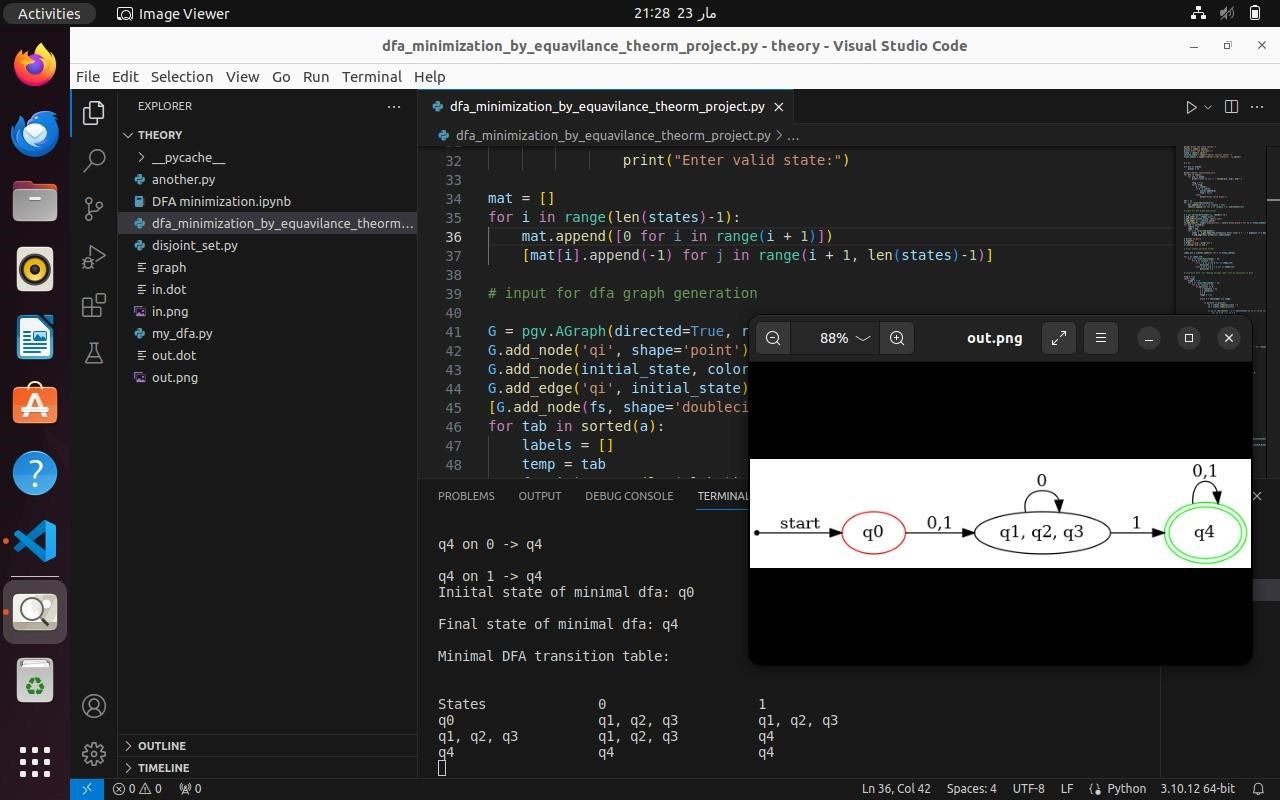
Input:



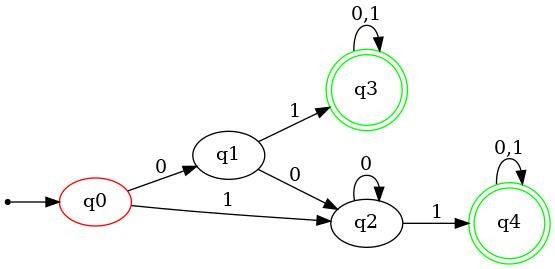


Output:





Input:



Output:

