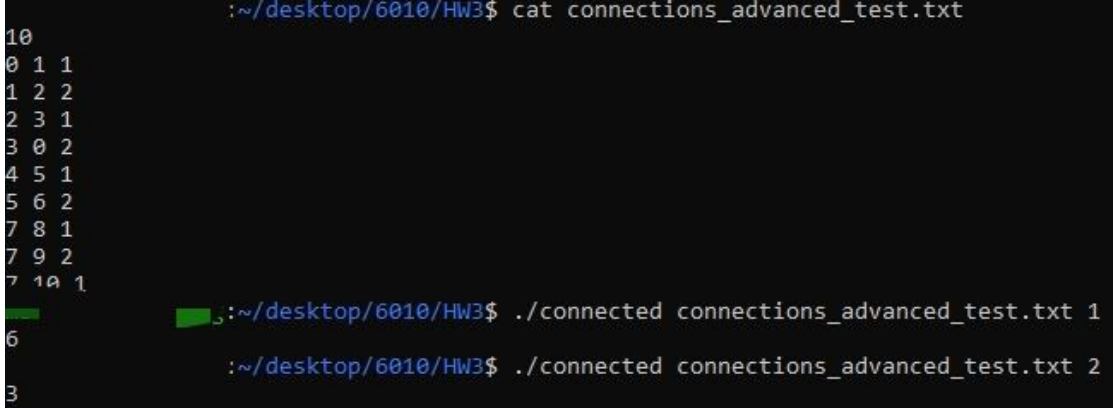
Program Structure and Development

Mingzheng Michael Huang

Translating Algorithm Components

- ***Repetition:*** Utilized while loops for conditions requiring repetitive action, such as traversing the graph until all vertices are visited, and for loops for iterating over adjacency lists during DFS. Break statements halt repetition for specific conditions like already visited vertices.
- ***Decision Making:*** If/else statements are crucial in determining a vertex's status, either visited or unvisited, and handling edge cases like checking for a valid vertex.
- ***Math:*** While this program mainly revolves around graph traversal without much emphasis on mathematical computations, the structure ensures clarity and readability for the user.
- ***Names:*** Vertices in the graph are represented as integers, and the adjacency list forms a dynamic array of lists to signify connections.
- ***Altering Values:*** Upon traversal, a vertex's visited status gets updated. Similarly, discovering a new component results in an incremented connected component count.
- ***Complicated Steps:*** Abstracting the depth-first search and reading the graph from a file into separate functions streamlines the main program. The depthFirstSearch function is pivotal in exploring the graph and identifying connected components.
- ***And the answer is...!:*** Post traversal and processing, the program promptly prints the number of connected components in the graph before concluding its execution.

Program Validation



Stress Testing Approach (See README.TXT)

- Comprehensive testing of complex graph structures with varying connected components.
- Ensured detection of distinct connected components with alternating edge weights.
- Validation of correct weight interpretation.

Validation Criteria

- Depth-First Search must correctly identify separate connected components in the graph.
- Accurate reading and processing of edge weights.
- Handling of repeated edges and unique vertex identifiers.

Results

- All tests, including the advanced graph structure test (connections_advanced_test.txt), were successfully executed, verifying the accurate and robust operation of the connectivity detection program. The algorithm effectively determines connected components and respects given edge weights, even in complex scenarios.