Report - assignment 6

Camilla Kristiansen Birkelund IDATT2101 October 8th, 2022

Assignment description

The assignment was to make a program that could read a graph from a file, find the strongly connected components of a graph and print how many there are. If there are less than 100 nodes, the nodes in each component should be printed as well.

Implementation

The source code for the assignment can be viewed in the file “scc.cpp”. The library “vector” was implemented to create a graph, which is a vector of vectors. To be able to create graphs from files, the method from_file was implemented. By using several libraries, such as fstream and string, the method is first able to set the size (the amount of nodes) by reading the first element on the first line in the file. The method then adds the nodes and their related edges by using a while loop that lasts until there are no more lines in the file.

The method visit was implemented to keep track of which nodes have been visited during a depth first search. The visit method uses a boolean value to be able to determine whether or not the current node has been visited. The default value is false (the node has not been visited), and will be changed to true when it has been visited. The dfs method iterates through the nodes and returns the order of the depth first search.

The invert method inverts the original graph, which means that the edges change directions. The inverted method is used in the printSCC method, where it is used to find the strongly connected components along with the dfs method and the visit method. The method iterates through the nodes and registers the visited nodes to find the components. The components are printed with the belonging nodes if the total number of nodes is less than 100. If there are more than 100 nodes, the print will only be the amount of components in the graph.

Testing

To ensure that the from_file method was working as expected, the print method was implemented. The print method prints the resulting graph from the from_file method. When using the example given in the assignment (“graphø6g6.txt”) the results correctly displayed the graph by listing the nodes and their neighbors.

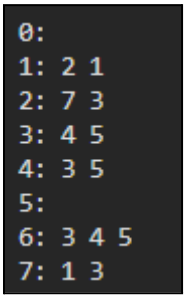

Figure 1: The result of the print method of the graph retrieved from the “graphø6g6.txt” file. The result is as expected.

Using the same file, the program was run to see if it was able to give the expected results, which would be 5 components, where component one would contain node 6, component two would have 1, 7 and 2, component three would have 3 and 4, component four would have 5, and component five would have no nodes (0).

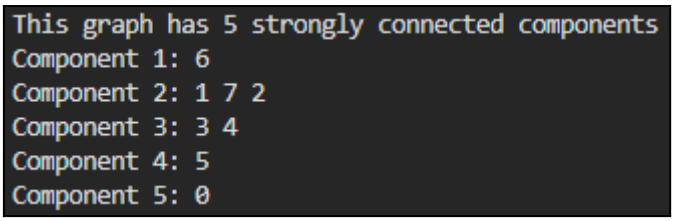

Figure 2: The results of running the program using the “graphø6g6.txt” file. The results are as expected.

Results

The program is able to perform the tasks described in the assignment, but has a few weaknesses in terms of exception handling. If the amount of nodes in the graph does not match the amount described in the first element on the first line, the program will crash. The same applies to the amount of edges. The program does not require a certain file type, which means that the user has to make sure that the content of the file is compatible with the way the program is implemented. This means that an improvement of the solution presented would be adding exception handling to make the program more user friendly.