CSCE 221 Cover Page
Programming Assignment #5

Due Date: April 26th

First Name Annemarie Last Name

Bell UIN User Name

aeb178 E-mail
address aeb178@tamu.edu

Please list all sources in the table below including web pages which you used to solve or implement the current homework. If you fail to cite sources you can get a lower number of points or even zero. According to the University Regulations, Section 42, scholastic dishonesty are including: acquiring answers from any unauthorized source, working with another person when not specifically permitted, observing the work of other students during any exam, providing answers when not specifically authorized to do so, informing any person of the contents of an exam prior to the exam, and failing to credit sources used. Disciplinary actions range from grade penalties to expulsion read more: Aggie Honor System Office

Type of sources			
People			
Web pages (provide URL)	https://www.cplusplus.com	https://www.geeksforgeeks.org	https://www
Printed material			
Other Sources			

*https://www.cplusplus.com/reference/unordered_map/unordered_map/insert/
*https://www.geeksforgeeks.org/tokenizing-a-string-cpp/
*https://www.geeksforgeeks.org/topological-sorting/

I certify that I have listed all the sources that I used to develop the solutions/codes to the submitted work.

"On my honor as an Aggie, I have neither given nor received any unauthorized help on this academic work."

Electronic signature Annemarie Bell Date 04/26/2021

DESCRIPTION:

- In this program using unordered_map we create a directed acyclic graph to find the topological ordering in a DAG. In the first part of this assignment, we focus on the creation of the graph, then for the second half create the topological ordering to find the cycle of the graph.

FEATURES:

- Vertex Structure: The vertex structure is going to define a vertex in the terms of a graph. In this structure contains, the label, an adjacency vector (holds the nodes that correspond to the vertex), the indegree (how many vertex's point to said vertex), top_num (the topological order num of the vertex) and the vertex constructors.
- Vertex Compare: This structure compares to vertexes and decides the method which vertex is greater than the other, then places it into the priority queue as such.
- BuildGraph : This function creates a graph, taking from the file and importing it into an unordered map. This functions BigO algorithm is going to be equal to O(n^2) since the function must loop through the entirety of the file, then also through each line a second time to split the line up into the label and adj_vert.
- At: This function returns the vertex at a certain label. The bigO of this function is going to be equal to O(1) because it simply just returns a value without any iteration.
- Size: This function returns the number of vertices in the graph, the bigO of this function is equal to O(1) because it simply calls on the function.
- displayGraph: This function outputs the graph staring with the label: all the adjacent vertices. This function has a bigO of O(n) because the function goes through the entire graph once.
- Compute_indegree: this function returns how many vertices point to the vertex, this would be a big O of O(n^2) because it loops though the elements and adds them to a queue before doing it again to then add to get the n degree of the vertex.
- Topological_sort: This function sorts through the graph placing the vertices in a linear sort, that follows their input degree, when visited. The bigO of the topological sort is normally O(logn) but since I used a priority queue to help with the sort the bigO would be O(nlogn).
- PrintTopological_sort: This function prints through the queue of the topologically sorted graph. The big O of this function is O(n) because the function goes though the graph one time to get all the elements of the graph.

· Why does the topological sort algorithm use a queue? Can we use a stack instead?

- A queue is used inorder to help the algorthim not have to scan through each vertex to find the indegree that is equal to zero for the sorting, rather it can just be poped off the queue to save a bit of time in the sorting progress.
- Using a stack would sort depth first rather than breadth first, it is possible to use a stack but it could take significiantley more time then using a queue.

Can you explain **why** the algorithm detects cycles?

- The idea behind a Topological sort is to visit the parent node followed by the child node, so that each parent node has a child node, and in this atleast one node will be both a parent and a cycle and this will break the cycle this ending the Topolgocial sort since it is a linear sort.