THE UNIVERSITY OF WESTERN AUSTRALIA

School of Computer Science and Software Engineering

CITS2200 Data Structures and Algorithms Home

Project

Due: Friday, May 31, 2019, 11:59pm
This project can be done in pairs, although you may work on your own if you prefer.

Please note that the project report must be in pdf format, your project may not be checked if the report is in any other format.

This project is a multi-part project, which means that it comprises several questions. The topic of this project is to analyse parts of the Wikipedia graph. You will need to create a class that can store and analyse various features of a Wikipedia page graph. This class should implement this interface. Wikipedia contains numerous pages on varied topics. These pages may link to other pages inside Wikipedia. If we treat pages as vertices, and links as directed edges, then Wikipedia can be viewed as a graph. For more information about graphs see the Trees and Graphs and Graph Traversals CITS2200 lectures. We will not be dealing with the entire Wikipedia page graph, as it is too large. Instead, your class should be expected to work with arbitrarily selected subsets of pages along with the links between these pages.

Question 1. Write a method that, given a pair of pages, returns the minimum number of links you must follow to get from the first page to the second. See here for more information.

Question 2. Write a method that finds a Hamiltonian path in a Wikipedia page graph. A Hamiltonian path is any path in some graph that visits every vertex exactly once. This method will never be called for graphs with more than 20 pages. See here for more information about Hamiltonian paths.

Question 3. Write a method that finds every 'strongly connected component' of pages. A strongly connected component is a set of vertices such that there is a path between every ordered pair of vertices in the strongly connected component. See here for more information about strongly connected components.

Questions 4. Write a method that finds all the centers of the Wikipedia page graph. A vertex is considered to be the center of a graph if the maximum shortest path from that vertex to any other vertex is the minimum possible. See here for more information about 'graph center'.

Testing Data: It is recommended that you test your implementation thoroughly with your own test data. However, a small sample test graph can be found here. Note that a pair of consecutive lines define en edge in this file. A class that loads this test data and can be used to test your implementation can be found here.

This project is worth 20% of your final grade. The marking scheme is as follows:

- The marks for individual questions are, Q1 (3 marks), Q2 (7 marks), Q3 (6 marks), Q4 (4 marks)
- You have to submit a report. The report should contain the pseudo codes or descriptions of all the algorithms that you have used, the source of the algorithms (if you have used existing algorithms), complexity analysis of the algorithms, some performance studies on different size graphs.
- Corrent implementations of algorithms will get 70% of the marks, you need to design/use/explain efficient algorithms in order to score higher.

Here are some more clarifications and explanations.

The project should be submitted using cssubmit.

ALGORITHM QUOTES

"The Feynman Problem Solving Algorithm: (1) write down the problem; (2) think very hard; (3) write down the answer." *Richard Feynman*

"Simplicity is a great virtue but it requires hard work to achieve it and education to appreciate it. And to make matters worse: complexity sells better." *Edsger Dijkstra*

"Beware of bugs in the above code. I have only proved it correct, not tried it." *Donald Knuth*

"There are two ways of constructing a software design: One way is to make it so simple that there are obviously no deficiencies, and the other way is to make it so complicated that there are no obvious deficiencies. The first method is far more difficult.", *Tony Hoare*

"We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil.", *Donald Knuth*