Reflective Report

Aaron Tsatsu Tamakloe

49622024

Dr. Ebo Adjepon-Yamoah

September 30, 2022

Taking on this individual project greatly challenged my critical thinking skills and programming ability. In previous programming courses that provided step-by-step guidelines for completing projects, I found more straightforward ways to solve the project problems. However, this project introduced me to a new 'ambiguous' way of programming. In this reflection, I ponder on my thought processes throughout the challenge.

To begin with, I resorted to UML diagrams to get an overview of the needed classes in the program. Thinking through the problem at a high level, I figured that the files from which the program's data was extracted had similar attributes in terms of identification (ID) numbers. This discovery led me to think of the data structures I could use to store the data after reading from the files. I utilised online resources like programiz.com and geeksforgeeks.org to research the relevance of some structures to the project. At a high level, I considered using a hash map, a linked list, and an array list.

Moving on to the relevant algorithms to use, I inferred from my knowledge gained in the Introduction to Artificial Intelligence course. I considered the uniform cost search algorithm and Dijkstra's algorithm. At the implementation level, I struggled to integrate the read data files into the algorithm to find the routes as expected. To fix this confusion, my research led me to a java library, org.jgrapht, which provides functions to depend on for implementing the graph data structure, and algorithms to traverse the graphs. Although I struggled with implementing the graph structure in the problem context, I understood problem-solving better.

In conclusion, this project introduced me to solving programming problems within a somewhat ambiguous setting. I learnt that incremental development is an essential aspect of programming. Finally, this project improved my ability to conduct technical research with online resources and Java API documentation.

References

GeeksforGeeks. (2021, August 25). *Uniform-Cost Search (Dijkstra for large Graphs)*. Retrieved September 30, 2022, from https://www.geeksforgeeks.org/uniform-cost-search-dijkstra-for-large-graphs/

*Java Platform SE 7*. (n.d.). Retrieved September 30, 2022, from https://docs.oracle.com/javase/7/docs/api/

*Java Program to Implement Uniform-Cost Search*. (2022, May 24). Sanfoundry. Retrieved September 30, 2022, from https://www.sanfoundry.com/java-program-implement-uniform-cost-search/

*org.jgrapht (JGraphT : a free Java graph library)*. (n.d.). Retrieved September 30, 2022, from https://jgrapht.org/javadoc/org.jgrapht.core/org/jgrapht/package-summary.html