GTU DEPARTMENT OF COMPUTER ENGINEERING

CSE 222/505 DATA STRUCTURES AND ALGORITHM

HOMEWORK 8 REPORT

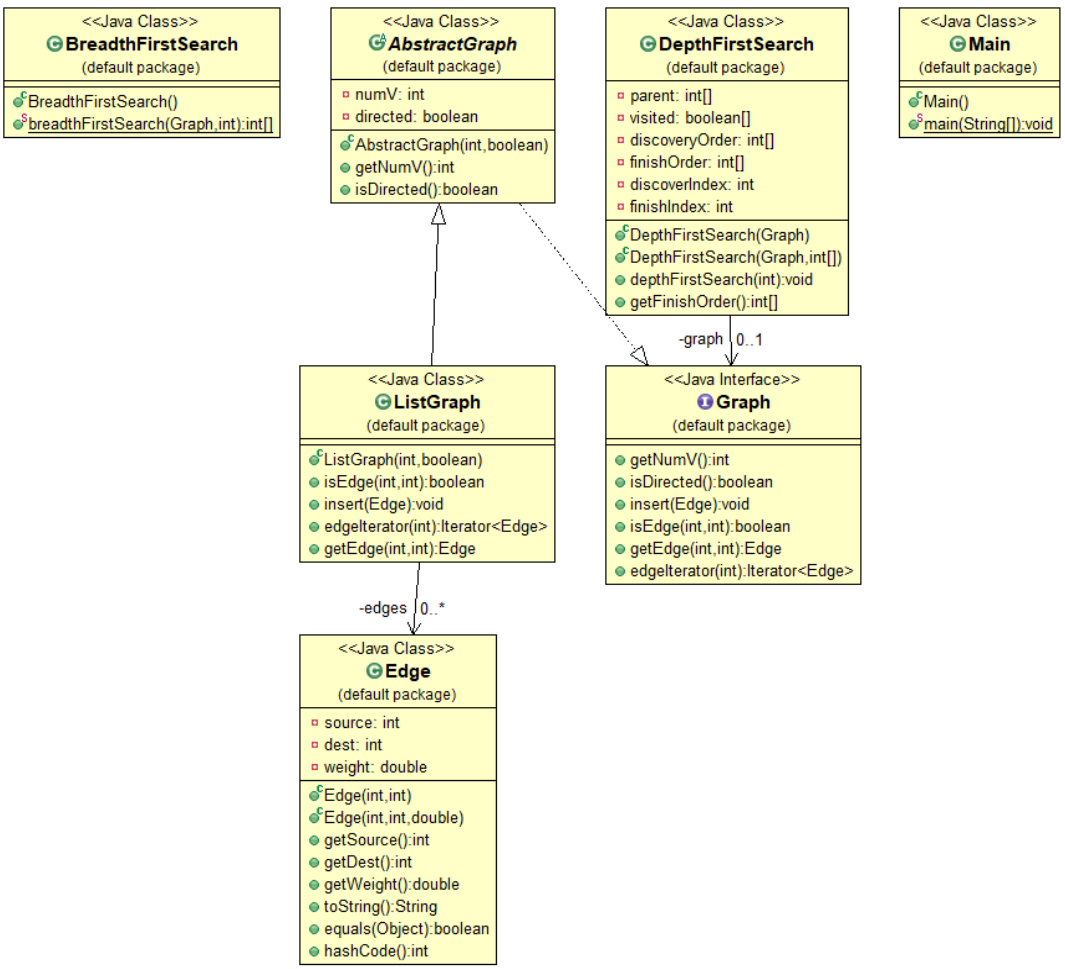
171044076

ŞEYMA NUR CANBAZ

1. SYSTEM REQUIREMENTS:

In this part, we measured the breadth first search and depth first search. So I created this class. For creating graph, I added the Graph and ListGraph class.

2. CLASS DIAGRAMS:



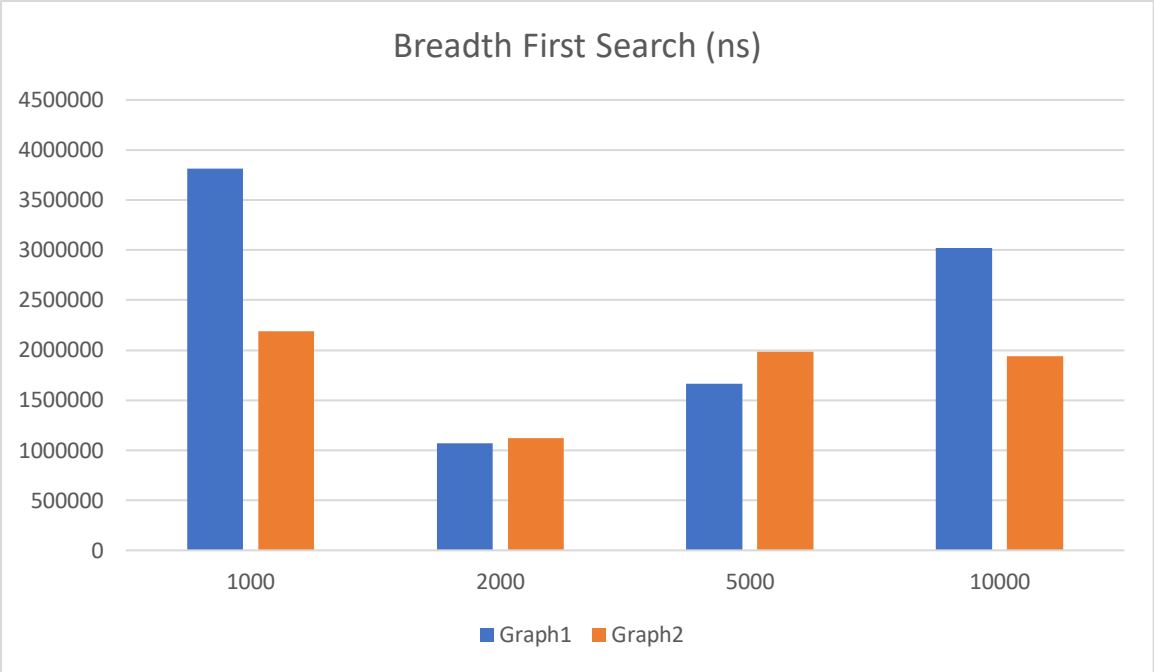
3. PROBLEM SOLUTION APPROACH:

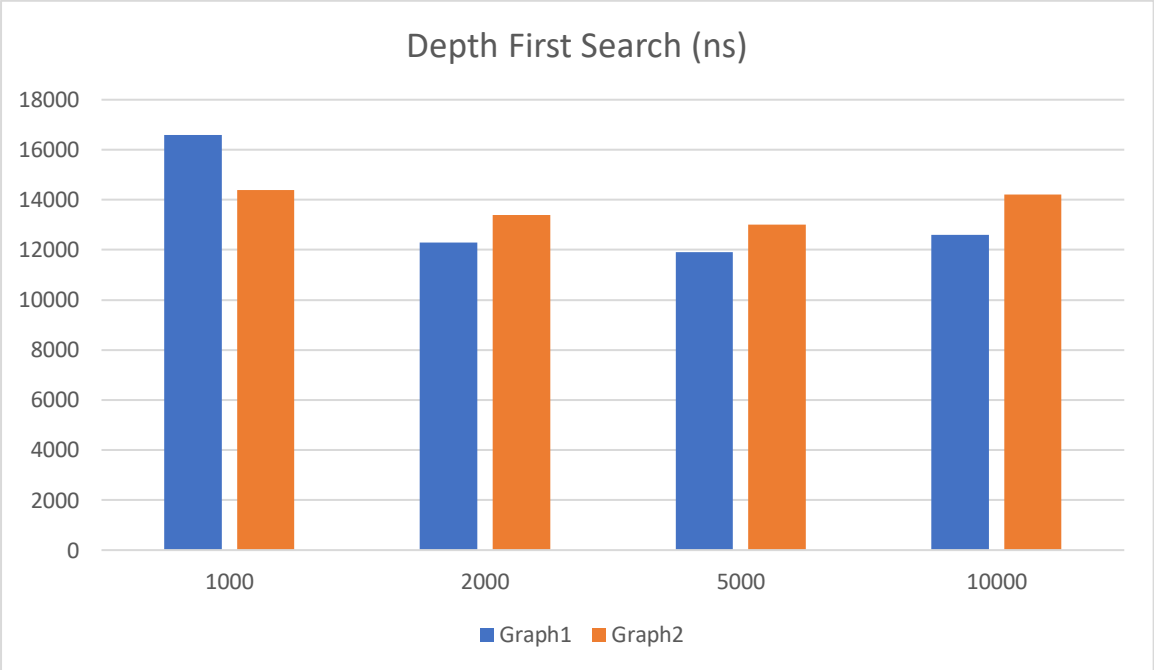
   Firstly, I created the graph interface and listGraph. Then I added the breadthFirstSearch nd depthFirstSearch class. In main test class, I created the random vertices for all different size of the graph. Then I measured the search time for all graphs. I used the list graph. Because in dense graph, list graph takes more running time and I can made an easier comparison.

4. TEST RESULTS:

```
***Breadth First Search***
Length 1000: 5815200 ns
Length 1000: 2186200 ns
Length 2000: 1067000 ns
Length 2000: 1123900 ns
Length 5000: 1662000 ns
Length 5000: 1896000 ns
Length 10 000: 3023300 ns
Length 10 000: 1938000 ns

***Depth First Search***
Length 1000: 16600 ns
Length 1000: 14400 ns
Length 2000: 12300 ns
Length 2000: 13400 ns
Length 5000: 11900 ns
Length 5000: 13000 ns
Length 10 000: 12600 ns
Length 10 000: 14200 ns
```





As seen in the graphs above, the depth frist search algorithm is faster than the breadth frist search algorithm. Because the graphs I create are "dense graph". If adjacency list is used in Dense graphs, time complexity increases. (I used this to better compare the data.) BFS is more suitable for searching vertices which are closer to the given source. DFS is more suitable when there are solutions away from source. Since the number of verteex in these graphs is

high, the breadthfisrtsearch algorithm takes longer. The depthFirstSearch algorithm, on the other hand, takes much less time than the other. Because it works with recursive. therefore, as the number of vertexes in the graph increases, the times for the depthFirstSearch algorithm become more efficient, so it should be used. Although this data may seem like a lot because it is in nanoseconds, in reality both run in O(|E|) time. In other words, the working speed changes according to the number of edges.