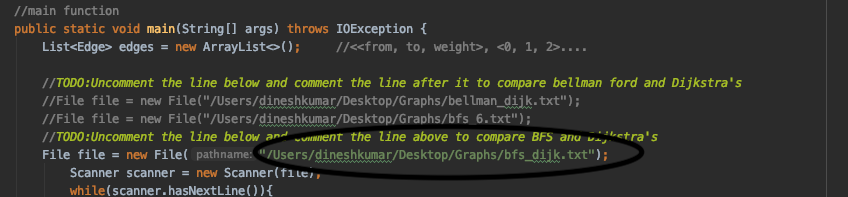
Steps to Run the project in Intellij IDE

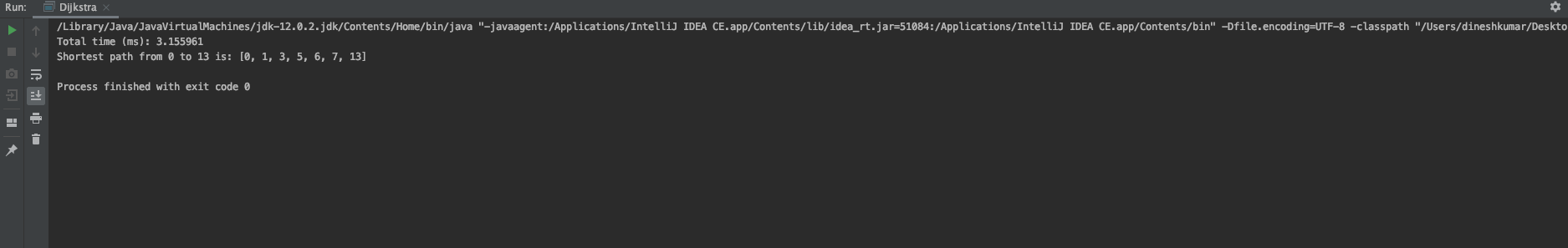
1.Unzip the project folder, go to ShortestPathAlgorithms directory and open the project in an IntelliJ IDE.

2.Go to the main function in each java class (BFS, Dijkstra and BellmanFord) and change the file path to point to your local directory.

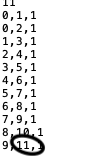


3.In order to test each input file, you need to change the text file name in the path in the main function. For example, to run "bfs\_6.txt", go to the file path in the main function and change the filename.

4.When you run the application for a specific class, the output will get printed out to the console. The output is the shortest path from source to destination vertex for the given graph file.

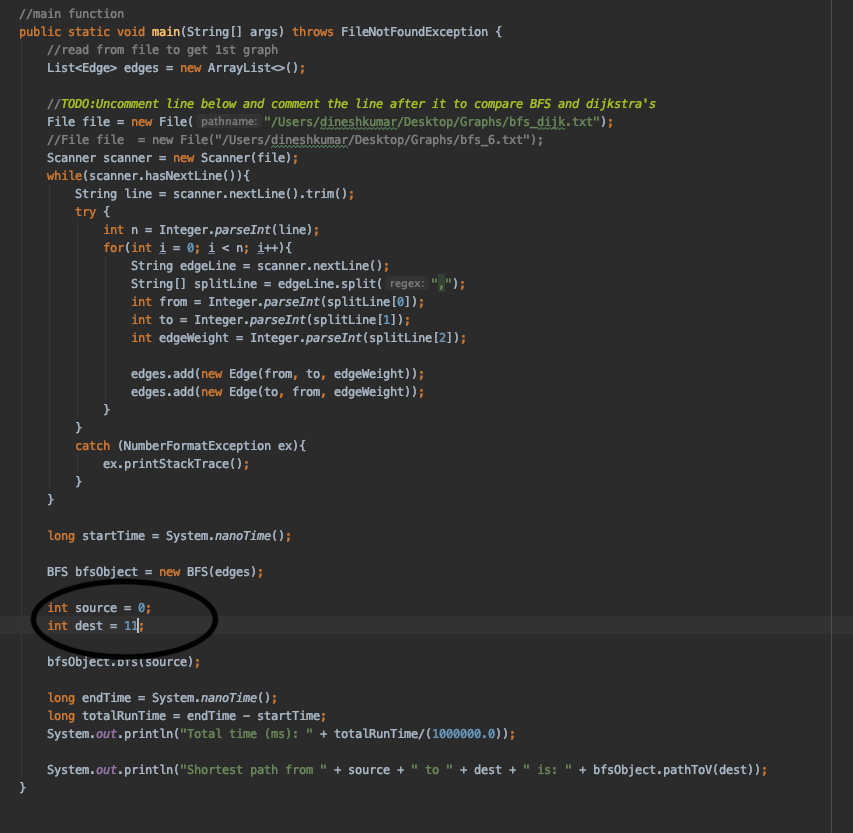


5.Whenever choosing to run the application for a different file, you need to make some changes in the main function to see the changes. Please enter the last vertex in the graph as the destination vertex. The algorithms work for finding shortest paths between any 2 vertices in the graph, but for best results for my project, my input graphs are written such that it shows us best results when run from source (first) to last vertex. To do this, open the corresponding input text file to find the last vertex and change the "dest" int field in the main function.



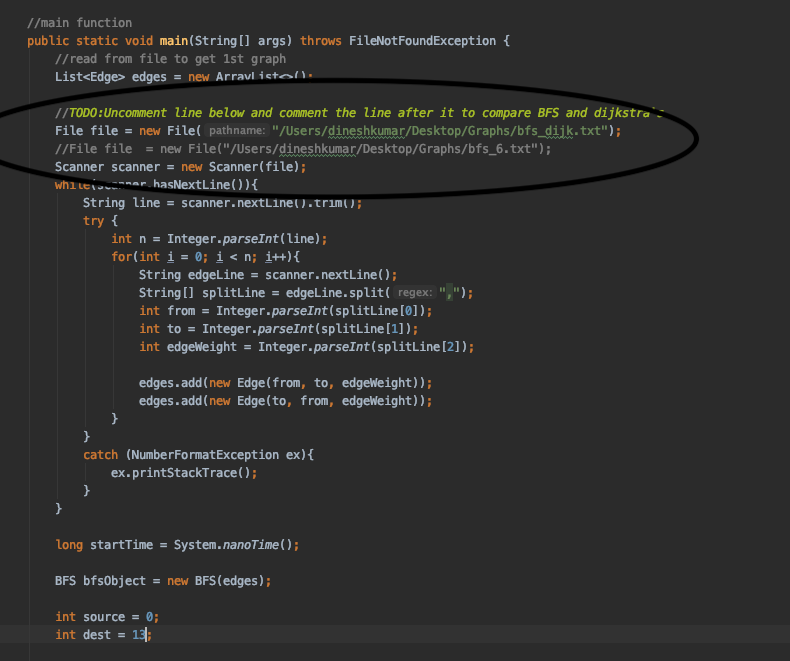
Last vertex in last line of input file

Modify "dest" field in the main function to reflect changes



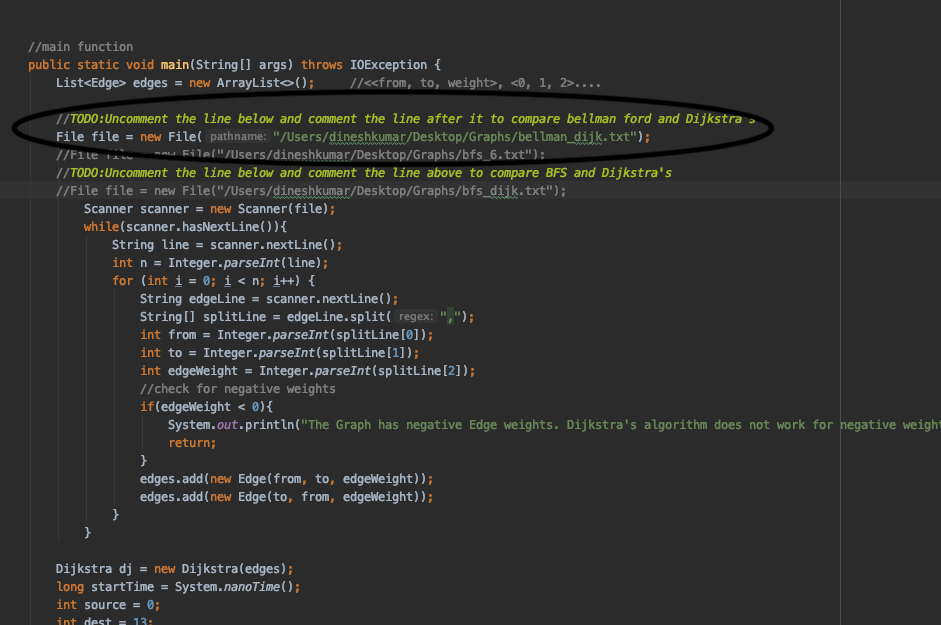
6.Using **"bfs\_dijk.txt"** to compare BFS and Dijkstra for shortest path cost:

* The pathname in the main functions have been set to run the text file for a specific algorithm. Inorder to compare the shortest path for BFS and Dijkstra's algorithm, comment out line in the main function where we read the default file, and uncomment this line shown below. Also change the "dest" field as shown above to reflect changes for each graph file.

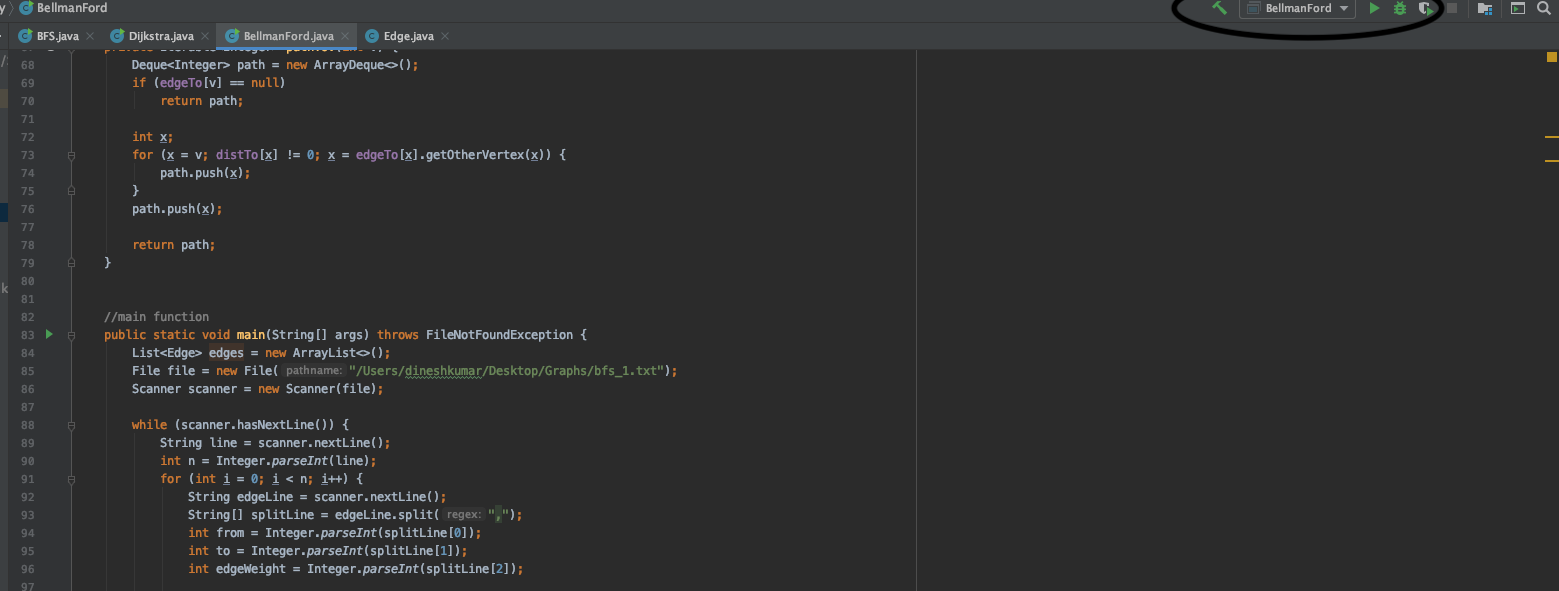


7.Using **"bellman\_dijk\_input.txt"** to compare Dijkstra and Bellman-Ford for Negative weighted graphs:

* Inorder to compare Dijkstra and Bellman-Ford for Negative weighted graphs, comment out line in the main function where we read the default file, and uncomment this line shown below. Also change the "dest" field as shown above to reflect changes for each graph file.



8.To test Bellman\_Ford algorithm, just run the application with the BellmanFord class selected.



9. I have used a "GraphGenerator" class that generates graphs for specified vertices and edges. I have not used this class for the input files added to submission. But I am planning to use that class during the Demo, so I have left that class as is in my Java project. I did not write this class, and I have used it from this source - <https://algs4.cs.princeton.edu/41graph/GraphGenerator.java.html>