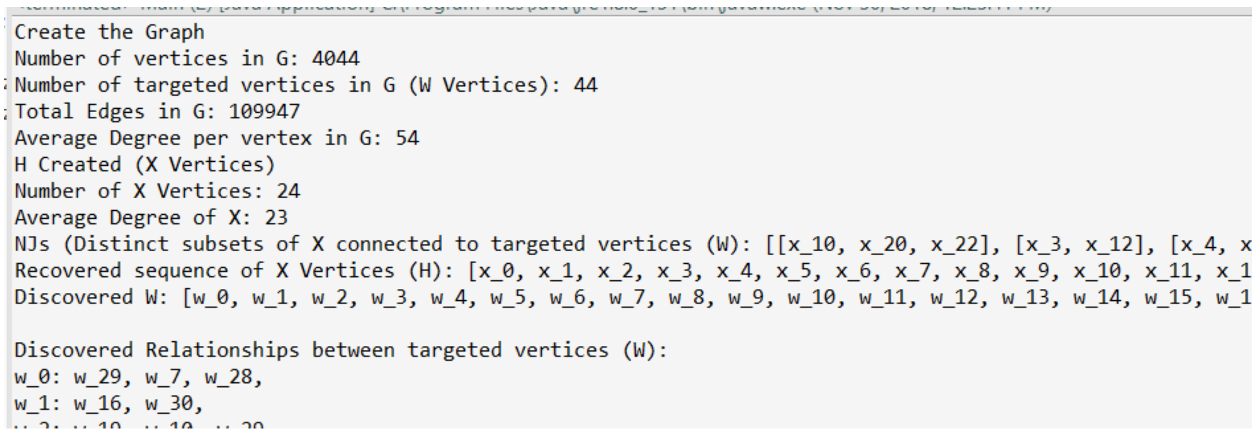
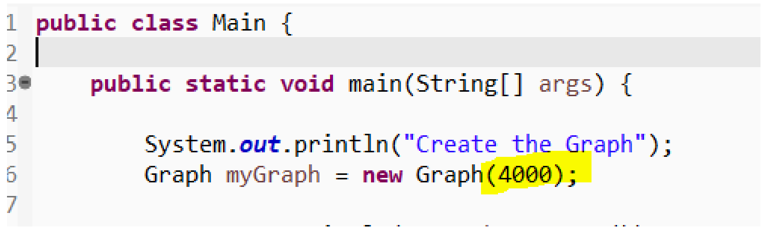
Running the Code

1. Open the source in any IDE that supports Java.
2. Run the source as is, and the console output will provide the results of the simulation.
   - The output will first provide information about the simulated graph G such as the number of vertices, number of targeted vertices, and the average degree per vertex.
   - Further information will be provided about graph H (i.e. the graph created by the vertices X – the fake accounts made by the adversary) including the number of vertices and the average degree of the X vertices.
   - The subsets made up of X vertices that connect H to the targeted vertices W are also printed. This shows that each W is connected to a distinct subset of the vertices in H. These subsets are called NJ to match the language in the paper.
   - When H is recovered, we print the sequence X vertices in H that we found. If no sequence is found (the algorithm fails), a display message will be printed indicating that.
   - The discovered targeted vertices (W) are also printed out. If a targeted vertex could not be recovered, then null will take its place in the printed list.
   - Finally, the discovered relationships between the targeted vertices (W) are printed out. This can be further compared with the expected relationships between the targeted vertices (W).



```
Create the Graph
Number of vertices in G: 4044
Number of targeted vertices in G (W Vertices): 44
Total Edges in G: 109947
Average Degree per vertex in G: 54
H Created (X Vertices)
Number of X Vertices: 24
Average Degree of X: 23
NJs (Distinct subsets of X connected to targeted vertices (W): [[x_10, x_20, x_22], [x_3, x_12], [x_4, x
Recovered sequence of X Vertices (H): [x_0, x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_10, x_11, x_1
Discovered W: [w_0, w_1, w_2, w_3, w_4, w_5, w_6, w_7, w_8, w_9, w_10, w_11, w_12, w_13, w_14, w_15, w_1

Discovered Relationships between targeted vertices (W):
w_0: w_29, w_7, w_28,
w_1: w_16, w_30,
```

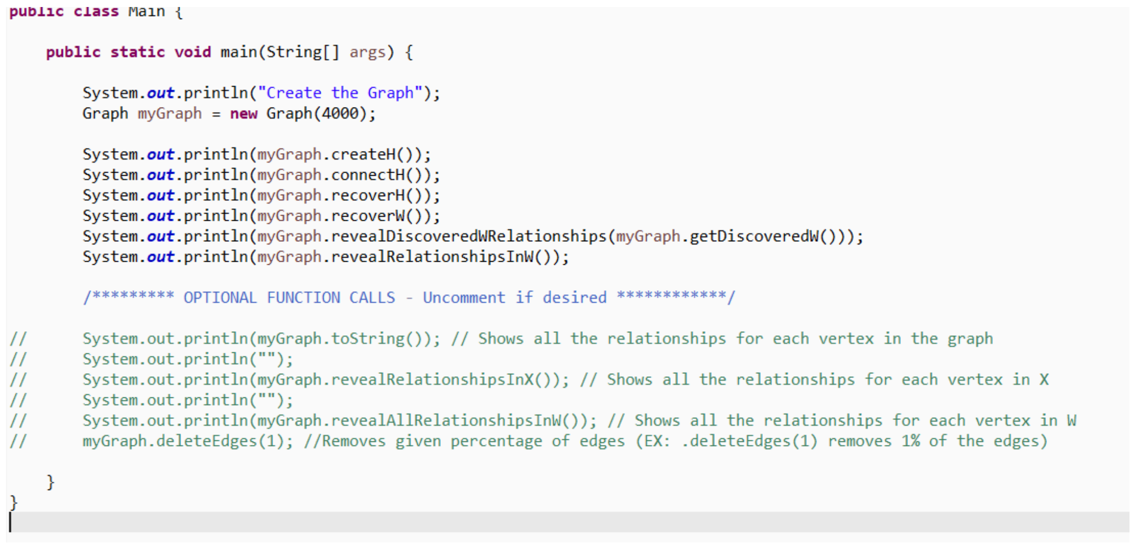
3. In the Main class, the number of vertices in the graph can be changed by altering the number passed into the Graph constructor if desired.



```java
public class Main {

    public static void main(String[] args) {

        System.out.println("Create the Graph");
        Graph myGraph = new Graph(4000);

```

4. In addition, there are optional function calls in Main that are commented out that will provide additional information.
   - The first function is to print the entire graph. This will show every vertex in the graph as well as it’s relationships with other vertices in the graph.
   - The second function is to show all of the relationships of the X vertices (graph H).

- The third function is to show all of the relationships of the W vertices.
- The fourth function allows the user to delete a percentage of edges of the complete graph. This was used in our experiments to observe the effectiveness of the algorithm with the scenario of edge deletion.



```
public class Main {

    public static void main(String[] args) {

        System.out.println("Create the Graph");
        Graph myGraph = new Graph(4000);

        System.out.println(myGraph.createH());
        System.out.println(myGraph.connectH());
        System.out.println(myGraph.recoverH());
        System.out.println(myGraph.recoverW());
        System.out.println(myGraph.revealDiscoveredWRelationships(myGraph.getDiscoveredW()));
        System.out.println(myGraph.revealRelationshipsInW());

        /********* OPTIONAL FUNCTION CALLS - Uncomment if desired ************/

//      System.out.println(myGraph.toString()); // Shows all the relationships for each vertex in the graph
//      System.out.println("");
//      System.out.println(myGraph.revealRelationshipsInX()); // Shows all the relationships for each vertex in X
//      System.out.println("");
//      System.out.println(myGraph.revealAllRelationshipsInW()); // Shows all the relationships for each vertex in W
//      myGraph.deleteEdges(1); //Removes given percentage of edges (EX: .deleteEdges(1) removes 1% of the edges)

    }
}
```