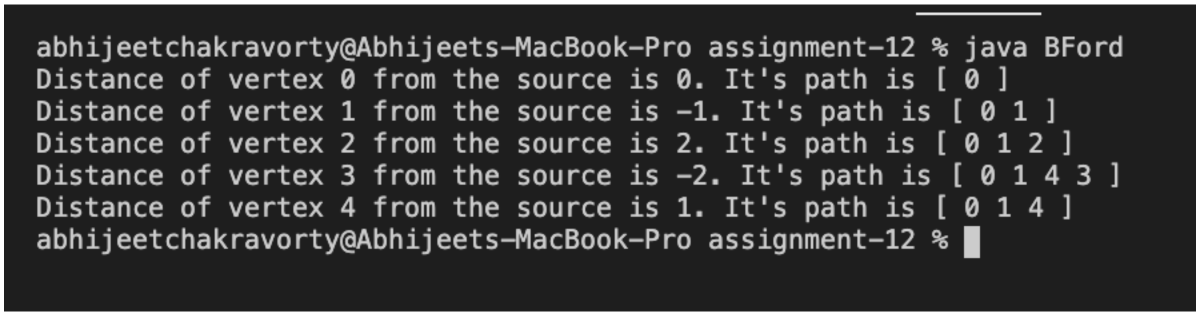
Q 1) Perform a Bellman Ford algorithm

Ans:

```
/* Bellman Ford Algorithm*/
import java.util.Arrays;
import java.util.List;
class EdgeData {
	int source, dest, weight;

	public EdgeData(int source, int dest, int weight) {
		this.source = source;
		this.dest = dest;
		this.weight = weight;
	}
}

public class BFord {
	static void printPathTraversal(int parent[], int vertex) {
		if (vertex < 0)
			return;

		printPathTraversal(parent, parent[vertex]);
		System.out.print(vertex + " ");
	}

	public static void bf(List < EdgeData > edges, int source, int N) {
		int distance[] = new int[N];
		int parent[] = new int[N];
		Arrays.fill(distance, Integer.MAX_VALUE);
		distance[source] = 0;
		Arrays.fill(parent, -1);
		for (int i = 0; i < N - 1; i++) {
			for (EdgeData edge: edges) {
				int u = edge.source;
				int v = edge.dest;
				int w = edge.weight;

				if (distance[u] + w < distance[v]) {
					distance[v] = distance[u] + w;
					parent[v] = u;
				}
			}
		}

		for (EdgeData edge: edges) {
			int u = edge.source;
			int v = edge.dest;
```

```
                int w = edge.weight;

                if (distance[u] + w < distance[v]) {
                    System.out.println("Negative Weight Cycle Found!");
                    return;
                }
            }

            for (int i = 0; i < N; i++) {
                System.out.print("Distance of vertex " + i + " from the " + "source is " +
distance[i] + ". It's path is [ ");
                printPathTraversal(parent, i);
                System.out.println("]");
            }
        }

        public static void main(String[] args) {
            List < EdgeData > edges = Arrays.asList(
                new EdgeData(0, 1, -1), new EdgeData(0, 2, 4),
                new EdgeData(1, 2, 3), new EdgeData(1, 3, 2),
                new EdgeData(1, 4, 2), new EdgeData(3, 2, 5),
                new EdgeData(3, 1, 1), new EdgeData(4, 3, -3)
            );

            final int N = 5;

            int source = 0;

            bf(edges, source, N);
        }
}
```

Output:



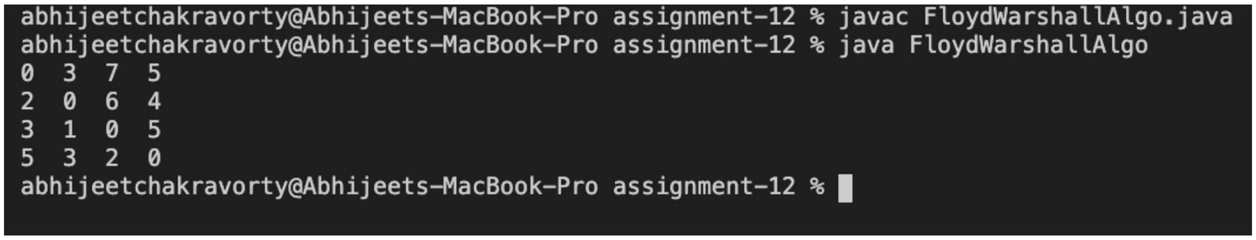
```
abhijeetchakravorty@Abhijeets-MacBook-Pro assignment-12 % java BFord
Distance of vertex 0 from the source is 0. It's path is [ 0 ]
Distance of vertex 1 from the source is -1. It's path is [ 0 1 ]
Distance of vertex 2 from the source is 2. It's path is [ 0 1 2 ]
Distance of vertex 3 from the source is -2. It's path is [ 0 1 4 3 ]
Distance of vertex 4 from the source is 1. It's path is [ 0 1 4 ]
abhijeetchakravorty@Abhijeets-MacBook-Pro assignment-12 %
```

Q 2) Perform a Floyd Warshall Algorithm.

Ans:

```
// Floyd Warshall Algorithm in Java
class FloydWarshallAlgo {
	final static int INF = 9999, nV = 4;

	// Implementing floyd warshall algorithm
	void floydWarshallAlgo(int graph[][]) {
		int matrix[][] = new int[nV][nV];
		int i, j, k;

		for (i = 0; i < nV; i++)
			for (j = 0; j < nV; j++)
				matrix[i][j] = graph[i][j];

		for (k = 0; k < nV; k++) {
			for (i = 0; i < nV; i++) {
				for (j = 0; j < nV; j++) {
					if (matrix[i][k] + matrix[k][j] < matrix[i][j])
						matrix[i][j] = matrix[i][k] + matrix[k][j];
				}
			}
		}
		printMatrix(matrix);
	}

	void printMatrix(int matrix[][]) {
		for (int i = 0; i < nV; ++i) {
			for (int j = 0; j < nV; ++j) {
				if (matrix[i][j] == INF)
					System.out.print("INF ");
				else
					System.out.print(matrix[i][j] + "  ");
			}
			System.out.println();
		}
	}

	public static void main(String[] args) {
		int graph[][] = {
			{
				0,
				3,
				INF,
                    5
                },
                {
                    2,
                    0,
                    INF,
                    4
                },
                {
                    INF,
                    1,
                    0,
                    INF
                },
                {
                    INF,
                    INF,
                    2,
                    0
                }
            };
            FloydWarshallAlgo a = new FloydWarshallAlgo();
            a.floydWarshallAlgo(graph);
        }
}
```

Output:



```
abhijeetchakravorty@Abhijeets-MacBook-Pro assignment-12 % javac FloydWarshallAlgo.java
abhijeetchakravorty@Abhijeets-MacBook-Pro assignment-12 % java FloydWarshallAlgo
0  3  7  5
2  0  6  4
3  1  0  5
5  3  2  0
abhijeetchakravorty@Abhijeets-MacBook-Pro assignment-12 %
```