Reverse the Directed Graph

Objective: Given a directed graph, write an algorithm to reverse the graph.
Example:

Approach:
Create a new graph with the same number of vertices. Traverse the given graph. Here we are using the adjacency list to represent the graph. Traverse each adjacency list and while traversing keep adding the reverse edges (making source as destination and destination as source). See the code for better understanding.
Code:

```java
import java.util.LinkedList;

public class ReverseGraph {
    static class Graph {
        int vertices;
        LinkedList<Integer>[] adjList;

        public Graph(int vertices) {
            this.vertices = vertices;

            adjList = new LinkedList[vertices];
            //Initialize lists
            for (int i = 0; i < vertices; i++) {
                adjList[i] = new LinkedList<>();
            }
        }

        public void addEdge(int source, int destination) {
            //add forward edge
            adjList[source].addFirst(destination);
        }

        public Graph reverse(Graph graph){
            Graph reverseGraph = new Graph(vertices);
            for (int i = 0; i <vertices ; i++) {
                LinkedList<Integer> nodeList = adjList[i];
                int source = i;
                for (int j = 0; j <nodeList.size() ; j++) {
                    int destination = nodeList.get(j);
                    //add reverse node
                    reverseGraph.addEdge(destination, source);
                }
            }
            return reverseGraph;
        }

        public void printGraph(){
            for (int i = 0; i <vertices ; i++) {
                LinkedList<Integer> nodeList = adjList[i];
                System.out.println("Vertex connected from vertex: "+i);

                for (int j = 0; j <nodeList.size() ; j++) {
                    System.out.print("->" + nodeList.get(j));
                }
                System.out.println();
            }
        }
    }


    public static void main(String[] args) {
        Graph graph = new Graph(5);

        graph.addEdge(0,1);
        graph.addEdge(1, 2);
        graph.addEdge(1, 3);
        graph.addEdge(2, 3);
        graph.addEdge(3, 4);
        graph.addEdge(4, 0);
        System.out.println("---------------Original Graph---------------------------");
        graph.printGraph();

        Graph reverseGraph = graph.reverse(graph);
        System.out.println("---------------Reverse Graph---------------------------");
        reverseGraph.printGraph();
    }
}
```

Output:

```
---------------Original Graph---------------------------
-

Vertex connected from vertex: 0

->1

Vertex connected from vertex: 1

->3->2

Vertex connected from vertex: 2

->3

Vertex connected from vertex: 3

->4

Vertex connected from vertex: 4

->0

---------------Reverse Graph----------------------------

Vertex connected from vertex: 0

->4

Vertex connected from vertex: 1

->0
```

```
Vertex connected from vertex: 2

->1

Vertex connected from vertex: 3

->2->1

Vertex connected from vertex: 4

->3
```