210CT Week 6 Coursework Tasks
Dr. Diana Hintea

LEARNING OUTCOMES

1. Understand the graph data structure.

BASIC/INTERMEDIATE TASKS (MANDATORY COURSEWORK TASKS)

1. Implement an unweighted and undirected graph data structure in the programming language of your choice, where the nodes consist of positive integers. You can either use an adjacency matrix or an adjacency list approach. The program should have functions for the following:
   a. Adding a node to the graph.
   b. Adding an edge to the graph.
   c. Printing the graph.

   Note: You must use Object Oriented Programming for this task.

ADVANCED TASK (OPTIONAL)

1. Implement the structure for an unweighted, undirected graph G, where nodes consist of positive integers. You can either use an adjacency matrix or an adjacency list approach. You must use Object Oriented Programming for this task. In addition, implement a function called isConnected() to check whether or not the graph is strongly connected. Output Yes or True if the graph is connected and No or False it the graph is not connected.
2. Adapt the previous data structure for a weighted, directed graph G, where nodes consist of positive integers. You can use either an adjacency matrix or an adjacency list approach. You must use Object Oriented Programming for this task. In addition, implement a function called findLongestSimplePath() which will return the longest cost simple path. Assume that the graph is a Directed Acyclic Graph (DAG).

READING

Felzenszwalb, P. and Zahib, R. (2010). Dynamic Programming and Graph Algorithms in Computer Vision.