Laboratory 3

def **getMinCostPathBellman**(*self*, g, s, t):

*'''*

*Return the minimum cost in a directed graph.Also determine*

*if there is a cycle*

*input: g - graph adjacency list*

*s - source vertex*

*t - target vertex*

*output: return a list of vertices representing the graph*

*print a message in case of a negative cycle or*

*if there is no walk between 2 vertices*

*'''*

def **getPath**(*self*, s, t, prev):

*"""*

*Helper function, recove the path from a*

*dictionary in which we store the predecessor*

*input: s -source vertex*

*t - target vertex*

*"""*