Complete these questions as part of your final exam. Submit them through blackboard by the date posted.

TH1. Implement a graph using the graph interface below.

interface Graph<T>

{

// Return the number of vertices

int nodeCount();

// Return the current number of edges

int edgeCount();

// Get the value of node with index v

T getValue(int v);

// Set the value of node with index v

void setValue(int v, T val);

// Adds a new edge from node v to node w with weight wgt

void addEdge(int v, int w, int wgt);

// Get the weight value for an edge

int weight(int v, int w);

// Removes the edge from the graph.

void removeEdge(int v, int w);

// Returns true iff the graph has the edge

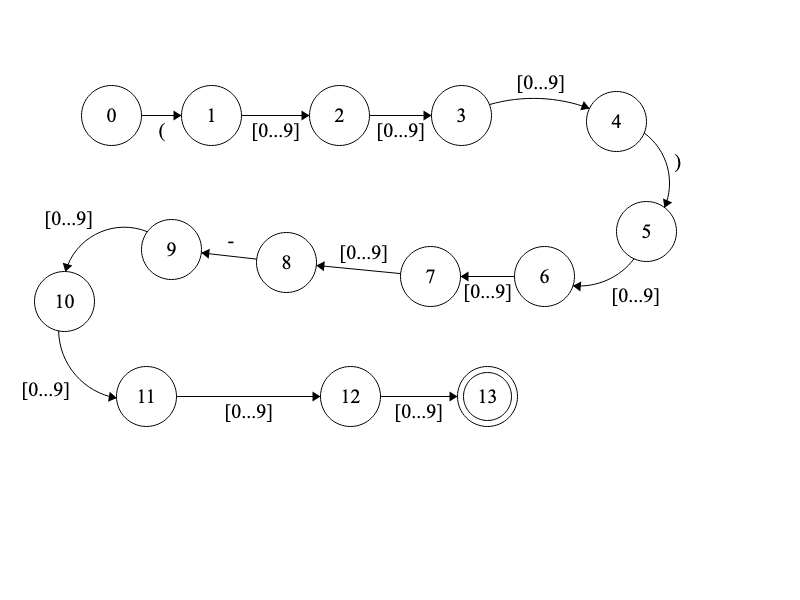
boolean hasEdge(int v, int w);

// Returns an array containing the indicies of the neighbors of v

int[] neighbors(int v);

}

TH2. Using your graph implementation create a class called FSM (finite state machine) – you can extend the graph or use a composition. Create an instance of FSM that models the inset graph.



A FSM is a kind of directed graph. It has a starting vertex, it has at least one acceptance vertex, and may have one or more rejection vertices. An input is given and processed by the machine. If the machine processes all input and remains in the acceptance state then the input is accepted. Otherwise the input is rejected. In the case of the FSM the weights on the graph edges are the value that triggers that transition. More information on FSM can be found here - <https://www.javatpoint.com/finite-state-machine> and here - <https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-01sc-introduction-to-electrical-engineering-and-computer-science-i-spring-2011/unit-1-software-engineering/state-machines/MIT6_01SCS11_chap04.pdf>