CGraphPrimitive

Fields

This abstract class contains primitive operations and fields common to Nodes, Edges and Graphs

m\_graph: A reference to the graph owning the element

M\_SerialNumber : A unique number identifying the element

M\_ElementType : Type of element can be any of the three types : node, edge, graph

M\_Label : Each element has a label which can be set either centrally from the CGraph, or locally using the SetLabel() method <see cref="SetLabel"/> or automatically given a default label consisting of the serial number a prefix indicating the type of element ( i.e for the case of nodes <see cref="CGraphNode.ToString()"/>. This property is realized in the descentant classes for nodes, edges and graphs correspondingly

Locally defined labels must be verified from the specialized class of Labeller objects.

1. Default labels must be created locally to the object and verified from the labeller class which must have the labels of the of the graph nodes or edges.

m\_algorithmicOutput : Each element may store information under a specific integer key. The information is easily available through an integer given the integer key

CGraphNode: CGraphPrimitive

Fields

The CGraphnode class models the nodes of the graph. It contains the appropriate operations and fields.

m\_OutgoingEdges: A list containing the node’s outgoing edges

m\_IngoingEdges: A list containing the node’s ingoing edges

m\_Successors : A list containing references to the successor of this node

m\_Predecessors : A list containing references to the predecessor of this node

m\_nodeSerialNumber: Each node has a unique serial number. It corresponds to the sequence in which the node is added to the graph

Methods

Internal Interface

void AddOutgoingGraphEdge(CGraphEdge edge) : It is called by the CGraph object while inserting an edge to the graph. It updates the relevant member variables of the class for the current node.

void RemoveOutgoingGraphEdge(CGraphEdge edge) : Removes the outgoing graph edge. It is called by the CGraph object while deleting an edge to the graph. It updates the relevant member variables of the class for the current node

void AddIngoingGraphEdge(CGraphEdge edge) : It is called by the CGraph object while inserting an edge to the graph. It updates the relevant member variables of the class

void RemoveIngoingGraphEdge(CGraphEdge edge): Removes the ingoing graph edge. It is called by the CGraph object while deleting an edge to the graph. It updates the relevant member variables of the class

void SetLabelContext(CGraphLabeling<CGraphNode> labeler) : Sets the label for the current node using the name specified in the given labeller. After this call the node exposes the given label from the M\_Label property. This function is better to be called for all graph nodes in order to avoid complexity and inconsistency. For this reason it is an internal method. To do this call the method void CGraph::SetNodeLabelContext(object labelContructor=null)

Public Interface

CGraphNode Successor(int index) : Returns the i-th successor in the list

CGraphNode Predeccessor(int index) : Returns the i-th predecessor in the list

string ToString(object labeler) : Returns a string that represents the label of the node given by the specifier labeler object

override string ToString() : It returns the current label of the node. The node label can be specified by the user

CGraphEdge: CGraphPrimitive

Fields

m\_sourceNode : A reference to the source edge node

m\_sinkNode : A reference to the sink edge node

m\_edgeType : The edge can be any of the two types directed or non-directed

m\_edgeSerialNumber : Each edge has a unique serial number. It corresponds to the sequence in which the edge is added to the graph

CGraph: CGraphPrimitive

Models a graph and has exclusively the methods to construct it.

Fields

m\_graphNodes : A list containing all graph nodes

m\_graphEdges : A list containing all graph edges

m\_graphType : Depending on its edges type the graph may belong to any of the following categories : directed, undirected mixed

m\_NodeLabels: The graph nodes can be labelled by different code contexts which are objects used as keys to get access to the labelling objects of nodes

m\_EdgeLabels: The graph edges can be labelled by different code contexts which are objects used as keys to get access to the labelling objects of edges

m\_graphSerialNumber : Each graph has a unique serial number.