The ADT ‘GRAPH’

```
module GRAPH ( module Graph, graph, ugraph, outNeighbours, start, finish ) where

import Graph

--------------------------------------------------------------------------------
-- I N T E R F A C E  :  P U B L I C  :  all exports of module ’Graph’, plus :
--------------------------------------------------------------------------------

-- graph sns ses : the directed graph formed from
--                    all nodes corresponding to
--                       the list of strings ’sns’
--                    all edges corresponding to
--                       the list of 2-tuples of strings ’ses’

graph :: [ String ] -> [ ( String, String ) ] -> Graph

--------------------------------------------------------------------------------

-- ugraph sns ses : the undirected graph formed from
--                     all nodes corresponding to
--                        the list of strings ’sns’
--                     all edges corresponding to
--                        the list of 2-tuples of strings ’ses’

ugraph :: [ String ] -> [ ( String, String ) ] -> Graph

--------------------------------------------------------------------------------

-- outNeighbours n g : a list of the nodes reachable along a single out-edge
--                     from node ’n’ in graph ’g’

outNeighbours :: Node -> Graph -> [ Node ]

--------------------------------------------------------------------------------

-- start e : the start node of edge ’e’

start :: Edge -> Node

--------------------------------------------------------------------------------

-- finish e : the finish node of edge ’e’

finish :: Edge -> Node

--------------------------------------------------------------------------------
```

```
--------------------------------------------------------------------------------
-- I M P L E M E N T A T I O N  :  P R I V A T E
--------------------------------------------------------------------------------

graph sns ses = insertEdges [ ns2e ( s2n s1, s2n s2 ) | ( s1, s2 ) <- ses ]
                            ( insertNodes [ s2n s | s <- sns ] emptyGraph )

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

-- insertNodes ns g : the graph formed by inserting all nodes in the list ’ns’
--                    into graph ’g’

insertNodes :: [ Node ] -> Graph -> Graph

insertNodes ns g = foldr insertNode g ns

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

-- insertEdges es g : the graph formed by inserting all edges in the list ’es’
--                    into graph ’g’

insertEdges :: [ Edge ] -> Graph -> Graph

insertEdges es g = foldr insertEdge g es

--------------------------------------------------------------------------------

ugraph sns ses = graph sns ( foldr ( \( s, f ) -> \acc ->
                                          ( s, f ) : ( f, s ) : acc )
                                   [ ]
                                   ses )

--------------------------------------------------------------------------------

outNeighbours n g = [ finish e | e <- outEdges n g ]

--------------------------------------------------------------------------------

start e = fst ( e2ns e )

--------------------------------------------------------------------------------

finish e = snd ( e2ns e )

--------------------------------------------------------------------------------
```