11.3 Graph example

Description

In this section, we show a further example of using virtual class attributes from the domain of graphs. Graphs have many applications. Here we sketch the structure of a graph of cities connected with roads. Such a graph may e.g. be used for tour planning. We also show a very brief sketch of a graph for computer networks connected with cables.

We start by defining a class defining an overall structure of a graph:

```
class Graph:
   class Node(id: var String):<
      addEdge(to: ref Node)-> E: ref Edge:<
         :::
      :::
   class Edge(from: ref Node, to: ref Node):<
      :::
   nodes: obj Set(#Node)
   addNode(nm: var String) -> n: ref Node:
      :::
   display: ...
   :::
```

Class `Graph` has the following attributes:

- Two virtual classes `Node` and `Edge` representing the nodes and edges of the graph. They will be further extended in subclasses of `Graph` as we shall se below.
- An object `nodes`, which contains the set of nodes of a given graph.
- A method `addNode` for adding a `Node` to a given `Graph`.
- A method `display` for displaying the graph.

Class `Node` has a method `addEdge` for adding an edge from the `Node` to a node represented by the parameter `to`.

The `Node` class has the structure:

```
class Node(id: var String):<
   addEdge(to: ref Node)-> e: ref Edge:<
      e := Edge(this(Node),to)
      inner(addEdge)
      edges.insert(e)
   edges: obj Set(#Edge)
   display:< ...
      displayEdges:< ...
```

- The parameter `id` represents the id/name of the `Node`.
- The method `addEdge` adds an `Edge` to the `Node`.
- The object `edges` holds all the edges from the `Node`.
- The method displays the `Node` and `displayEdges` display the edges of the `Node`.

In this simple example, the `Edge` class just has a `display` method, which invokes `inner`.

```
class Edge(from: ref Node, to: ref Node):<
      display:<
         inner(display)
```