CS3319 Lab 3 Fall 2013 Burris

Due: Friday November 8, 2013
"C" Option (best grade is 70):

Implement the simple version of the topological sort algorithm. Sort the following relations: 3<4, 3<4, 11 < 9, 6<1, 8<7, 10<9, 8<3, 6<7, 8<10, 2<3, 5<4, 2<7, 5<1, 6<9, 11 < 1and 5<6. Note that the data contains duplicate relations. Your program should not be affected by duplicate data except for additional run time and space. Your output should clearly indicate if no solution exists.

Now process the following relations: 11 < 1, 6<1, 8<7, 10<9, 6<9, 3<4, 6<7, 8<10, 2<3, 5<4, 8<3, 2<7, 5<1, 5<6, 4<8, 3 < 5, 2<5, 11 < 9, 1 < 6 and 1<2.

"B" Option(best grade is 80):

Implement the topological sort algorithm. If a loop (no solution) is encountered, print the actions that make up the loop. Process the data for the "C" option. You need not implement the "C" Option program. You must get at least one loop. Did you get all possible loops? If you can find multiple (preferably all) loops, brag on yourself. It is worth a celebration.

"A" Option(best grade is 90):

Allow the user to specify actions using any programmer defined <u>enumeration</u> data type they desire using generic instantiation. Process the following data in addition to the "C" option data. A partial specification follows. The "A" option must print the contents of a loop if encountered. **You must use OOP to receive credit utilizing generics** (templates).

Additional Data: Garza < Taylor, Funk < Taylor, Sido < Dedear, Rocha < McLeod, McLeod < Dedear, Taylor < Sido, Funk < Sido, Reioux < Dedear, Davies < Funk, Funk < Dedear, and Funk < Taylor.

Additional Data: Garza < Taylor, Funk < Taylor, Sido < Dedear, Rocha < McLeod, McLeod < Dedear, Taylor < Sido, Funk < Sido, Reioux < Dedear, Davies < Funk, Funk < Dedear, Funk < Taylor, and Dedear < Funk.

Hint for A option: See the generic circular queue pages 54-55 (approximately) of DataStructuresPgms.doc. In the program notes a method, GIOEX, is demonstrated to pass generic I/O routines to a generic package. Assume a partial order of the form JobA < JobB read as JobA precedes JobB. <u>To receive full credit you must pass the I/O routine to the sort program (which does the printing)</u>.

General Structure:

```
generic  -- You may modify this as required but observe the spirit.
        type SortElement is private;  -- An element J (or K) of the partial ordering
        -- J < K processed by the topological sort.  J and K represent jobs in the partial ordering.

        with procedure get(Job:  out SortElement);  // Reads J or K.
        with put(Job:  in SortElement);  // Print the value of J or K.

package GenericTopologicalSort is
        TopologicalSort;
        --  additional procedures/functions to export if required
end GenericTopologicalSort;

package body GenericTopologicalSort is
        -- This should read (get) the relations and print (put) the results.
        type Node;
        type NodePointer is access Node;
        type Node is tagged record
                Suc:     SortElement;
                Next:    NodePointer;
        end record;

        type JobElement is record
                Count:  Integer := 0;
                Top:     NodePointer;
        end record;

        SortStructure:  Array(SortElement) of JobElement;
        -- other declarations

        procedure TopologicalSort is
        begin -- Program to obtain the relations in the partial ordering,
                -- sort the jobs, and print results;
        end TopologicalSort;
end GenericTopologicalSort;

with GenericTopologicalSort;
procedure Main is

        type NameType is (Mary, Joe, Tom, Bob, Sara, Julie, Larry, Sam);

        package NameTypeIO is new Ada.Text_IO.Enumeration_IO(NameType);
        use NameTypeIO;

        -- Overload definitions for sRocha parameter "get(Action : out SortElement)"
        -- and "put(Action: in SortElement)" for NameTypeIO.

        package NameTopologicalSort is new
                GenericTopologicalSort(NameType, get, put);
        use NameTopologicalSort;
begin
-- rest of program
end Main;
```

"A+" Option(best grade is 100):

Implement the "A" option. You must use the count field to build the linked queue, not a separate array.