COMP 280 : Assignment 5

due: Thursday, February 24, 2000

1. (2 pts) Does every transitive graph contain cycles? Either prove that they do or provide two transitive graphs, one with cycles and one without cycles.

2. (2 pts) Prove or provide a counterexample for the following statement:

   no graph is both symmetric and anti-symmetric

3. (6 pts) Recall the definition of the relation *PathInR*, which we defined relative to a relation R from $A \rightarrow A$ as follows:

   - $\langle a, b\rangle \in R \rightarrow \langle a, b\rangle \in \mathit{PathInR}$
   - $\langle b, c\rangle \in R \wedge \langle a, b\rangle \in \mathit{PathInR} \rightarrow \langle a, c\rangle \in \mathit{PathInR}$

   For each of the following properties of relations, indicate the conditions (relative to R) under which *PathInR* has that property:

   (a) Reflexive

   (b) Symmetric

   (c) Transitive

4. (3 pts) Given a relation R from A to A, the relation *ReachInSteps* from numbers to tuples in $A \times A$ indicates when the graph of R contains paths of particular lengths between particular elements of A. For example, given the relation

   $$R_1 = \{\langle a, b\rangle, \langle b, d\rangle, \langle a, d\rangle, \langle a, c\rangle, \langle c, e\rangle, \langle d, e\rangle\}$$

   ReachInSteps would be

   $$\{\langle 1, \langle a, b\rangle\rangle, \langle 1, \langle a, d\rangle\rangle, \langle 1, \langle a, c\rangle\rangle, \langle 1, \langle b, d\rangle\rangle, \langle 1, \langle d, e\rangle\rangle, \langle 1, \langle c, e\rangle\rangle, \langle 2, \langle a, d\rangle\rangle, \langle 2, \langle a, e\rangle\rangle, \langle 2, \langle b, e\rangle\rangle, \langle 3, \langle a, e\rangle\rangle\}$$

   For an arbitrary relation R, define *ReachInSteps* in terms of R.

5. (6 pts) The CS department wants to create a new website. The website must display information about faculty, research groups, students, and courses. Faculty belong to research groups and teach courses. Students belong to their faculty advisors' research groups.

   (a) Develop a data model for this information. Make your model as precise as possible. Remember to explicitly state all properties that should hold of your model.

   (b) Develop a relation which defines when links exist from one page to another. Your relation should contain:

   - Links from faculty to the courses that they teach
   - Links from courses to the faculty who teach them
   - Links from students to their faculty advisors
   - Links from faculty to their research groups
   - Links from research groups to all faculty and students in the group
   - Links from research groups to the department home page

   (c) The department suggests putting links to courses on the home page, but not links to faculty, students, or research groups. Under what conditions is every student's page accessible from the department home page under this organization? Justify your answer. (Note: there may be more than one set of conditions – state and justify each one separately.)

6. (5 pts) Given the relation defining the links that should exist in a website, we want to use programs to generate the website according to that relation. The following code proposes two programs, *gen-1* and *gen-2*, for producing websites from relations. The programs use several helper functions as described below. The helper function *produce-page* writes each page to a file and returns some information indicating the filename to which it wrote the page. While writing a page, *produce-page* uses the given *pagenameinfo* to generates links. If asked to create a link to a page which has not yet been generated, *produce-page* will generate an error.

   Describe the kinds of relations that each of *gen-1* and *gen-2* can generate pages for without *produce-page* resulting in an error.

```
;; rel[α] is a binary relation from α to α
;;
;; find-next : page rel → (listof page)
;; returns a list of all range pages in given relation that have page as the domain element
;;
;; produce-page : page pagenameinfo → pagenamedata
;; write the page to file and return information about the page's location
;;
;; combine : pagenamedata pagenameinfo → pagenameinfo
;; add the new pagenamedata to the pagenameinfo
;;
;; gen-1 : page rel[page] → true
;; traverses links-rel from home-page and produces all pages
;; the helper function consumes a list of pages to generate and info about page locations
(define (gen-1 home-page links-rel)
  (local [(define (gen-help gen-pages page-names)
            (cond [(empty? gen-pages) true]
                  [else
                   (gen-help (append (find-next (first gen-pages) links-rel)
                                     (rest gen-pages))
                             (combine (produce-page (first gen-pages) page-names)
                                      page-names))]))]
    (gen-help (list home-page) empty)))
;;
;; gen-2 : page rel[page] → true
;; traverses links-rel from home-page and produces all pages
;; the helper function consumes a list of pages to generate and info about page locations
(define (gen-2 home-page links-rel)
  (local [(define (gen-help gen-pages page-names)
            (cond [(empty? gen-pages) true]
                  [else
                   (gen-help (append (rest gen-pages)
                                     (find-next (first gen-pages) links-rel))
                             (combine (produce-page (first gen-pages) page-names)
                                      page-names))]))]
    (gen-help (list home-page) empty)))
```