Bashir Ogbunji 2005

UNIVERSITY OF SASKATCHEWAN
DEPARTMENT OF COMPUTER SCIENCE

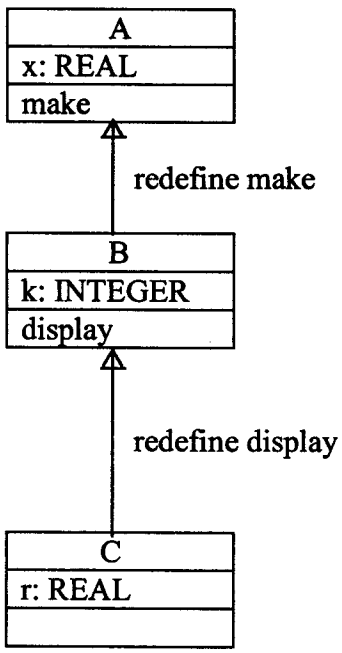
CMPT 250.6
Final Examination

3 Hours **Closed Book** **April 18, 2005**
Marks

This exam is out of 150. Thus, you have 1 minute per mark with 30 minutes to spare. Use this in judging how much time to spend on a question.

12 1. Consider the inheritance taxonomy shown in the diagram. Note that class B inherits from class A with procedure *make* redefined, and class C inherits from class B with the procedure *display* redefined. Suppose that the following local variables exist in some routine, and the following instructions are to be part of the code:

```
local
      a: A
      b: B
      c: C
do
      create a.make
      create b.make
      create c.make
```



a) Which of the following are valid?

```
a := b
c := b
```

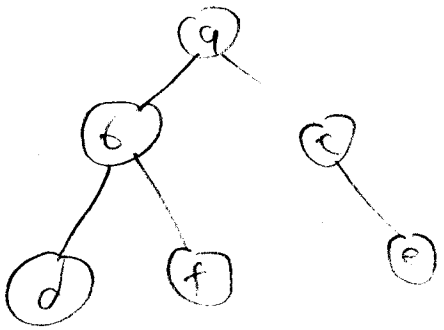
b) Assuming that the valid instructions from part (a) have been executed, which of the following are valid? If an instruction is valid, which version of the routine is called?

```
a.make
a.display
c.display
```

c) Which of the two variables a and c is polymorphic, i.e., can have many forms? Why?

10 2. Give the expression for the time requirements of the function *d* below. It calls the functions *f(k)*, a BOOLEAN function with time requirements $O(m^2)$, m independent of k
g(k), an INTEGER function with time requirements O(n), n independent of k
and *h(k)*, an INTEGER function with time requirements O(m), m independent of k

```
d : INTEGER is
local j : INTEGER
do
    from
        j := 1
        Result := 0
    until j > p or else f(j)
    loop
        Result := Result + g(j)*h(p-j)
        j := 2*j
    end
end
```



15 3. Use any one of the three approaches taught in class to provide a formal specification for a bounded queue ADT. **Clearly indicate** which of the three approaches you are using and specify the following operations: *create*, *insert*, *delete*, *front*, and *is-empty*. The *front* operation returns the element at the front of the queue.

15 4. Develop the algorithm for deletion from a weight balanced tree. The time requirements for this algorithm are proportional to what?

6 5. When working with a file on disk, a buffer is normally used. What is a buffer? Why is a buffer used?

12 6. Prove by induction the following:

$$1^3 + 2^3 + \ldots + n^3 = [n(n+1)/2]^2$$

15 7. A hypothetical insurance premium program computes annual car insurance premiums based on two parameters: the policyholder's age and driving record as follows –

premium := (*base_rate* * *age_multiplier*) - *safe_driving_reduction*

The *age_multiplier* is a function of the policyholder's age. The *safe_driving_reduction* is given when the *current_points* (assigned by traffic courts for moving violations) on the policyholder's driver licence are below an age-related cutoff. The *base_rate* changes from time to time; it is currently $1000 for an annual premium.

The following table is used to compute premiums:

age_range	age_multiplier	points_cutoff	safe_driving_reduction
$16 \le age < 26$	2.9	7	$75
$26 \le age < 46$	1.0	4	$125
$46 \le age < 70$	1.3	6	$150
$70 \le age$	1.5	8	$175

For example, for a policyholder of age 70 with a *current_points* value of 6,
the premium is: $1000 * 1.5 – 175 = $1325

The input in this example is as follows:
age = 70 *current_points* = 6

Using black-box testing techniques, formulate test cases for this program.

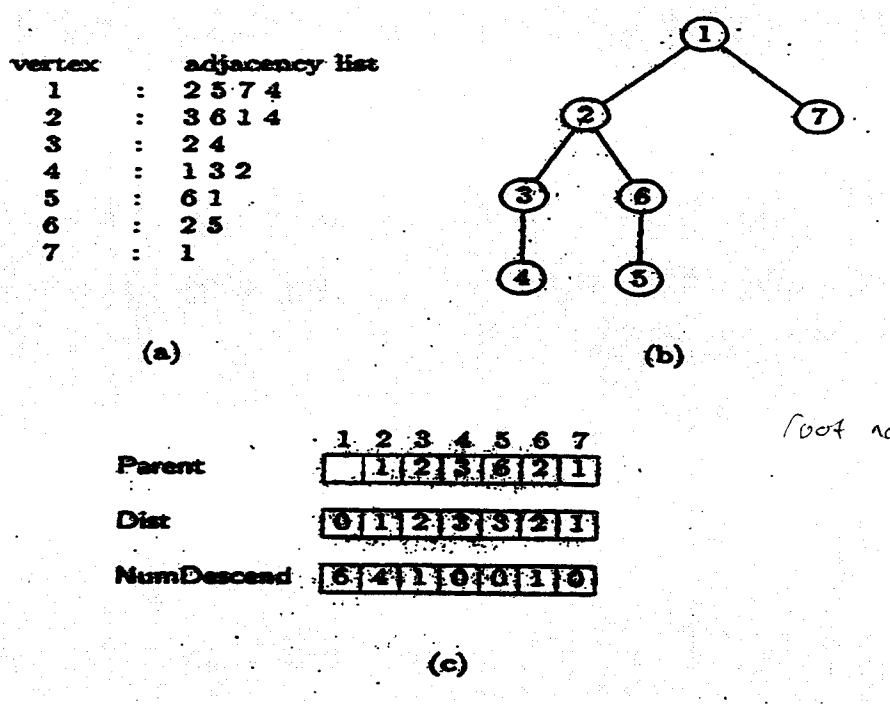
15 8. Formulate an algorithm or Eiffel routine based on a depth-first search to construct the DFS (depth-first search) tree T starting a vertex v_0. For example, the adjacency lists for an undirected graph in the Figure (a) have the associated DFS tree in Figure (b), which is obtained from a depth-first search starting at vertex 1. Your algorithm is to generate the following arrays to represent the DFS tree T.

Parent[u]: where $u \neq v_0$ denotes the parent of u in T

Dist[u]: stores the distance of vertex u from vertex v_0 in T

NumDescend[u]: stores the number of descendants of u in T

The arrays for the example DFS tree are given in Figure (c).



root node has parent 0

50 9. A hospital system for patient intake is to be developed. The reception office registers all patients when they are admitted and assigns each of them to a particular hospital ward (that each contains several beds) and a particular surgeon. The surgeon arranges surgery for his or her own patients.

A ward has many inpatients. An inpatient may undergo many instances of surgery. A surgeon may perform many instances of surgery and an operating theatre may be the site of many instances of surgeries. An instance of surgery can occur in one, and only one, theatre, and there must be a theatre available before there can be instances of surgery.

A ward contains nurses and beds. Nurses are assigned to one ward. There are several types of wards in the hospital: Maternity, Cardiac, Intensive Care, Skin diseases (contagious and non-contagious), Burn, etc..

The state of a patient object has a life history, from creation to destruction. The following table denotes some of the significant events that might take place in the life of an inpatient.

An inpatient object state table for a hospital system.

State	Event	Action	New State
Created	Patient assigned to ward	--	Admitted
Admitted	Surgery booked	Set notification timer running	Operation_Scheduled
Operation_Scheduled	Operation complete	Notify ward nurse	Recovery
Recovery	Fully recovered	Notify reception	Discharged
Discharged	Delete	Remove records	Terminated
Terminated	--	--	--

The table shows the list of allowable states for the object (Created, Admitted, Operation_Scheduled, Recovery, Discharged, and Terminated in this case). For example, a new instance of Inpatient may be created at some time. When a new patient is received at the hospital reception desk, that patient is assigned to a hospital ward and an appropriate event (or message) is sent to the idle Inpatient object, causing its state to change from Created to Admitted.

Wards are created, assigned nurses to them, and contain Beds, each of which is either unoccupied or has an Inpatient.

Suitable user interfaces must be provided for Nurses, Surgeons, Hospital Administrators, and Theatre Administrators.

Inpatients can be sponsored by Saskatchewan Health or privately by a Worker's Compensation Board or an Insurance Company. Inpatient invoices are sent to the Sponsors and Payments are received by the Hospital Business Office.

A Theatre Administrator schedules Surgeries in each theatre. Each Surgery takes some period of time.

Many reports that are to be produced for various users of the system include:

- Daily new Inpatient listing
- Daily Discharge of Inpatients
- Daily scheduled Surgeries
- Weekly theatre open time slots that are available for surgery
- Total monthly invoices
- Total monthly payments
- Surgeon daily surgery schedule
- Available beds by Ward
- History of Inpatients stays at the hospital when discharged, including what surgeries she or he may have had during the stays.

You are to design an object-oriented software system for this problem. The deliverables consist of the following:

10 (a) complete context model diagram with all external (direct and indirect) entities and all events/flows shown.

15 (b) number of sequence or collaboration diagrams involving events/actions that takes an Inpatient from its initial created state in the above state table to the Terminated state.

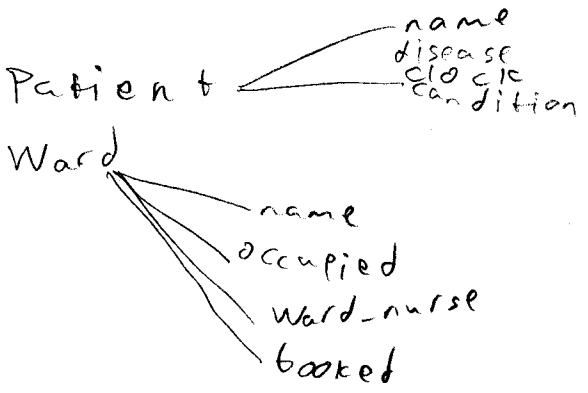
15 (c) An overall class diagram for your system, i.e., include all classes and their associated relationships for your system. Make it as inclusive as you can (don't just include the classes and relationships from part (b)). Note that subsystems are <u>not</u> required.

5 (d) Any inheritance diagrams with attributes and routines shown.

5 (e) The attributes and routines for the main classes that are not in the inheritance diagrams of part (d).

Total 150

THE END



bank

Actors
Nurses
Surgeons
Hospital Admin
Theatre Admin