UNIVERSITY OF EDINBURGH

FACULTY OF SCIENCE

COMPUTER SCIENCE 2Ah

Sat 6th June 1998

9.30 a.m. to 12 noon

Examiners

Colin Stirling (Chair)

Ursula Martin (External)

INSTRUCTIONS TO CANDIDATES

Section A questions are worth 10 marks each.

Section B questions are worth 30 marks each.

Answer all FOUR questions in Section A.

Answer any TWO of the three questions in Section B.

If you are in the third or later year of your degree course, please tick the box on the front cover of the script book.

Section A

1. a) The SML function `insertions` is declared as follows:

```
local
fun cons x xs = x :: xs
in
fun insertions x [] = [x]
|   insertions x (h :: t)  =
    (x :: h :: t) :: (map (cons h) (insertions x t)
end
```

For example, evaluating the expression `insertions 1 [3]` would return the result `[[1,3],[3,1]]`.

i. What value is returned when we evaluate the expression `insertions 1 [2,3]` ?

ii. What is the type of `insertions` ?

b) Define a function `perms: 'a list -> 'a list list` that takes a list, xs, of elements and returns all the permutations of xs.

As an example, evaluating `perms ["a", "b", "c"]` could return the list

```
[ ["a","b","c"],["a","c","b"],["b","a","c"],
 ["b","c","a"],["c","b","a"],["c","a","b"] ]
```

(the ordering of this list is not important).

In your answer, you may use the library functions `map` and `flatten : 'a list list -> 'a list`, and the function `insertions` declared in part *a*.

[10 marks]

2. a) What does it mean to say that $T(n) = O(f(n))$, that is that $T(n)$ is of order $f(n)$?

b) Prove that if $T(n) = 8n^2 - 10n + 5$ then $T(n) = O(n^2)$.

c) Write down a sorting algorithm in SML-speak that sorts a list of items and whose running time is $O(n\lg n)$.

[10 marks]

3. Consider the following SML datatype.

```
datatype 'a tree = Leaf of 'a | Node of 'a tree * 'a tree
```

   a) Define a function `reflect` of type `'a tree -> 'a tree` that maps any tree to its mirror image (i.e. a leaf reflects to itself, and an internal node is reflected by first reflecting both the left and right subtrees and then swapping them).

   b) Use the notation of predicate logic to express the principle of structural induction for the datatype `'a tree`.

   c) Prove that `t = reflect (reflect t)` for all trees `t`.

[10 marks]

4. The following signature describes a package of list operations.

```
signature LIST =
sig
  type 'a List
  exception Empty
  val Nil : 'a List
  val cons : 'a -> 'a List -> 'a List
  val hd : 'a List -> 'a
  val tl : 'a List -> 'a List
end
```

   a) Given the datatype declaration below,

   ```
   datatype 'a Llist = Emp | Cons of 'a * ('a Llist ref)
   ```

   implement a structure with the following header.

   ```
   structure Linked :> LIST
     where type 'a List = 'a Llist
   ```

   b) Briefly describe one possible application of the structure.

[10 marks]

Section B

5. This question concerns the optimisation of programs in a simple low-level programming language in which the example program below is written.

```
    x = 0
    i = 0
    k = n
L1: i = i + 1
    j = 0
L2: j = j + 1
    x = x + j
    h = i * i
    if j < h goto L2
    k = k - 1
    if k > 0 goto L1
```

   a) Describe in detail how an arbitrary program is divided into *basic blocks* and how its *flow graph* is constructed. [8 marks]

   b) Draw the flow graph obtained from the example program above. [4 marks]

   c) Define the notion of when one vertex in a flow graph *dominates* another. Assuming that you have available an algorithm checking whether any one given vertex dominates any other given vertex, describe how you can identify the *loops* in a flow graph. [4 marks]

   d) What is the domination relation between vertices in the flow graph given as your answer to part (b). Identify the loops in this flow graph. [4 marks]

   e) Describe two *global transformations* that can be applied to optimise programs, and discuss their benefits. [6 marks]

   f) Perform the optimisations described in part (e) to optimise the example program above. [4 marks]

6. Consider the following SML signatures:

```
signature WeightedGraphSig =
sig (* an integer-weighted, directed graph    *)
    eqtype Node
    val adj   : Node -> (Node * int) list
end

signature TableSig =
sig (* a train timetable                      *)
    eqtype Station
    type   Time
    eqtype Train
    val table : (Train* (Station*Time)list )list
    (* a list of trains, each with a list of  *)
    (* scheduled stops, station and time      *)
end

signature TimeSig =
sig
    eqtype Time (* represents times *)
    val -- : Time * Time -> int       (* infix *)
    (* time difference in minutes              *)
    val ++ : Time * int -> Time       (* infix *)
    (* (t ++ n) : time n minutes after time t  *)
end

signature SetSig =
sig
    type ''a Set
    val empty : ''a Set          (* empty set *)
    val member: ''a -> ''a Set -> bool
                           (* test membership *)
    val add   : ''a -> ''a Set -> ''a Set
                              (* add a member *)
end
```

Both parts of this question relate to these signatures.

a) Complete the following declaration of a functor that implements a search for a path, from a given node of a graph to a node with some property.

```
functor EXISTSPATH( structure G : WeightedGraphSig
                    structure S : SetSig) :
sig eqtype  Node

    path: Node -> (Node -> bool) -> bool
    (* searches for a path from a node n           *)
    (* to a node with some given property p        *)
    (* (path n p)  returns true if there is one    *)
end = struct
```

Write in your answer book the code that should appear here. [8 marks]

```
end
```

b) In this part, you may assume that an implementation of a functor SHORTESTPATH is provided, with the following specification:

```
functor SHORTESTPATH(structure G : WeightedGraphSig):
sig type  Node
    exception None

    path: Node -> (Node -> bool) -> Node list * int
    (* finds a shortest path from a node n         *)
    (* to a node with some given property p        *)
    (* (path n p)  returns the path and its length *)
    (* -- or raises exception None                 *)
end
```

It is proposed to use the functor SHORTESTPATH in an application that will search a train timetable to find a shortest itinerary to a destination station, leaving from a given station at or after some specified time. A functor ITINERARY is specified as follows.

```
functor ITINERARY( structure Table : TableSig
                   structure Time  : TimeSig
                   sharing type
                       Time.Time = Table.Time) :
sig type  Station
    type  Time
    type  Train
```

```
val itinerary :
    (Station * Time) * Station
    (* departure and destination *)
        -> (Train *              (* train       *)
            (Time * Station) *   (* departure   *)
            (Time * Station)     (* arrival     *)
           )list

end
```

You are asked to complete three steps that, taken together, implement the top-level functor `ITINERARY`.

Step 1. Connections may be made at any station provided the departing train is scheduled to leave at least five minutes later than the arriving train. Complete the following declaration of a functor that will represent the possible connections as edges of a graph.

```
functor GRAPHofTABLE(
        structure Table : TableSig
        structure Time  : TimeSig
        sharing type Time.Time = Table.Time
        ) : GraphSig =
struct
   type Node = Train option * Station * Time
   (* (None, st, tm) represents waiting at a *)
   (* given station and time - not on train  *)
   (* (Some tn, st, tm) boarding a train     *)

   (* any auxiliary declarations go here     *)
```

A. *Write in your answer book code that should appear here* [4 marks]

```
   fun adj (None, st, tm) =
   (* can board any train that calls at this *)
   (* station at some later time             *)
```

B. *Write in your answer book the code that should appear here* [4 marks]

```
   |  adj (Some tn, st, tm) =
   (* can be waiting at a station this train *)
```

```
      (* calls at, to board any train that      *)
      (* calls there at least 5 minutes later   *)
```

C. *Write in your answer book the code that should appear here* [4 marks]

```
end
```

This functor should construct a graph from a table in such a way that a shortest path through this graph will provide the information you require to construct an itinerary.

Step 2. Now construct an itinerary from a path through your graph. Complete the following declaration.

```
functor MKITINERARY(
        structure Table : TableSig
        structure Time  : TimeSig
        structure Graph : GraphSig
        where type Graph.Node
            = Train option*Station*Time
        sharing type Time.Time  = Table.Time
        ) :
sig type  Station
    type  Time
    type  Train
    type  Node

    val mkitinerary :
        Node list ->
          (Train *
           (Time*Station) * (Time*Station)) list
end
= struct
```

Write in your answer book the code that should appear here [4 marks]

```
end
```

Step 3. Write a declaration of the functor `ITINERARY` specified above, by calling the functors `SHORTESTPATH`, `GRAPHofTABLE` and `MKITINERARY`. [6 marks]

7. This question is about cryptosystems. Later sections can be answered independently of earlier sections.

   a) Explain the main features of a public-key cryptosystem. [4 marks]

   b) In the RSA cryptosystem each public key is a pair of long integers (e, n) and a secret key is a pair of integers (d, n).

      i. If the value represented by a block is B, what is the value of the encrypted block using the public key (e, n)?

      ii. If the value represented by an encrypted block is C, what is the value of the decrypted block using secret key (d, n)?

      iii. What is the relationship between e and d if decryption is the inverse of encryption?

      [6 marks]

   c) The procedure for choosing values d, e and n is as follows:

      1. Select two large random prime numbers p and q.
      2. Compute $n = pq$.
      3. Select a small odd integer e that is relatively prime to $(p - 1)(q - 1)$.
      4. Compute d such that $de \bmod ((p-1)(q-1)) = 1$ and $0 < d < (p - 1)(q - 1)$.

      Suppose $p = 11$ and $q = 7$. Calculate public and private keys from p and q using this procedure. [8 marks]

   d) Write down in an outline form a randomized algorithm for deciding whether a large integer is prime. [8 marks]

   e) What features does the security of the RSA cryptosystem depend on? [4 marks]