UNIVERSITY OF LONDON

IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 1996

BSc Honours Degree in Mathematics and Computer Science Part I
MSci Honours Degree in Mathematics and Computer Science Part I
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER MC1.7

TURING PROGRAMMING AND DATA STRUCTURES

Wednesday, May 8th 1996, 10.00 - 11.30

Answer THREE questions

For admin. only: paper contains
4 questions
4 pages (excluding cover page)

Section A *(Use a separate answer book for this Section)*

1a Write in Turing a function called `Hash` that takes string `s`, and a non-negative integer `n` as parameters and returns a number that is obtained by adding up the ordinal values of the characters in the string `s` modulo `n`. You may use the inbuilt function `length` in your function.

b i) Declare a data type `Student` suitable for holding a name, an email address, and a student number (non-negative integer).

ii) Declare a data structure `Class` suitable for holding `n Students`.

iii) Write a function `InitClass` that returns a `Class` where each entry has been set to

```
email = "xx"

name = "xx"

student number = 0
```

c Write a procedure `Include` that takes a `Student` and a `Class` and puts the `Student` into the `Class` at the position determined by the `Hash` function produced in part a of this question when applied to the `Student`'s name. If this space is already occupied then put the student in the next free space in `Class`. The pre-condition for `Include` is that there is at least one free place.

d Write a procedure `Email` that takes a `Student`'s name and a `Class` and prints on the screen the `Student`'s email address. If the `Student` is not in the `Class` the procedure should print a message to say this.

Make sure you include pre conditions in your code. They may be written in Turing or logic.

The four parts carry, respectively, 20%, 25%, 25%, and 30% of the marks.

2 A three-dimensional vector **v** has components v_x, v_y, and v_z .We can represent vectors in Turing programs using declarations such as

```
type Coord:enum(x,y,z)

type Vector : array Coord of real
```

Using these declarations, write functions corresponding to each of the following vector calculations.

a The *magnitude* of a vector **v** is $|\mathbf{v}| = \sqrt{(v_x^2 + v_y^2 + v_z^2)}$.

b The *dot product* of two vectors **u** and **v** is the real number

$$\mathbf{u} \bullet \mathbf{v} = u_x v_x + u_y v_y + u_z v_z .$$

c The *cross product* of two vectors **u** and **v** is a vector $\mathbf{w} = \mathbf{u} \times \mathbf{v}$ with components

$$w_x = u_y v_z - u_z v_y$$

$$w_y = u_z v_x - u_x v_z$$

$$w_z = u_x v_y - u_y v_x.$$

d The *triple product* of three vectors **u**, **v** and **w** is the real number $\mathbf{u} \bullet (\mathbf{v} \times \mathbf{w})$ in which $\bullet$ is the dot product and $\times$ is the cross product defined above.

e Declare a data structure suitable for holding `n Vectors`. Write a function that searches this data structure and returns the `Vector` with the smallest magnitude. If more than one `Vector` has this magnitude then return any of the `Vectors` with this magnitude.

The five parts carry equal marks.

Section B *(Use a separate answer book for this Section)*

3a Define a *binary search tree.* Draw an example.

List the *access procedures* for the abstract data type binary search tree.

Write *data declarations* and *code* for a dynamic implementation of the binary search tree in Turing.

b Write the following function in Turing:

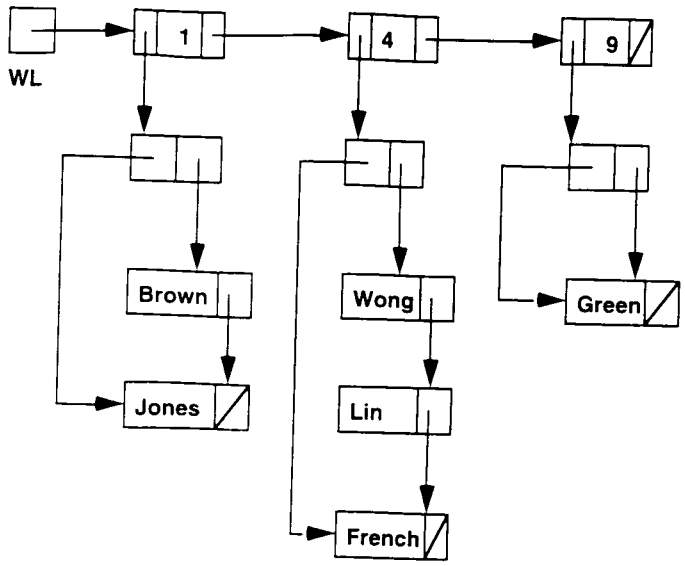
```
function TreetoList (T:Tree) : List
% pre :  takes a binary search tree with integer data items.
% past : returns a list containing the item values in the tree
%        in descending order.
```

You may assume the availability of the ADT List with access procedures Empty, IsEmpty, Head, Tail, Cons, and also Append.

The two parts carry, respectively, 80%, 20% of the marks.

4. This question is about an Abstract Data Type WaitingList.

The diagram is of a dynamic implementation of a hospital waiting list data structure in which patients are grouped according to a priority rating from 1 (highest) to 10. Patients with the same priority wait on a first come, first served basis. The next patient to be taken from the waiting list for treatment is the one of the highest priority rating who has been waiting the longest. The diagram shows the structure with patients Brown and Jones having priority 1; Wong, Lin and French with priority 4 and Green with priority 9.



a) Give type declarations in Turing for this dynamic implementation of the ADT WaitingList.

b) Write Turing code for the following access procedures:

function NextPatient (WL : WaitingList) : Name
% post : returns name of next patient to be treated

procedure Remove Patient (var : WL : WaitingList)
% post : removes from waiting list next patient to be treated.

c Suppose standard ADTs List and Queue were available

Describe how you could use these to construct the ADT WaitingList.

The three parts carry, respectively, 40%, 40%, 20% of the marks.

End of paper