UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2002

BEng Honours Degree in Computing Part I
MEng Honours Degrees in Computing 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 City and Guilds of London Institute

PAPER C120

DECLARATIVE PROGRAMMING

Wednesday 5 June 2002, 11:30
Duration: 90 minutes
(Reading time 5 minutes)

Answer TWO questions

Paper contains 2 questions
Calculators not required

Preliminaries

Enter `?- use_module(library(lists)).`
at the start of your Prolog script to gain access to the list-processing module.

A supporting file named tests.PL is made available to you.

2a Simple algebraic expressions can be represented by terms of three kinds:

atomic term	such as 2 or y
a(E1, E2)	representing (E1+E2) where E1 and E2 are expressions
m(E1, E2)	representing (E1*E2) where E1 and E2 are expressions

For example, m(a(x, m(2, y)), a(y, a(1, z))) represents (x+(2*y))*(y+(1+z)).
Here the atoms x, y and z represent algebraic variables.

Write a program for the relation subs(A1, A2, E, S) which holds when S is the result of replacing every occurrence, if any, in expression E of the atomic term A1 by the atomic term A2.

For example, if E = a(a(a(x, 1), 2), a(a(y, 4), m(x, 2))), A1=x, A2=5
then S = a(a(a(5, 1), 2), a(a(y, 4), m(5, 2)))

Use only the relations subs and the Prolog primitives atomic and \==.

Test your program by executing the queries

```
?- expr(1, E), subs(5, 6, E, S).
?- expr(2, E), subs(y, y, E, S).
?- expr(3, E), subs(z, x, E, S).
```

The expr calls will be evaluated automatically by the supporting file.

b Assuming expressions are of the same kinds as in part a, write a program for the relation deriv(E, F) which holds when F is an expression for the *first derivative* of expression E with respect to the algebraic variable x.
Atomic terms in E other than x are treated as constants when differentiated.

Your program needs two recursive clauses to construct the derivatives of a(E1, E2) and m(E1, E2), respectively, implementing the standard rules of calculus for differentiating a sum and a product.
Follow these by two base cases to compute the derivative of x and the derivative of any atomic term other than x, respectively.
Use only the relations deriv and the Prolog primitives atomic and \==.

Test your program by executing the queries

```
?- expr(1, E), deriv(E, F).
?- expr(2, E), deriv(E, F).
?- expr(3, E), deriv(E, F).
```

c Assume now that the representation is generalized so that terms of the form a(...) and m(...) can have 1 or more arguments instead of exactly two.

For example, a(3, x, m(2, y, a(4, x))) represents (3+x+(2*y*(4+x))).

Write a program for the relation allvars(E, Vs) which holds when Vs is a list of all the algebraic variables, if any, occurring in E.

```
For example,  if      E  = a(3, x, m(2, y, a(4, x)))
              then    Vs = [x, y, x]
```

The ordering in Vs does not matter, and duplicates need not be removed.

The program can be written using only the relations allvars and the Prolog primitives =.., findall, member, atom and number, but you may use other relations if you prefer.

Test your program by executing the queries

```
?- expr(1, E), allvars(E, Vs).
?- expr(3, E), allvars(E, Vs).
?- expr(4, E), allvars(E, Vs).
```

The three parts carry, respectively 25%, 30% and 45% of the marks.