SOLUTION NOTES

Software Engineering II 2003 Paper 2 Question 8 (LCP)

(a) This question covers Lecture 1 (Program Refinement).

```
program calendar
begin
get (month,year);
print_calendar(month,year)
end
```

```
procedure print_calendar(month,year)
begin
print_heading (month,year);
print_days (month,year)
end
```

```
procedure print_heading(month,year)
begin
print (month_name[month]); print(year); newline;
print " S  M Tu  W Th  F  S"; newline
end
```

Here we use 1 = Sunday, 2 = Monday, ..., 7 = Saturday. It is hard to break down this code sequence into smaller pieces because of the interactions between the first part (printing initial blanks) and the second part (printing the real days).

```
procedure print_days(month,year)
begin
weekday := weekday_of(1,month,year);
   (*weekday of the first of this month*)
print 3*(weekday-1) blank spaces;
for d := 1 to number_of_days(month,year) do
```

```
  begin
  if weekday = 7 then begin weekday :=1; newline end
  else weekday := weekday+1;
  print d (*in a field three spaces wide*)
  end
end
```

(b) This question covers Lecture 6 (Proving Programs Correct).

First problem:

The base case of the induction is when `xs=[]`. We have `nlength(app([],ys)) = nlength ys = 0 + nlength ys = nlength [] + nlength ys`.

The inductive case is when the list is `x::xs`. We have `nlength(app(x::xs,ys)) = nlength(x::app(xs,ys)) = nlength(app(xs,ys)) + 1 = nlength xs + nlength ys + 1 = nlength (x::xs) + nlength ys`.

Second problem:

The base case of the induction is when `xs=[]`. Then we have `nlength(nrev []) = nlength []` trivially.

The inductive step concerns a list of the form `x::xs`. We have `nlength(nrev (x::xs)) = nlength(app (nrev xs, [x])) = nlength(nrev xs) + nlength[x] = nlength(nrev xs) + 1 = nlength xs + 1 = nlength (x::xs)`.