

Typical Maple-based exam questions

- Computational Mathematics I:

2000 final exam paper, last question (LHS)

[9] *Vectors and matrices.*

Consider the recursive sequence of vectors in \mathcal{Q}^2 , defined by

$$u_1 = \left(1, -\frac{1}{2} \right); \quad u_{t+1} = u_t M + u_1, \quad 1 \leq t,$$

where

$$M = \begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}.$$

(a) (3 marks) Load the **linalg** package, and construct M and u_1 .

(b) (8 marks) Employ the **do**-structure to compute the (vector) sum of the first 100 elements of u_1, u_2, \dots .

- Solution (FJW)

```
> # with(linalg): # but actually completely unnecessary!
M := matrix([[2, 1], [1, 1]]); u1 := vector([1, -1/2]);

M :=  $\begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$ 
u1 :=  $\begin{bmatrix} 1 \\ -\frac{1}{2} \end{bmatrix}$ 

> u := u1: S := u1:
from 2 to 100 do
    u := evalm(u&*M + u1);
    S := evalm(S + u)
end do:
evalm(S);

 $\left[ \frac{627376215338105766356982006981782561278225}{2}, \right.$ 
 $\left. 193869912406111233457769413770852706167225 \right]$ 
```

Alternative Maple 6/7 solution (using rtables):

```
> M := << 2 | 1 >, < 1 | 1 >>; u1 := < 1 | -1/2 >;

M :=  $\begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix}$ 
u1 :=  $\begin{bmatrix} 1 \\ -\frac{1}{2} \end{bmatrix}$ 

> u := u1: S := u1:
from 2 to 100 do
    u := u.M + u1;
    S := S + u
```

```
end do:
```

```
S;
```

```
[ 627376215338105766356982006981782561278225,  
  2
```

```
193869912406111233457769413770852706167225 ]
```

- Computational Mathematics II:

2001 final exam paper, last question (FJW)

{Examiners: Unseen, although a power set procedure was covered in lectures.}

4. [20 marks] Write a procedure called **Subsets** that takes two arguments, a set S and a non-negative integer n , and returns the set of all subsets of S of size n . Note that a set can contain no subsets larger than itself and that every set contains the empty set as a trivial subset. These two conditions together provide a base case for the following recursive algorithm.

If S is non-empty then select any element s and let T denote S with the element s removed. Then the subsets of S of size n that do not contain the element s are given by $\text{Subsets}(T, n)$ and the subsets of S of size n that do contain the element s are given by inserting s into each subset returned by $\text{Subsets}(T, n - 1)$.

Here are some examples:

```
[ > Subsets({a,b,c}, 0);  
                                { { } }  
[ > Subsets({a,b,c}, 1);  
                                { {a}, {b}, {c} }  
[ > Subsets({a,b,c}, 2);  
                                { {a,b}, {b,c}, {a,c} }  
[ > Subsets({a,b,c}, 3);  
                                { {a,b,c} }  
[ > Subsets({a,b,c}, 4);  
                                { } ]
```

- Solution

```
[ > Subsets := proc(S::set, n::nonnegint)  
    # Return the set of subsets of S of size n  
    local s, T;  
    if n > nops(S) then return {} end if;  
    if n = 0 then return {{}} end if;  
    s := S[1];  
    T := S[2..-1]; # S minus {s}  
    Subsets(T, n) union  
        map(t->t union {s}, Subsets(T, n-1))  
end proc;
```