Curs pràctic de Maple

Pràctica 6

6 Llistes, conjunts, seqüències, taules i "matrius"

El programa Maple permet treballar amb *conjunts* de dades. Aquests *conjunts* es poden entrar de diverses maneres i podem canviar l'estructura segons ens interessi.

6.1 Llistes

Una llista és un conjunt ordenat d'expressions on es permeten repeticions. L'entrada al Maple es fa entre claudàtors (`[ ]`), separant els elements amb comes.
Exemple 6.1 Definim la variable a com la llista $[1, 2, 3, 2, 1, 2]$:

```
> a:=[1,2,3,2,1,2];
```

Els elements d'una llista es poden cridar un a un mitjançant la posició entre claudàtors. Si utilitzem nombres negatius, retorna l'element començant per la cua.
Exemple 6.2 Si volem el tercer element de la llista a és:

```
> a[3];
```

Mentre que si volem l'últim és:

```
> a[-1];
```

De la mateixa manera també podem extreure una subllista. Si volem una subllista amb els elements entre les posicions i i j, cal introduir l'argument `i..j`:

```
> a[2..4];
```

També podem construir una llista com la unió de vàries. Per a això necessitem abans "*treure*" els claudàtors i afegir-los després. La funció que ens permet treure els claudàtors és la funció `op( )`. Un cop hem tret els claudàtors podem concatenar dues llistes:

Exercici 6.1

Definiu la funció `concatenar`, depenent de dues variables, que retorni la unió de dues llistes donades.
Comproveu que la definició que heu fet és correcta fent la prova sobre les llistes $a = [1, 2, 3, 2, 1]$ i $b = [3, 2, 4]$. El resultat ha de ser la llista $c = [1, 2, 3, 2, 1, 3, 2, 4]$.

6.2 Conjunts

En un conjunt no hi ha repeticions i les dades no tenen un ordre prefixat (Maple pot variar l'ordre, depenent del context). L'entrada al Maple es fa entre claus (`{ }`), separant els elements per comes. La majoria d'opcions que hem vist per les llistes són també vàlides per als conjunts, tenint en compte les possibles diferències.
Exemple 6.1 Definiu a b el conjunt $\{1, 2, 5, 1, 3\}$:

```
> b:={1,2,5,1,3};
```

Observeu la sortida que us torna.

Els elements d'un conjunt també es poden cridar mitjançant els claudàtors, tenint en compte que, a priori, no coneixem l'ordre en que Maple guarda els elements.
Exemple 6.2 Per a cridar el tercer element del conjunt b fem:

```
> b[3];
```

Observeu que el tercer element no coincideix amb el que hem introduït quan hem definit b.

Exercici 6.2

Definiu a i b com els conjunts $\{x, y, z\}$ i $\{t, u\}$. Definiu $c = a \cup b$.
Definiu una funció que, a partir de dos conjunts, doni com a resultat la seva unió. Proveu-la sobre els conjunts $A = \{1, 4, 9, 16\}$ i $B = \{2, 4, 6, 8, 10\}$.

6.3 Seqüències

Una seqüència d'elements és una conjunt ordenat però que s'interpreta com n arguments (i no una sola llista), on n és el número d'elements de la seqüència. L'entrada al Maple és sense cap delimitador.
Exemple 6.1 Si volem definir com a la seqüència $1, 4, 9, 16$ posem

```
> a:=1,4,9,16;
```

Una altra manera d'aconseguir seqüències és mitjançant la comanda `seq` que vam introduir a la primera pràctica.
Exemple 6.2 La llista c amb els primers 10 quadrats enters.

```
> c:=seq(i^2,i=1..10);
```

Igual que en el cas de les llistes podem treure un element d'una seqüència, unir seqüències, ...

Exercici 6.3

Definiu la funció de dues variables $f(x, y) = x^2 - y^2$. Considereu $a = (5, 7)$. Com s'ha d'introduir a per aconseguir que $f(a)$ doni com a resultat el valor -24?

6.4 Taules (`table( )`)

Una taula és una llista d'objectes no necessàriament indexada per nombres enters consecutius. Per exemple:

```
> T:=table([a,b]);
```

Genera un objecte `T` de tipus taula amb valors per a `T[1]` i `T[2]`

```
> T[1];T[2];
```

Però també podem generar un objecte del mateix tipus on els índexs per als que tenim valors associats siguin indeterminades. Per exemple:

```
> S:= table();
> S[k]:= sin(k*Pi);
> op(S);
> S[1];
```

```
> S[k];
```

6.5 array's

Per a Maple els `array` són un tipus especial de `table` i són el tipus d'objecte bàsic en el que poden guardar dades de tipus matricial (és a dir que depenen d'un multiíndex).

```
> M:= array(1..2,1..2,[[a,b],[c,d]]);
> M[1,2];
> op(M);
> M;
> A:= array(1..3,1..2);
> op(A);
> A[1,1]:= 3:  A[1,2]:= 4:  A[2,1]:= 5:
> A[2,2]:= 2:  A[3,1]:= 1:  A[3,2]:= sin(Pi/7):
> op(A);
> B:= array(1..2,1..3,1..2);
> op(B);
```

també podem veure el contingut de `B` amb

```
> print(B);
```

6.6 Comptar elements de llistes i conjunts

La funció que permet comptar el nombre d'elements d'una llista o conjunt és la funció `nops( )`.

Exemple 6.1 Volem saber quants elements diferents té la llista $a = [3, 4, 2, 1, 3, 5, 3, 4, 3, 4, 5, 3, 2, 2, 4]$. La funció `nops( )` aplicada directament a la llista:

```
> a:=[3,4,2,1,3,5,3,4,3,4,5,3,2,2,4];
> nops(a);
```

retorna el nombre d'elements de la llista (amb repeticions incloses). Una manera de saber quants elements diferents té és passar primer la llista a conjunt b i comptar el nombre d'elements del conjunt:

```
> b:=convert(a,'set'); nops(b);
```

Exercici 6.4

Quants elements té la llista `a:=[[1,2,3],[x,y],{2, 3,4}, x, 1, 2]`?

6.7 Canvis de format entre llistes, seqüències i conjunts

Maple permet transformar una expressió que té guardada com una llista, seqüència o conjunt als altres formats.

A l'apartat de llistes ja hem vist com podem canviar una llista d'expressions en una seqüència, mitjançant la comanda `op( )`.

Exemple 6.1 Podem convertir la llista $a = [1, 2, 3, 2, 1]$ en una seqüència b fent

```
> a:=[1,2,3,2,1];
> b:=op(a);
```

Un cop tenim una seqüència podem transformar-la un altre cop en una llista afegint els delimitadors.

Exemple 6.2 Si volem recuperar la seqüència b en format llista podem afegir els claudàtors:

```
> [b];
```

Podem fer el mateix procediment amb els conjunts i les seqüències. Hem de tenir en compte que quan passem una seqüència o una llista a format conjunt perdem les repeticions i l'ordre que hi teníem.

Una altra manera de fer conversions entre llistes, seqüències i conjunts és amb la comanda `convert( )`. Aquesta comanda s'utilitza per a canvis en altres contexts dins de Maple (taules a matrius, ...).

Per a passar d'un format a l'altre tan sols cal conèixer la terminologia de Maple corresponent a cada un d'ells: `set` per a conjunts i `list` per a llistes.

Exemple 6.3 Considerem la llista formada pels elements $a = [3, 2, 1, 4, 3]$. Podem passar-la a format conjunt mitjançant:

```
> a:=[3,2,1,4,3];
> b:=convert(a,'set');
```

I podem tornar al format llista mitjançant:

```
> convert(b,'list');
```