Maple Lecture 21. Sequence, Set, and List

A set is a sequence enclosed by curly braces, a list is a sequence enclosed by square braces. There are two more fundamental differences between sets and lists. First, every element in a set occurs only once, while elements may repeat themselves in a list. Second, in a set Maple may arrange the order of the elements as it pleases, while the order of the elements in a list stays fixed. This lecture corresponds to [1, Sections 12.1-3].

21.1 Sequence

We have seen how to build sequences with higher order derivatives:

```
[> sqx24 := x$24;
[> whattype(sqx24);
```

To make a sequence of indexed elements we use the seq command:

```
[> indseq := seq(x[i],i=1..24);
```

Observe the difference with a sequence where the indices are concatenated to x:

```
[> cctseq := seq(x||i,i=1..24);
```

We know already how to get the number of elements in a sequence and how to select from a sequence:

```
[> indseq[13] , "is not the same as ", cctseq[13];
```

To get the number of elements in a sequence, we need to convert to a list:

```
[> nops([indseq]);
```

Suppose we wish to make the sequence a1,a2,..,b1,b2,..,c1,c2,..:

```
[> a||(1..5);
[> (a,b,c)||(1..5);
```

The last command gives an error message. We have to concatenate to the empty symbol “ (two left quotes):

```
[> ``||(a,b,c)||(1..5);
```

21.2 Set

A set in Maple corresponds to the mathematical notion of a set: there is no order and the set contains no duplicates.

```
[> S := {a,b,c};
[> whattype(S);
[> nops(S);
[> op(2,S);
[> S2 := {b,d,e};
[> S union S2; S intersect S2;
```

We can build sets of sets, for example the set of all subsets:

```
[> pS := combinat[powerset](S);
```

The number of elements in the set of all subsets is $2^{nops(S)}$. This cardinality (two to the power the number of elements in the set) helps understanding the naming of the command.

If we want repeated elements, we need to use a list:

```
[> L := [a, a, b];
[> pL := combinat[powerset](L);
```

Because of the repetition, we have fewer than 8 elements in pL.

Note that we can easily remove duplicates from a list by converting to a set and back :

```
[> LL := [op({op(L)})];
```

So the conversion from list to set and set to list went via a sequence (twice used op).

21.3 List

Remember the anonymous functions we have seen to execute on a list. Let us make a list of 20 random numbers between 1 and 6:

```
[> die := rand(1..6);
```

Here is the short way to build a list :

```
[> ranlist := [die()$20];
```

What went wrong? We must prevent premature evaluation:

```
[> randlist := ['die()'$20];
```

Here is an alternative way (in case you forgot about the right quotes) :

```
[> secondlist := []:
[> for i from 1 to 20 do
 >    secondlist := [op(secondlist),die()]:
 > end do:
[> secondlist;
```

It is interesting to replace the colons in the loop above by semicolons to see the evolution of the build up. Let us now make a list of random pairs:

```
[> zl := zip((x,y) -> [x,y],randlist,secondlist);
```

And here we show how to "flatten" the list :

```
[> map(op,zl);
```

21.4 Assignments

1. Generate the sequence of the first one hundred quotients of consecutive positive natural numbers, i.e.: $\frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \ldots$. Add up all the elements in the sequence.

2. Use the command **ithprime** to generate a sequence of the first 100 prime numbers. To visualize how this sequence grows, do the following:

   (a) Convert the sequence of the 100 first prime numbers to a list;

   (b) Generate a list of the first 100 natural numbers (starting at 1);

   (c) Combine the two lists into a list of points, like `[[1,2],[2,3],[3,5],[4,7],[5,11],[6,13],...]`, so the ith point in this list is $[i, \text{ithprime}(i)]$;

   (d) Calling the list of points `l`, type `plots[pointplot](l);` to visualize the sequence of primes.

3. Generate a sequence of cubes x^3, for x ranging over all natural numbers between 1 and 70. To determine which cubes are squares of natural numbers, take the square root of those cubes and select those who square root is a natural number.

4. Give the Maple commands

   (a) to make a list of 30 elements, the n-th one being $\cos(nt)$;

   (b) to compute numerical approximations (the default precision of 10 decimal places) for all 30 elements in the list, for $t = \frac{\pi}{13}$;

   (c) to combine two lists into a list of equations: the left hand side of the n-th entry is $\cos(n\frac{\pi}{13})$, and the right hand side is the numerical approximation for $\cos(n\frac{\pi}{13})$.

5. Consider `f := n -> trunc(evalf(Pi*10^n,n+3)) mod 10;`

   $f(n)$ returns the nth digit after the decimal point of π, i.e.: $f(0) = 3, f(1) = 1, f(2) = 4, \ldots$.

   (a) Use `f` to generate the list of the first 100 significant decimal places of π. Call this list `l`.

   (b) Given `l`, what is the Maple command to create an approximation for π with 100 decimal places? Compare your answer to `evalf(Pi,100)`.

6. Give the Maple commands to do the following tasks:

   (a) Generate two sequences, assign to `px` and `py`. The sequence `px` contains the x-coordinates of 8 points evenly spaced on the unit circle $x^2+y^2 = 1$, while the sequence `py` contains the corresponding y-coordinates. (*Hint:* use $(\cos(x))^2 + (\sin(x))^2 = 1$.)

   (b) Convert the two sequences in two lists, turning `px` into `lpx` and `py` into `lpy`. Combine the lists into one list of lists of the form $[[x_1, y_1], [x_2, y_2], \ldots, [x_8, y_8]]$. Call this lists of lists `pts`.

   (c) Use `pts` as input to create a list of floating-point approximations (default precision of 10 digits) for the coordinates.

7. Use the **seq** command in a nested way to generate the list of all triplets `[i,j,k]` of all possible combinations of `i`, `j`, `k`, ranging between 1 and 10. This is a list of 1000 lists of 3 elements.

   Select from this list all triplets whose sum equals 23. Compute the sum of all triplets in the selection to verify their sum is indeed 23. How many such triplets do you find?

8. A Farey sequence [2] is defined by

$$\mathcal{F}_n = \left\{ \frac{p}{q} \mid 1 \leq q \leq n, 0 \leq p \leq q \right\}, \quad n \geq 1.$$

   Define a Maple function `F` to compute $\mathcal{F}_n$, which takes as sole input argument n.
   Show the first five sets in this sequence.

9. Give all Maple commands (**NOT** the output) to do the following.

   (a) Make a list `L` of 100 random digits. A digit belongs to $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

   (b) Select from `L` all digits strictly larger than 5. Call this list `sL`.

   (c) Compute `n`, the number of elements in `sL`.

   (d) Use the digits in `sL` to make a natural number with `n` digits. Starting from the left (as usual), the ith digit in the number is the ith element of `sL`.

References

[1] A. Heck. *Introduction to Maple*. Springer-Verlag, third edition, 2003.

[2] F. Vivaldi. *Experimental Mathematics with Maple*. Chapman & Hall/CRC Mathematics, 2001.