

# Maple Crash Course



Ole Nielsen  
School of Mathematical Sciences  
Australian National University, Canberra



Ole.Nielsen@anu.edu.au

April 29, 2002

## Introduction

This text provides an informal introduction to selected parts of the programming language Maple for use in the ANU, SMS Summer school 2002. It should take about 10-15 minutes to complete most of it.

More comprehensive tutorials are available at

<http://www.mapleapps.com/categories/whatsnew/html/SCCCmapletutorial.html>

[http://saaz.lanl.gov/Maple/Maple\\_Home.html](http://saaz.lanl.gov/Maple/Maple_Home.html)

<http://www.indiana.edu/~statmath/math/maple/gettingstarted/index.html>

<http://www.ms.washington.edu/staff/paulm/mapletutorial/Maple.html>

## 1 A first glance

The first thing to try is to invoke the Maple interpreter by typing the command

maple

You should see the following message and the Maple prompt (>):

```
> maple
  | \ ^ / |      Maple 7 (DEC ALPHA UNIX)
._ | \ |      | / | _ . Copyright (c) 2001 by Waterloo Maple Inc.
 \  MAPLE  / All rights reserved. Maple is a registered trademark of
 < _____ > Waterloo Maple Inc.
   |
   Type ? for help.
>
```

Maple is now ready to take your order.

Like Python Maple can do calculations: End the command line with a semi-colon (;) to execute the line and display the result. Alternatively suppress the display of output by ending the line with a colon (:). For example

```
> a := 3+4;
a := 7
> a := 3+4:
```

both assigns the value 7 to a, but only the first variant displays the result on the screen.

We have already encountered some Maple code in the Python tutorial. Let us try it out bit by bit. To find out if a number is a prime number write

```
> isprime(16);
false
> isprime(17);
true
```

To create a list of numbers from 1 to 50, say, you can write

```
[seq(i,i=0..50)];
```

and to select those that are primes try

```
> primes := select(isprime, [seq(i,i=0..50)]);
primes := [2, 3, 5, 7, 11, 13, 17, 19, 23, ...]
```

The function `select` takes those elements from the list that are evaluated true by the given boolean function (here 'isprime'). Another Maple function `nextprime(n)` returns the smallest prime that is larger than n. For example:

```
> nextprime(13);
```

17

You can define your own functions in Maple using the symbol  $\rightarrow$  known as a functional operator. For example  $x \rightarrow x^2$  represents the function that squares its argument:

```
> f := x -> x^2;
```

```
> f(2);
```

$f := x \rightarrow x^2$

4

Let us now create a boolean function `is_primepair` of one argument  $p$  such that `is_primepair(p)` is true if both  $p$  and  $p + 2$  are primes.

```
> is_primepair := p -> nextprime(p)-p = 2;
```

To try it we can write

```
> is_primepair(3);
```

2 = 2

but Maple needs to know that we are evaluating it as a boolean function. Therefore we must write

```
> evalb(is_primepair(3));
```

true

```
> evalb(is_primepair(13));
```

false

We can now use this function to select those numbers from primes that are the first in a prime pair.

```
> pripai := select(is_primepair, primes):
```

```
> print(pripai):
```

[3, 5, 11, 17, 29, 41]

You will also need to concatenate lists in Maple, for example

```
> l := [2,3,5];
```

```
l := [2, 3, 5]
```

```
> m := [op(l), 6];
```

```
m := [2, 3, 5, 6]
```

```
> nops(m);
```

```
4
```

The function `op` 'unpacks the list' and the function `nops` counts the number of elements. To concatenate text with text or numbers you can use the function `cat`:

```
> cat("Formula ",1);
```

```
"Formula 1"
```

Finally, Maple can invoke Unix commands using `system`: For example

```
system("ls -la");
```

```
system("wc -w");
```

The first command lists your directory, the last waits for you to type some text followed by CTRL-d on a blank line.

