

BASIC COMMANDS

The MATHEMATICA constructs listed below are only a small subset of the language. But if you acquire a working knowledge of what they can do for you, they will be sufficient for all your needs in this course. This list should be used only as a guide. Look up any item in this list in a MATHEMATICA manual and using the information you get from the manual try testing a program that makes use of this particular item. In an attached handout we give sample programs that use some of these items. It is important to keep in mind that all MATHEMATICA reserved words start with a capital letter even when you put them together to make a composite MATHEMATICA command.

1. Working with "lists"

The command:

$$\mathbf{A} = \{\{1, 2\}, \{4\}, \{3\}, \{1, 3, 2\}\} \quad (1)$$

stores the given sequence under the name of **A**. The "=" sign is the "storing" command here. To get the first component of **A** you just type

$$\mathbf{A}[[1]] \quad (2)$$

the computer will reply {1, 2}. Similarly, typing $\mathbf{A}[[4]]$ yields {1, 3, 2}.

The command:

$$\mathbf{B} = \text{Table}[i^2, \{i, 1, 4\}] \quad (3)$$

stores in **B** the sequence {1, 4, 9, 16}.

What happens when you type the following command?

$$\mathbf{B} = \text{Table}[i + j, \{i, 1, 3\}, \{j, 1, 5\}] \quad (4)$$

If you haven't already guessed, try it on the computer.

Familiarize yourself with

Table , **Range** , **Length**
Take , **Drop**
Position , **Count** , **MemberQ** , **FreeQ**
Insert , **Delete** , **Select**
Join , **Union** , **Complement**
Sort , **Reverse** , **Rotate**
Apply , **Map**

The last two are somewhat tricky and we shall only use them in some very special situations. Example

$$\text{Apply}[\text{Plus}, \mathbf{A}]$$

gives the sum of the elements of **A** and

$$\text{Map}[\text{Cos}, \mathbf{R}]$$

which takes the cosine of every element of **R**. Of course **R** better be a sequence of real numbers or else you will get some complaints.

2. Constructing a MATHEMATICA Function

The structure of a MATHEMATICA function is

$$\mathbf{BLAHBLAH}[x_-, y_-, \dots] := (\mathit{CO}_1; \mathit{CO}_2; \dots; \mathbf{Return}[\mathbf{GOO}])$$

where **BLAHBLAH** is the function name, x, y, \dots are the input variables. $\mathit{CO}_1; \mathit{CO}_2$ is a sequence of composite commands of your manufacture. Hopefully, during the execution of this sequence of commands, you have an assignment which stores something in **GOO**. The final command **Return[GOO]** does exactly what it says, it spits out the value assigned to **GOO** in the middle of any expression which contains an invocation of **BLAHBLAH** precisely at the very location where this invocation takes place.

To be able to put together a sufficiently wide variety of functions, the basic commands that you must familiarize yourself with are as follows:

Block

Sum , **Product**

Do , **If** , **While** , **For** , **Return**

Which

The latter is an important command that allows to take alternate courses of action according to the results of a sequence of tests. More precisely, this command is called in the form

$$\mathbf{Which}[\mathbf{Test}_1, \mathit{CO}_1, \mathbf{Test}_2, \mathit{CO}_2, \dots, \mathbf{Test}_i, \mathit{CO}_i, \dots]$$

Then CO_i is the evaluated command if **Test_j** yields **True** and **Test_j** yields **False** for $j = 1, 2, \dots, i - 1$.

It might be good at this point to give an example. Say we want to write a MATHEMATICA function which gives you the positions of the elements of a sequence that are greater than zero. This function, which we shall name **findpos** (as an abbreviation of *find position*) should be such that if we type

$$\mathbf{seq} = \{2, 4, 0, 3, -1, 0, -2, 4, 2\} \tag{5}$$

then typing

findpos[seq]

should return

$$\{1, 2, 4, 8, 9\}$$

We are going to write such a function in a manner that also illustrates uses of the commands

Do, **If**, **Length**, **Return**.

The simplest way to select the positions of the positive components of **seq** is to look at each of them in succession and store their locations in a progressively growing output sequence which we shall call “**out**”. We can do this by a **Do** command with loop variable **i** which takes the values **1, 2, ..., Length[seq]**. In MATHEMATICA the i^{th} component of the sequence **seq** is addressed to as **seq[[i]]**. So the string **seq[[i]] > 0**

returns **True** or **False** according as the i^{th} entry in **seq** is positive or not. The **If** command in MATHEMATICA has the structure

If[**boo** , **CO1** , **CO2**] .

It executes the command **CO1** if the boolean expression **boo** evaluates to **True** and executes the command **CO2** if **boo** evaluates to **False**. Following these ideas we are led to the following MATHEMATICA function:

```
findpos[seq_] := ( out = {};
                  Do [
                      If [ seq[[i]] > 0 , out = Join [out, {i}] ] ,
                      {i, 1, Length [seq]}
                    ];
                  Return [out]
                )
```

Some additional useful commands which you may need are

Simplify , **Expand** , **MatrixForm**
Binomial , **Permutations**
Floor , **Ceiling** , **Min**
Get , **Save** , **Put** , **PutAppend**

If you need to declare some variables to be local to the given procedure you may do this by starting your procedure with the “**Block**” command. The structure of such a procedure is as follows

```
BLAHBLAH[x_, y_, ...] :=Block[{a, b, c, ...},
                              (COI1;
                               COI2;
                               COI3;
                               ...
                               Return[GOO])
                              ]
```

Where **a, b, c** ... are the local variables.