**Chapter 6 Short Summary**

\* *Boolean constants*

Maple uses an extension of Boolean logic that includes three Boolean constants, *True*, *False*, and *Fail*. (For most purposes you can ignore fail).

\* *Relational operators*

-> Most relational operator symbols found in palettes are inert. Some of these have defined meanings such as , and can be used in a logical context. Out of these, onlyis active and evaluates to either true or false if possible.

Note: is always simplified torespectively. e.g.

**> **



\* *lhs()* and *rhs()*

*lhs()* and *rhs()* return respectively the left-hand side and the right-hand side of any defined symbolic relation or range.

e.g.

**> **



e.g.  = , since is simplified to .

\* *evalb()* and *is()*

*evalb()* performs simple Boolean evaluation. *is()* performs more sophisticated Boolean evaluation.

(In simple cases *is()* and *evalb()* produce the same result.)

e.g.

**> **



**> **



Remark: *is()* is more sophisticated thus is less reliable and still has a few bugs. Generally,  will try harder to return a logical value whereas *evalb()* will just return its (simplified) argument if it cannot evaluate it directly, e.g.

**> **



**> **



Remark: The Boolean evaluator *evalb()* can only compare explicit numbers. (You can use floating point approximation for this). E.g.

**> **



**> **

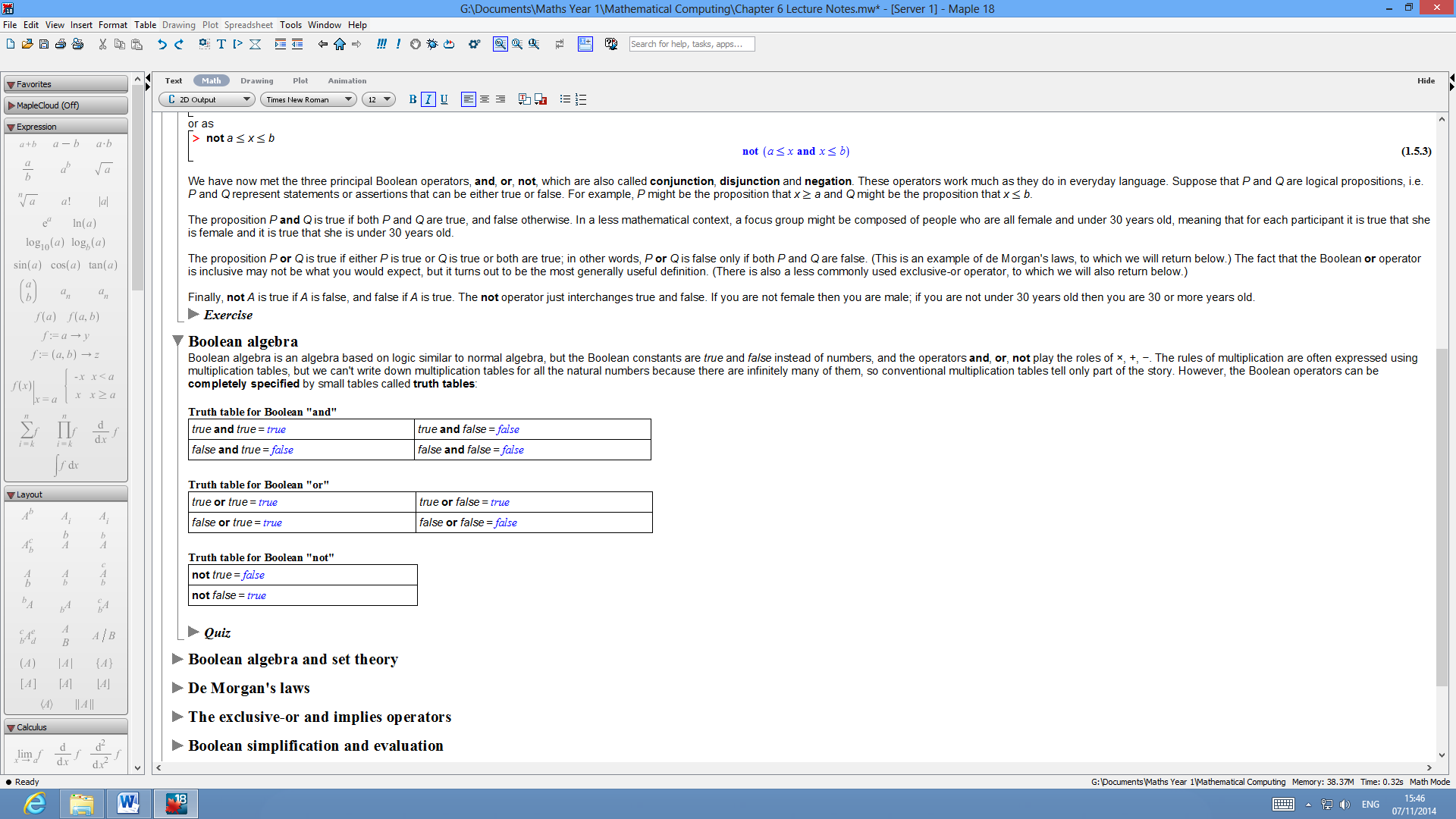


\* Boolean *Operators: and, or, not, exclusive-or, implies.*

-> *and*, *or*, *not*

(i.e. *conjunction*, *disjunction* and *negation*).

Truth tables:



Symbols: ****, ****, ¬, are found in the Common Symbols palette.

**> **

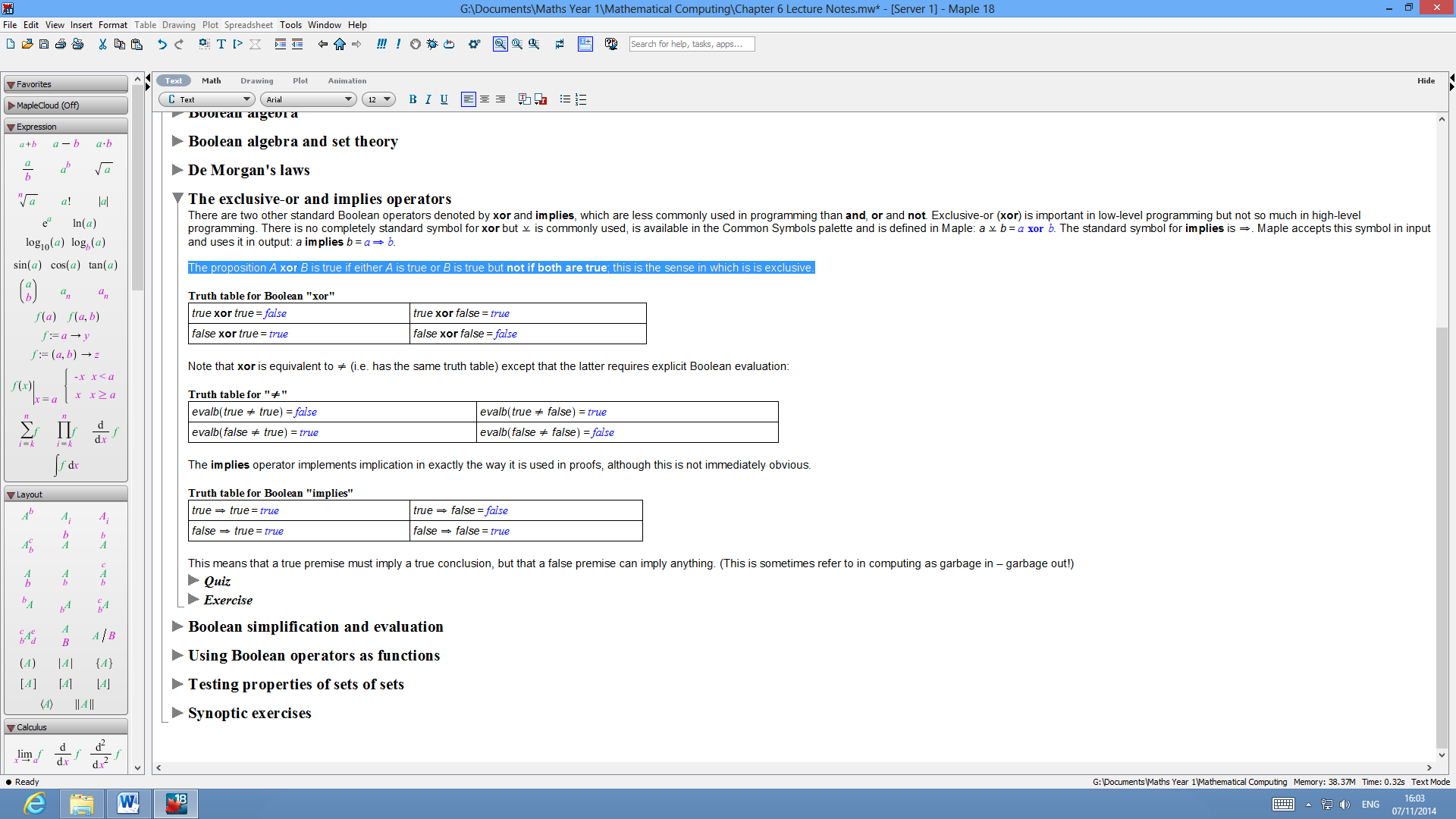


-> *Exclusive-or*

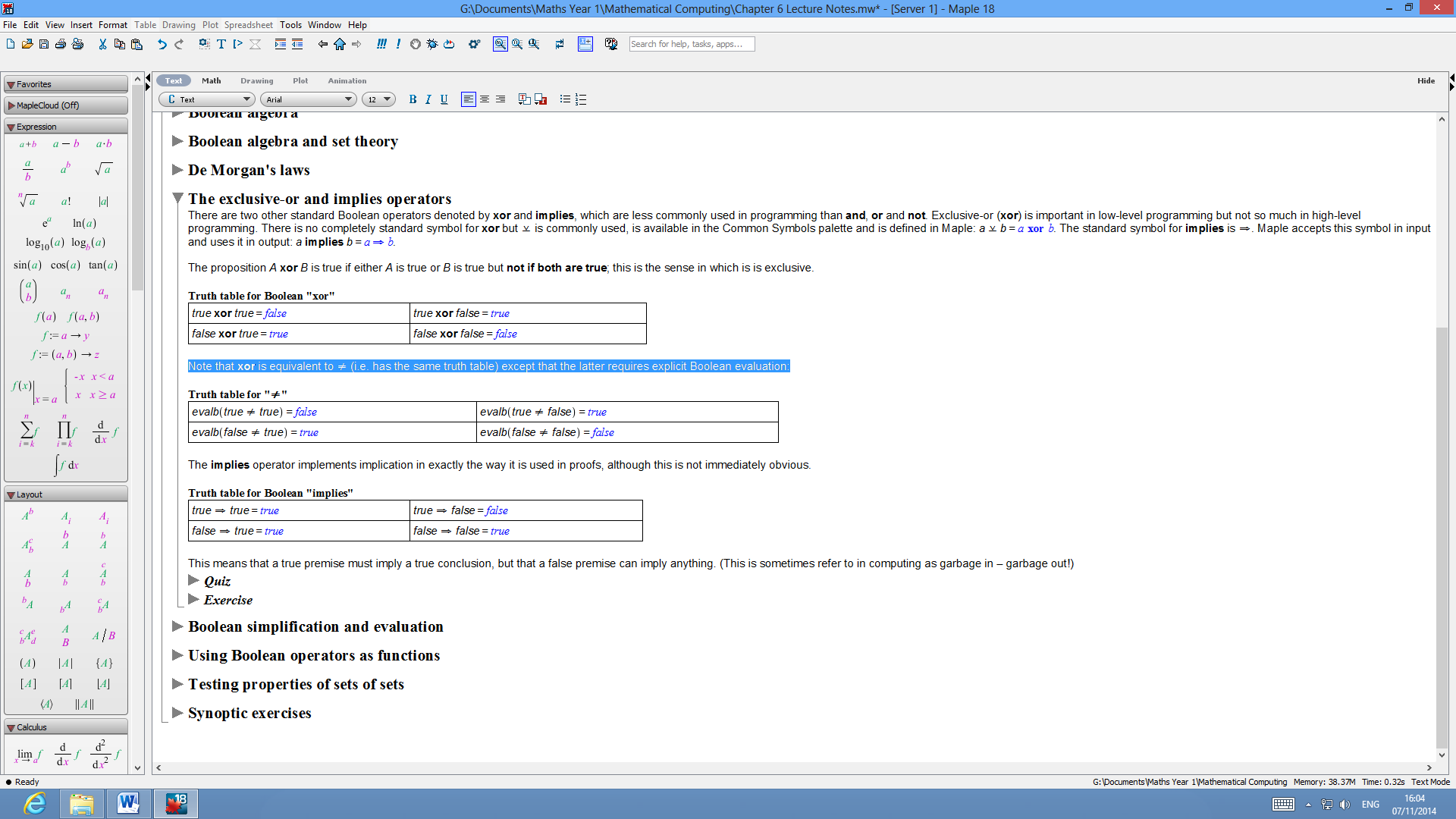
In Maple: **xor**, or ⊻ (common symbols palette)

i.e.  = 

The proposition  is true if either  is true or  is true but not if both are true. (This is the sense in which it is exclusive.)



(Remark:  is equivalent toexcept that the latter requires explicit Boolean evaluation):

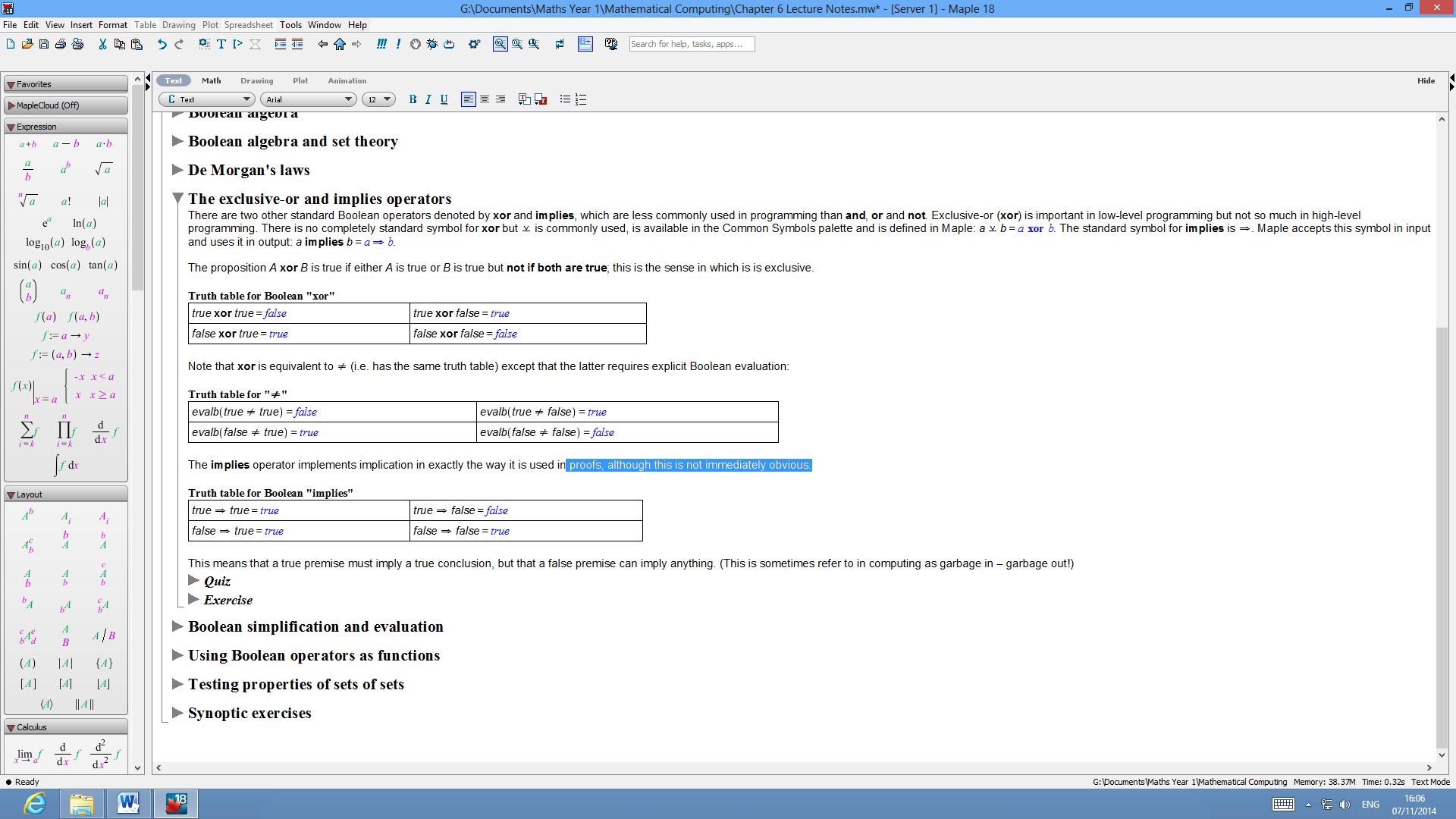


-> *Implies*

In Maple: **implies**, or

i.e.  = 

Truth Table:



(\* Boolean *simplification*)

e.g.

**> **



**> **



**> **



**> **



**> **



**> **



(\* Boolean *evaluation*)

In Maple,  implement ‘short-circuit’ or McCarthy evaluation rules. This means the left operand is evaluated first. If this determines the value of the expression then the right operand is not evaluated at all. Hence it matters which way round the operands are written.

(\* Using Boolean *operators as functions*)

**and**, **or**,can also be used as functions with zero or more arguments, e.g.

**> **



**> **



**> **



**> **



e.g. A test to see whether all elements of a set S are positive:

**> **



**> **



\* *Testing if a set of sets* S *is pairwise disjoint.*

To check if the elements of  are *pairwise disjoint* (i.e.  for every  such that ), use:

**> **

(\* Aside: Link between*sets and logic*)

-> For sets *A*, *B*, we have

,

,

.

-> *De Morgan’s Laws*

,

.

And analogously, if  denotes the complement of set , then

,

.