Computer Science 1FC3

Lab 1 – Logic and the Maple Environment

Author – Paul Vrbik – vrbikp@mcmaster.ca

The purpose of this lab is to implement the boolean functions in maple in order to evaluate the equivalence of some simple propositional statements.

MAPLE

Maple is a computer program that was developed to solve mathematical problems. In some sense it is like a glorified calculator where you can ask for more complicated things like the integral of a function or the result of a matrix multiplication. Today we will only concern ourselves with a very small aspect of Maple called the *boolean operations*, these are calculations which will always result in the answer `TRUE` or `FALSE`, referred to in your textbook as *propositional statements*.

Upon opening Maple you are presented with this: `[>_` , which is what we refer to as the *command line*. This is where we will type instructions to maple to receive feedback. All instructions must be followed by a semi-colon "`;`" and are executed by hitting enter. For example, to assign the values of `TRUE` and `FALSE` respectively to the names `P` and `Q` we do:

```
[>P:=true;
                              P:=true
[>Q:=false;
                              Q:=false
```

We can now refer to the values of `P` and `Q` for the entire Maple *worksheet*, which is what we call a collection of Maple *commands*. More interestingly we can carry out some calculations:

```
]>2+2;
                              4
]>true or true;
                              true
]>true and true;
                              true
]>not true;
                              false
]>P or Q;
                              true
```

FUNCTIONS

A function, in mathematics, maps values to values. For instance the function $f(x, y) = 2x + 3y$ maps the tuple $(4,5)$ to 23 which we denote as $(4, 5) \to 23$. To define `f` (and some other functions) in maple we do the following.

```
]>f:=(x,y)->2*x+3*y;
                              f(x,y):=2x+3y

]>f(4,5);
                                 23
]>Prop:=(a,b)->not (a and b);
                         Prop(a,b)→ not(a and b)
]>Prop(true,true);
                                false
```

We will be dealing with functions quite extensively in this course so it is really important to get comfortable with these types of definitions.

LOGIC

As promised we are now going to use maple to define `AND`, `OR`, `IMPLIES`, and `BI_IMPLIES` as defined in your textbook by conjunction, disjunction, implication and biconditional, respectively. Furthermore, we will do this by only using `and` and `not` which is possible because every logical statement may be converted into an equivalent statement using only these operators.

Verify that these function definitions are correct by using maple to fill in the blanks.

AND

```
]>AND:=(a,b)->a and b;
```

$\wedge$				AND		
A	*B*	$A \wedge B$		*A*	*B*	`AND(a,b)`
T	T	T		T	T	
T	F	F		T	F	
F	T	F		F	T	
F	F	F		F	F	

OR

```
]>OR:=(a,b)->not(AND(not a, not b));
```

$\vee$				OR		
A	*B*	$A \vee B$		*A*	*B*	`OR(a,b)`
T	T	T		T	T	
T	F	T		T	F	
F	T	T		F	T	
F	F	F		F	F	

IMPLIES

```
]>IMPLIES:=(a,b)->OR(not a, b);
```

$\rightarrow$				**IMPLIES**		
A	*B*	*A* $\rightarrow$ *B*		*A*	*B*	IMPLIES(a,b)
T	T	T		T	T	
T	F	F		T	F	
F	T	T		F	T	
F	F	T		F	F	

BI_IMPLIES

```
]>BI_IMPLIES:=(a,b)->AND(IMPLIES(a,b),IMPLIES(b,a));
```

$\leftrightarrow$				**BI_IMPLIES**		
A	*B*	*A* $\leftrightarrow$ *B*		*A*	*B*	BI_IMPLIES(a,b)
T	T	T		T	T	
T	F	F		T	F	
F	T	F		F	T	
F	F	T		F	F	

LOGICAL EQUIVALENCE

To verify that two propositions, P and Q, are *logically equivalent* we usually check that the corresponding truth tables are exactly similar. However, in maple, it would be more convenient to check that $P \leftrightarrow Q$ is a *tautology*, that is, this new statement is always true. Consider the first De Morgan's law $\neg(a \wedge b) \equiv \neg a \vee \neg b$ we will show that $\neg(a \wedge b) \leftrightarrow \neg a \vee \neg b$ is a tautology.

```
]>Check:=(a,b)->BI_IMPLIES(not(AND(a,b)),not(OR(not a, not b)));
]>Check(true,true);
                              true
]>Check(true,false);
                              true
]>Check(false,true);
                              true
]>Check(false,false);
                              true
```

PROBLEMS

1) Rewrite the last four functions using the notation given in the textbook, i.e. $AND(a, b) \equiv a \wedge b$.

```
OR(a,b)

IMPLIES(a,b)

BI_IMPLIES(a,b)
```

2) Verify, using maple, that the first Associative law is sound by showing $(p \vee q) \vee r \leftrightarrow p \vee (q \vee r)$ is a tautology.

3) Define a new logical operator # by $A \# B \equiv \neg(A \wedge B)$ what is `A#A` logically equivalent too? What is `(A#B)#(A#B)` logically equivalent too? Use Maple to explore the possibilities (HARD).