Computer Science 1FC3

Lab 2 – Predicates, Quantifiers and Proofs

Author: Dai Tri Man Le – ledt@mcmaster.ca

Introduction: *This lab will help you to understand logical quantifiers and some methods of proofs. Through examples using Maple, we will demonstrate how quantifiers over finite Universes work. We also show some examples where we use Maple to assist our proofs.*

Predicate and Quantifiers

Predicate

Question: Considering the ***universe*** of Integer numbers, is $x+1 = 0$ ***proposition***? Why or why not?
Answer: It is NOT a proposition, since we need to know the value of x in order to know whether the previous sentence is true or false. Such statement that "can be true of something" is called a ***predicate***. Thus we can write *P(x): x+1 = 0* to indicate that *P(x)* is the sentence "*x+1 = 0.*".

For a specific value of x, P(x) becomes a statement that is either true or false. For example, P(–1) is true and P(100) is false.

Formally, **a predicate is a *function* which maps each input to an output of type Boolean (TRUE or FALSE).**
(Predicate is also called "***propositional function***" in our textbook.)

Example 1

We can rewrite the following predicate "x is less than 10 and greater or equal to 5" in the logical symbolism as

$$\mathbf{P(x): x < 10 \wedge x > 5}$$

And then we translate it directly into Maple codes as a form of function definition:

```
> P:= x -> x < 10 and x>5;
```

$$P := x \rightarrow x < 10 \text{ and } 5 < x \qquad (1.1.1.1)$$

```
> P(2); P(11); P(7); P(5);
```

false

```
false
true
false                                    (1.1.1.2)
```

Example 2

Just like a normal function, it can take input as a tuple of objects. For simplicity, we will give a example of a predicate takes input as two Boolean variables "x or y but *not both*" (exclusive or).

We can rewrite the given predicate in the logical symbolism as

$$\mathbf{Q(x,y): (x \land \neg y) \lor (\neg\, x \land y)}$$

The correspondent Maple codes for Q(x,y) are given as follows. Also, type these commands to Maple and explain the results.

```
> Q:= (x,y) ->  (x and not y) or (not x and y);
> Q(true,true);Q(true,false);Q(false,true);Q(false,false);
```

Exercise 1

Can you do the same for the following predicate, which again takes input as two Boolean values : "if x or y but *not both, then x or y" using* ***and, or*** *and* ***not****?*

Write the corresponding Maple codes and run some test cases.

```
> #R:=
>
```

Exercise 2

Can you do the same for the following predicate, which again takes input as two natural numbers : "x and y are both non-zero" *?*

Write the corresponding Maple codes and run some test cases.

```
> #S:=
>
```

Quantifiers (∀ and ∃)

One way to create a proposition from a predicate is to use ***quantifiers*** with respect to some ***universe of discourse (universe)***.
(Question: Can you think of any other ways?)

We have two quantifiers:
Two quantifiers we will deal very often with in the scope of this course is the ***universal quantifier*** ∀ and ***existential quantifier*** ∃.

Example 3
Examples1 and Exercise 2 (previous section), we can extend the predicates P (x) and Q(x,y) to propositions as following:

Suppose the universe consists of all ***natural numbers***.

Logical formula	English Sentence
$\forall x\ (x < 10 \land x > 5)$	Every natural number is less than 10 and greater or equal to 5.
$\exists x\ (x < 10 \land x > 5)$	Some natural number is less than 10 and greater or equal to 5.

For the proposition *∀x (x < 10 and x > 5)*, we have

```
> P(0);
                              false                              (1.2.1.1)
```

The case x=0 which make P(x) false here is sufficient to prove that that $\forall x\ (x < 10 \land x > 5)$ is false. Hence, it is called a "***counter example***". Also this process of proving a proposition false is called "***disproving***".

Notes: *Very often in the future, when you will be asked to decide if a proposition with quantifiers is true of false, (this is usually the case when you have to deal with some "**open problem**" in Computer Science or Maths!) it is a very good practice to disprove the proposition first before you decide to prove it!*

For the proposition *$\exists x\ (x < 10 \land x > 5)$*, we have

```
> P(6);
                              true                               (1.2.1.2)
```

Question: can we conclude anything from the case x=6 here? Why?

Notes: *These examples seem a little "silly", since we can easily calculate the* *value of P(x) by ... hand! However, if P(x) is some complicated predicate, then Maple can help you speed the evaluation process a lot. More techniques will be discussed later in this tutorial (and in the rest of this course!).*

Exercise 3

Given the predicate *T(x,y): (xy>0)* $\rightarrow$ *(x>0)* in the universe of ***integers***, rewriting the following English sentences in the logical symbolism.

Logical formula	English Sentence
	i. For any x and any y, if the product of x and y is strictly greater than 0 then x is strictly greater than 0
	ii. There is an y such that for all x, the product of x and y is strictly greater than 0 implies x is strictly greater than 0

Is the sentence (i) true or false? Why?

```
>
>
>
```

Is the sentence (ii) true or false? Why?

```
>
>
>
```

Quantifiers over a finite Universe (Advanced)

As you can see the previous sections, to create a proposition from a predicate using quantifiers we need a ***universe of discourse (universe)***. When the universe is "small" and "simple" enough (you can count and list them on .... paper!), it is very easy to use a computer program to automate the process of proving or disproving using counter examples. Hence,
Maple is an extremely powerful tool to do this as you can see next.

Example 4

We can create a univese with Maple, by put the elements in the universe into a list! Suppose that your universe includes all integers from 1 to 10. We can create the Universe with elements from 1 to 10 as following:

```
> U1 := [1,2,3,4,5,6,7,8,9,10];
```

$$U1 := [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] \qquad (1.3.1.1)$$

```
> U2 := [seq( i , i=1..10)];
```

$$U2 := [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] \qquad (1.3.1.2)$$

```
> evalb(U1 = U2);
```

$$true \qquad (1.3.1.3)$$

Notice that U1 and U2 are actually the same, but which way you prefer?

Example 5

Suppose you have the Predicate Y(x): $x < 11 \wedge x > 0$, and you want to check the following propositions $\forall x\, (x < 11 \wedge x > 0)$ on the universe of all natural numbers . You can use the one of 3 following methods:

```
> Y:= x -> x<11 and x>0;
```

$$Y := x \rightarrow x < 11 \textbf{ and } 0 < x \qquad (1.3.2.1)$$

Method 1:

```
> [Y(1),Y(2),Y(3),Y(4),Y(5),Y(6),Y(7),Y(8),Y(9),Y(10)];
```

$$[true, true, true, true, true, true, true, true, true, true] \qquad (1.3.2.2)$$

Method 2:

```
> map(Y,U1);
```

$$[true, true, true, true, true, true, true, true, true, true] \qquad (1.3.2.3)$$

Method 3:

```
> [seq( Y(i) , i=1..10)];
```

$$[true, true, true, true, true, true, true, true, true, true] \qquad (1.3.2.4)$$

Which method do you prefer? Do you like Method 1?

Exercise 4

Can you use the same method as in Example 4 to prove the proposition *∀x $(x+x^2+x^3<156)$* on the universe all intergers from 1 to 5?

```
>
>
>
>
>
```

Note: If you do not fully understand Example 4, it will be ok, since we will deal these Maple coding styles very often in this course!

Negation of quantifiers

Negation of propositions with quantifiers can be derived using the following two rules:

Proposition	Negation
¬∀x P(x)	∃x ¬P(x)
¬∃x P(x)	∀x ¬P(x)

Example 6

We can get the negation of propositions in Example 3

Proposition	Negation
∀x (x < 10 ∧ x > 5)	¬∃x (x < 10 ∧ x > 5) ∃x ¬(x < 10 ∧ x > 5) ∃x (¬(x < 10) ∨ ¬(x > 5)) ∃x ((x >= 10) ∨ (x <= 5))
∃x (x < 10 ∧ x > 5)	¬∀x (x < 10 ∧ x > 5) ∀x ¬ (x < 10 ∧ x > 5) ∀x (¬(x < 10) ∨ ¬(x > 5)) ∀x ((x >= 10) ∨ (x <= 5))

Notes: As you can see you can get many equivalent negation forms of the same formula, and the question is "which one should we choose?". The answer is it all depends on the purpose of getting negation. Negation of a logical formular might *be very helpful in a lot of **proofs**, and depends on what you want to prove, you might decide which form suits the situation the most!*

Exercise 5

Get the negation of the following proposition:

Proposition	Negation
$\forall x\ \exists y\ ((xy>0) \rightarrow (x>0))$	
$\exists x \exists y \forall z((xy>0) \wedge (z<0))$	