Truth Tables using SML

This SML program is an exploration of techniques to print out truth tables of propositional expressions. For example, this table:

p	q	p or q
true	true	true
true	false	true
false	true	true
false	false	false

(The simplistic formatting is intentional anticipating limited support in SML for such things!)

To start, we will only use 2 propositional variables. The inputs to the table must be all the ways to combine values of p with values of q. That is a cartesian product.

SML

```
fun pair_with_first (x,[]) = []
  | pair_with_first (x,y::ys) = (x,y) ::
                                 pair_with_first (x,ys) ;
fun cartesian ([],ys) = []
  | cartesian (x::xs,ys) =
        pair_with_first (x,ys) @ cartesian (xs,ys) ;
```

That produces this list of inputs to the truth table.

SML

```
val ps = [true,false] ;
val qs = [true,false] ;
cartesian (ps,qs) ;
```

```
> val it = [(true, true), (true, false), (false, true), (false, false)]:
   (bool * bool) list
```

Now we need a way to format the columns. And a way to print out a list of strings. This will make the width of each column 8 characters, left justifed by padding on the right.

SML

```
fun fmt s = TextIO.print (StringCvt.padRight #" " 8 s) ;
fun prt (lst : string list) = List.app fmt lst ;
```

The list of boolean values are in tuples, so process that list, applying the propositional function to the 3rd column.

SML

```
fun bools_to_list_of_strings f ((x:bool,y:bool)::xs) =
    "\n" :: (Bool.toString x) :: (Bool.toString y) ::
    (Bool.toString (f (x,y))) :: "\n" ::
    (bools_to_list_of_strings f xs)
  | bools_to_list_of_strings f _ = [] ;
fun pf (p,q) = p orelse q ;
val tt = bools_to_list_of_strings pf (cartesian (ps,qs)) ;
```

With all the strings prepared, we can finally print out the table.

SML

```
val sep = ["------","------","------","\n"] ;
val hdr = ["p","q","p or q","\n"] ;
prt (sep @ hdr @ sep @ tt @ ["\n"]) ;
```

```
p       q       p or q
-----   -----   -----

true    true    true

true    false   true

false   true    true

false   false   false
```

And test it by trying a different propositional function. For example:

p	q	p and q
true	true	true
true	false	false
false	true	false
false	false	false

SML

```
fun pf (p,q) = p andalso q ;
val tt = bools_to_list_of_strings pf (cartesian (ps,qs)) ;
val hdr = ["p","q","p and q","\n"] ;
prt (sep @ hdr @ sep @ tt @ ["\n"]) ;
```

```
p        q        p and q
-----    -----    -----

true     true     true

true     false    false

false    true     false

false    false    false
```