Programming Language Concepts, CS2104 (10th September 2007) Tutorial 3.

Exercise 1. (Filter) Using the Filter higher-order function, write an Oz function which selects from a list of integers those which are positive even. For example, if the input is `[1 ~2 8 3 4]`, the output should be `[8 4]`.

Exercise 2. (Computing Maximum with Fold) Compute the maximum element from a list of numbers by folding. What is the initial value to choose for passing to `FoldL` or `FoldR` (remember: there is no smallest integer as integer precision is unlimited)? Which version of folding are you using (`FoldL` or `FoldR`)? Why?

Exercise 3. (Mapping Tuples) Develop a function `{MapTuple T F}` that returns a tuple that has the same width and label as the tuple `T` with its fields mapped by the function `F`. For example,

```
local
   fun {Sq X} X*X end
in
  {MapTuple a(1 2 3) Sq}
end
```

should return `a(1 4 9)`.

Hint: A tuple is constructed with `{MakeTuple L N}`, where `L` is the label and `N` is the width of the tuple.

Exercise 4. (Tupled Recursion) Consider the following program code to compute a list of factorials.

```
fun {FactList N}
    if N==0 then nil
    else {Fact N}|{FactList N-1} end
end
fun {Fact N} if N==0 then 1 else N*{Fact N-1} end end
```

Analyse the complexity of the `FactList` function in terms of the number of multiplications performed. By computing both `{Fact N}#{FactList N}` simultaneously, define a new tupled function that has a more efficient computation with fewer multiplications.