Types: Addendum

This failed to make the handout cut, but there are some additional facts regarding types in MLg. The precedence of the types is basically the same as SML, resulting in the following ordering from tightest binding to least:

Int = Bool = Unit
*t1 * t2*
t1 + t2
t1 -> t2

This order results in type definitions like
*t1 -> t2 * t3*
to be resolved as
*t1 -> (t2 * t3)*

Notice that all terminal types have the same precedence. There is no conflict because terminal types always bind to the corresponding unique tokens.

Additionally, we must specify associativity for chained types, which answers the question of whether
t1 -> t2 -> t3
is equivalent to
t1 -> (t2 -> t3)
or
(t1 -> t2) -> t3

And similarly for all compound types. In the tradition of SML, we define functions to be **right associative**, which means:
t1 -> t2 -> t3 = t1 -> (t2 -> t3)

Tuples, because of the way n-tuples are parsed into binary tuples are **left associative** as are unions.
*t1 * t2 * t3 = (t1* t2) * t3*

When in doubt, use parentheses to indicate the parse order.