Grammar

```
         <program> ::= <type definitions> program = <term>

<type definitions> ::= <type definition> |
                       <type definition> <type definitions>

 <type definition> ::= (type <type variable> = <constr> | ... | <constr>)

          <constr> ::= <constr variable> | <constr variable> of <type>

            <type> ::= unit | bool | i64 | <type> * ... * <type> |
                       <type variable>

         <literal> ::= () | false | true | -9223372036854775808 | ... |
                       9223372036854775807

           <value> ::= <literal> | <variable> | (<value>, ..., <value>)

         <pattern> ::= <variable> | (<pattern>, ..., <pattern>)

            <expr> ::= <value> | let <pattern> = <iso> <pattern> in <expr>

             <iso> ::= add | sub | negate |
                       (iso <value> <-> <expr> | ... | <value> <-> <expr>) |
                       fun <iso variable> -> <iso> | <iso variable> |
                       <iso> <iso>

            <term> ::= <literal> | <variable> | (<term>, ..., <term>) |
                       <iso> <term> | let <pattern> = <term> in <term>
```

Typing Rules - Terms

$$\frac{}{\Psi; \emptyset \vdash () : \mathtt{unit}} \qquad \frac{}{\Psi; x : A \vdash x : A} \qquad \frac{\Psi; \Delta_1 \vdash t_1 : A_1 \quad \ldots \quad \Psi; \Delta_n \vdash t_n : A_n}{\Psi; \Delta \vdash (t_1, \ldots, t_n) : A_1 \otimes \ldots \otimes A_n}$$

$$\frac{\Psi \vdash_\omega \omega : A \leftrightarrow B \qquad \Psi; \Delta \vdash t : A}{\Psi; \Delta \vdash \omega\, t : B}$$

$$\frac{\Psi; \Delta_1 \vdash t_1 : A_1 \otimes \ldots \otimes A_n \qquad \Psi; \Delta_2 \vdash x_1 : A_1, \ldots, x_n : A_n \vdash t_2 : B}{\Psi; \Delta_1, \Delta_2 \vdash \mathtt{let}\ (x_1, .., x_n) = t_1\ \mathtt{in}\ t_2 : B}$$

Typing Rules - Isos

$$\frac{}{\Psi; \phi : T \vdash () : \mathtt{unit}} \qquad \frac{}{\Psi; x : A \vdash x : A}$$