Tiger Typing Rules

1. Implicit type conversion and promotion only applies to base types (int -> float, but not in reverse), not to derived types, and the final result assumes the type after type conversion. It applies to arithmetic operators, assignment operator as well as compactors.
e.g.,
type MY_FLOAT = float; var x: MY_FLOAT; x := x + 4; /* illegal, type conversion not applied to derived types */

2. We are allowing only literals to be type converted at initialization point of variables if the types of the variables are 1st level derived types out of base types (int and float).
e.g.,
type FIRST_INT = int; var x: FIRST_INT := 0; /* legal */
x "= x + 1; /* illegal at any other points */
type SECOND_INT = FIRST_INT; var y: SECOND_INT := 0; /* illegal, SECOND_INT is not 1st level derived type of int */

3. Name type equivalence says variables a and b can be mixed in computation if a and b can be made compatible by type conversion as in (1) or are of the same type
e.g.,
type INT_TYPE_1 = int; var x: INT_TYPE_1;
type INT_TYPE_2 = int; var y: INT_TYPE_2;
x := x + y; /* illegal */