Report of Implementing Liquid Type Check

Lishun Su fc56375

To implement this functionality, I first modified the parser. When we declare a variable, I added an extra optional block named `liquid_type`, which contains the conditions for the checker. Then, I created a new class named `Checker`, which has its own context and other necessary information to perform the check.

Finally, I integrated the checker into the type checker. When a variable is declared, it saves the information to its own context, and during assignment, it performs the necessary checks.

The challenges in the implementation include defining the structure to represent the `liquid_type` block. I ultimately chose the format (Expression).

The second challenge was implementing the `Checker` class. Specifically, I needed to determine whether it required a new context or could use the context of the type checker. I also had to decide what kind of information needed to be stored in the context and how to process incoming data, as the value of an assignment could be an expression, which needed validation.

Finally, the `Checker` has its own context, where it stores the type, the value, a Z3 pointer, the condition, and a list of other variables where the condition is used. For validation, I chose the Z3 solver, which greatly simplifies the validation process.

Another challenge was determining whether the condition could include other variables (e.g., `var b: int (a > 0): 1;`) and, if so, how to implement it. The solution was to add a list to the context of the checker that contains the names of the variables used in the condition.