The following rules must be fulfilled by the type-checking phase.

For expressions:

- The +, -, * and / binary operators can be applied to two characters (returning an integer), two integers (returning int) and two doubles (returning double).
- The % operator can only be applied to two characters and two integers. Int is returned.
- Logical operators (&&, || and !) can only be applied to integers and return int.
- Comparison operators (>, <, >=, <=, == and !=) are applicable to two chars, two integers or two doubles, and they always return int.
- Unary minus can be applied to char (returning int), int (returning int) and double (returning double).
- The cast operator can only be applied to built-in types (void is not included).
- The [ ] operator requires the first operand to be an array and the second one to be an integer.
- The . operator can only be applied to records (structs). The name of the field must be defined in the record definition. The returned type is the field type.
- For function invocation, the number of arguments must be the same as the number of parameters. The type of each argument must be the same as the type of the corresponding parameter. Recall that parameter and return types must be built-in (void is only valid for the return type).
- Any other combination represents a semantic error.

For statements:

- Two expressions can be assigned if they have the same built-in type (void is not included) and the left-hand side expression is an l-value.
- Only built-in types can be read and written. Expressions to be read must be l-values.
- The conditions in while and if statements must be int.
- The type of the returned expression must be the same as the return type declared in the enclosing function.