**Q4:**

**How Function Works:**

During this stage, we store the value of a variable in memory, and each variable is displayed on the data grid view. Our class encompasses a variable and a list of class variables.

To begin, in the rule1 function, a flag (denoted as "f") is set to 1. This flag serves to indicate that the identifier is an integer.

Within the rule 2 function, we initialize the "firstVar" variable to capture the variable preceding the "=" symbol, and the "prevVar" variable is employed to capture the initial variable following the "=" symbol.

In this condition the variable becomes any integer value

For example, in the condition "int x=3;," the integer variable "x" assumes the value of another integer variable. Similarly, in the condition "int x= y;," the variable "x" takes on the value of the variable "y."

In this scenario, we capture the numerical value preceding the operator. The identified number then takes on the role of this numerical value.

In this situation, we capture the variable preceding the operator. The variable designated as "prevVar" then assumes the role of this identified variable.

In this scenario, we extract the values associated with the "+" operator in the following two cases:

* When the expression involves a numerical value added to another numerical value (Num + Num).
* When an integer variable is added to a numerical value (Integer variable + Num).

In this situation, we extract the values related to the "-" operator in the following two cases:

* When the expression involves subtracting a numerical value from another numerical value (Num - Num).
* When subtracting a numerical value from an integer variable (Integer variable - Num).

In this case, we extract the values associated with the "\*" operator in two specific instances:

* When the expression involves multiplying a numerical value by another numerical value (Num \* Num).
* When multiplying an integer variable by a numerical value (Integer variable \* Num).

In this circumstance, we capture the values corresponding to the "/" operator in two distinct scenarios:

* When the expression involves dividing a numerical value by another numerical value (Num / Num).
* When dividing an integer variable by a numerical value (Integer variable / Num).

In this scenario, we extract the values associated with the "%" operator in two specific cases:

* When the expression involves calculating the modulus of a numerical value by another numerical value (Num % Num).
* When calculating the modulus of an integer variable by a numerical value (Integer variable % Num).

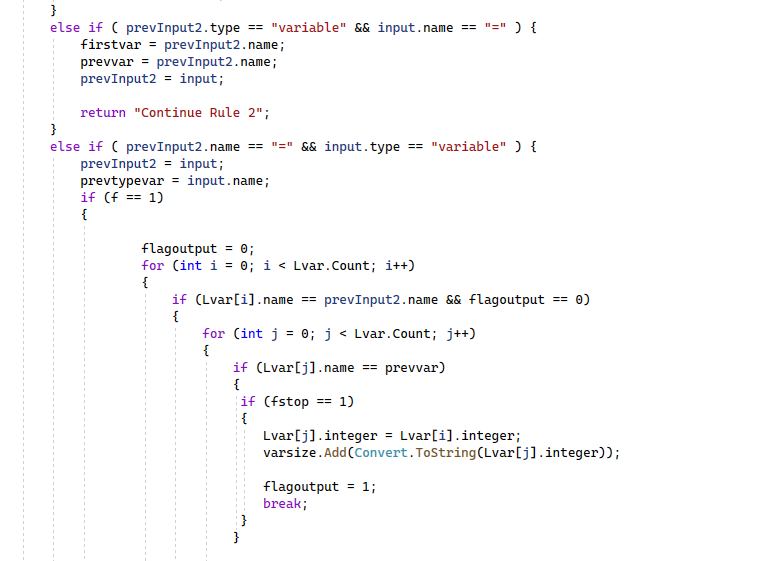
In this condition, we consider the values associated with the "+" operator when it involves the addition of two integer variables (Integer variable + Integer variable).

In this case, we consider the values associated with the "-" operator when it involves subtracting one integer variable from another (Integer variable - Integer variable).

In this scenario, we consider the values associated with the "\*" operator when it involves the multiplication of two integer variables (Integer variable \* Integer variable).

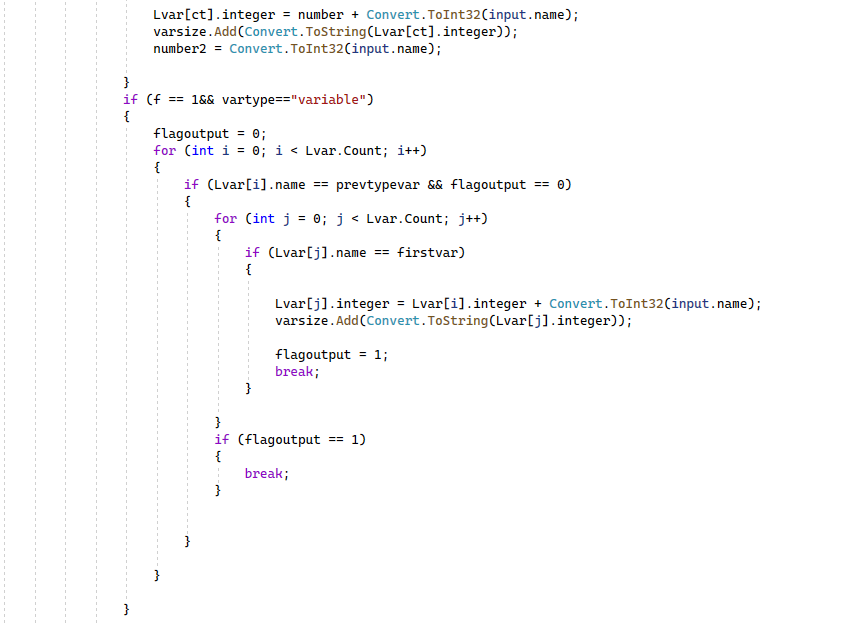
n this situation, we consider the values associated with the "/" operator when it involves dividing one integer variable by another (Integer variable / Integer variable).

In this circumstance, we consider the values associated with the "%" operator when it involves calculating the modulus of one integer variable by another (Integer variable % Integer variable).







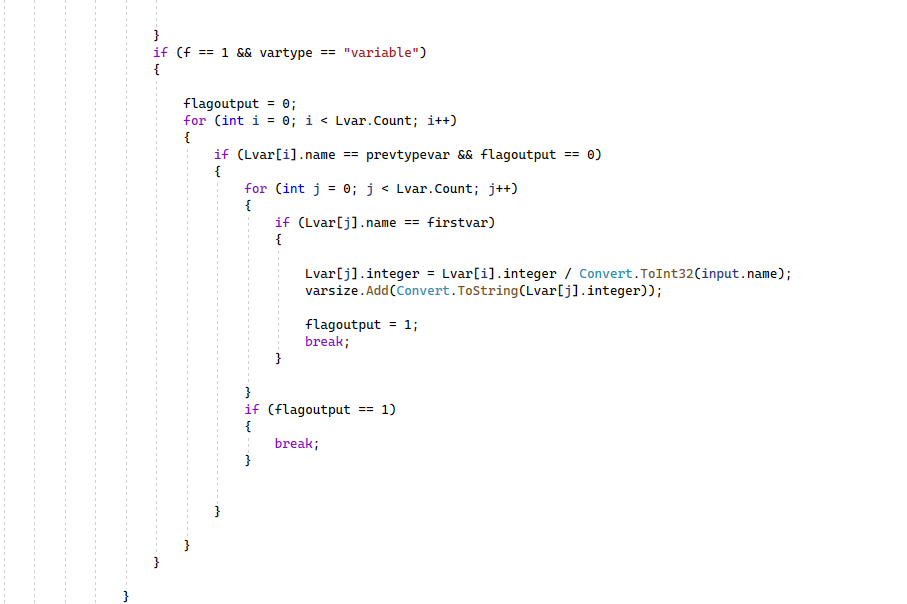








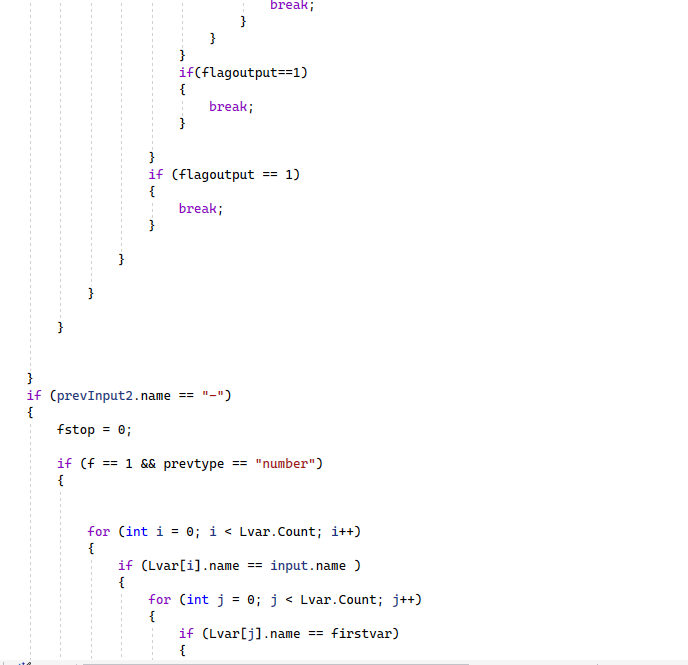


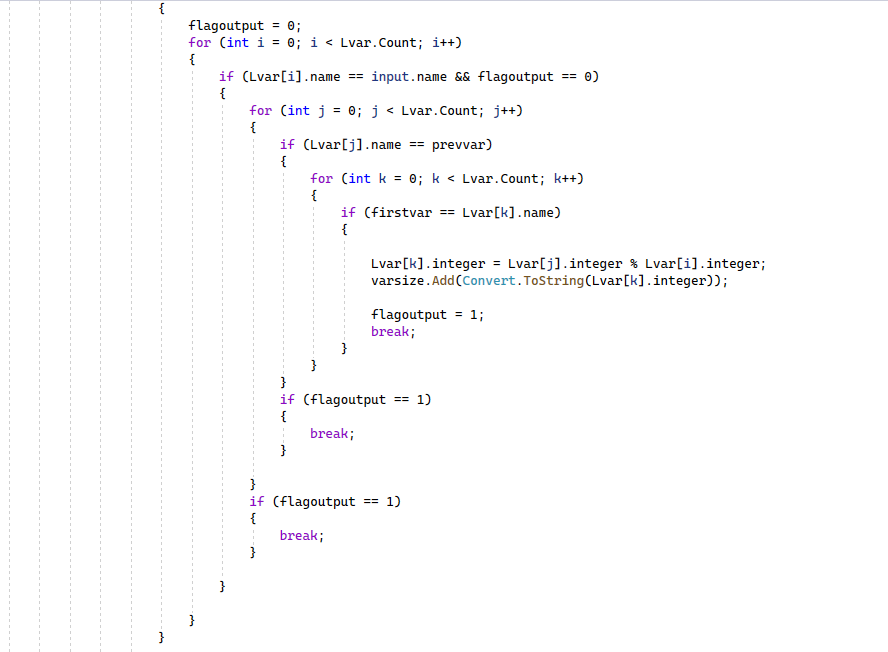
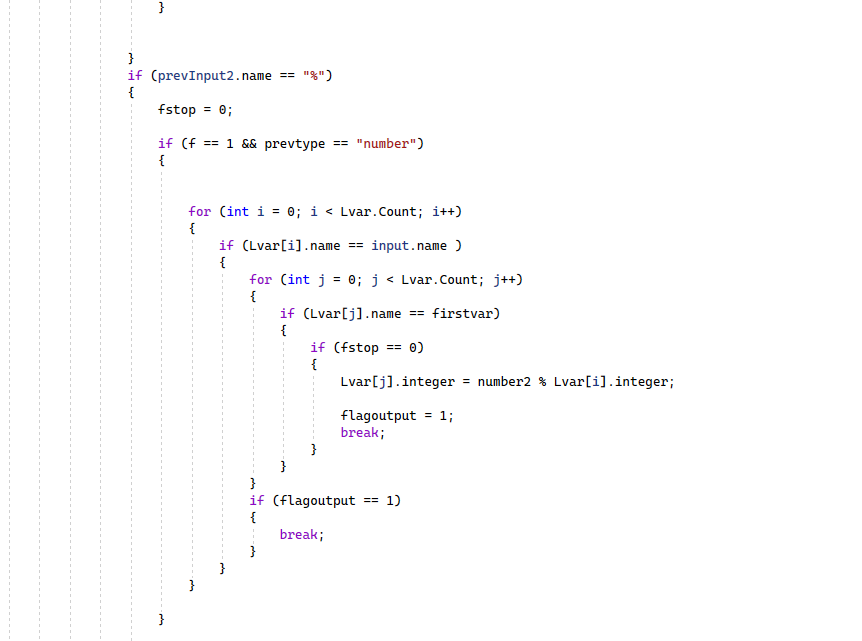
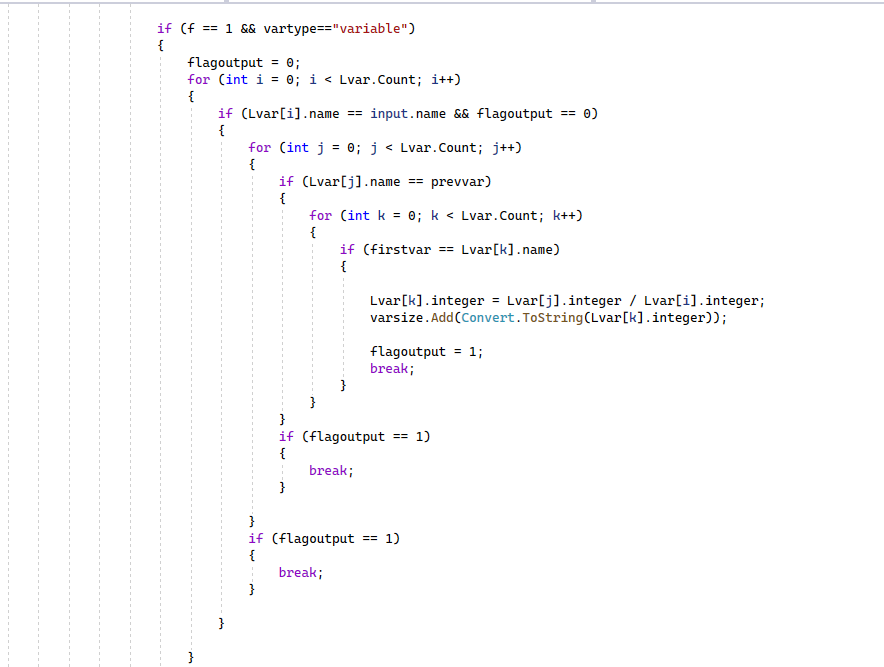
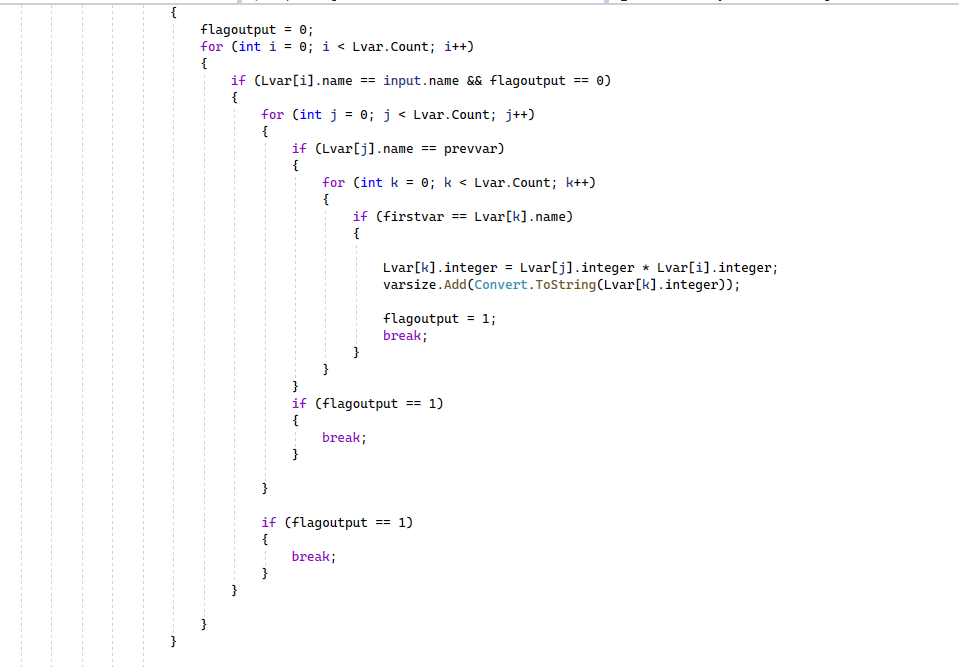
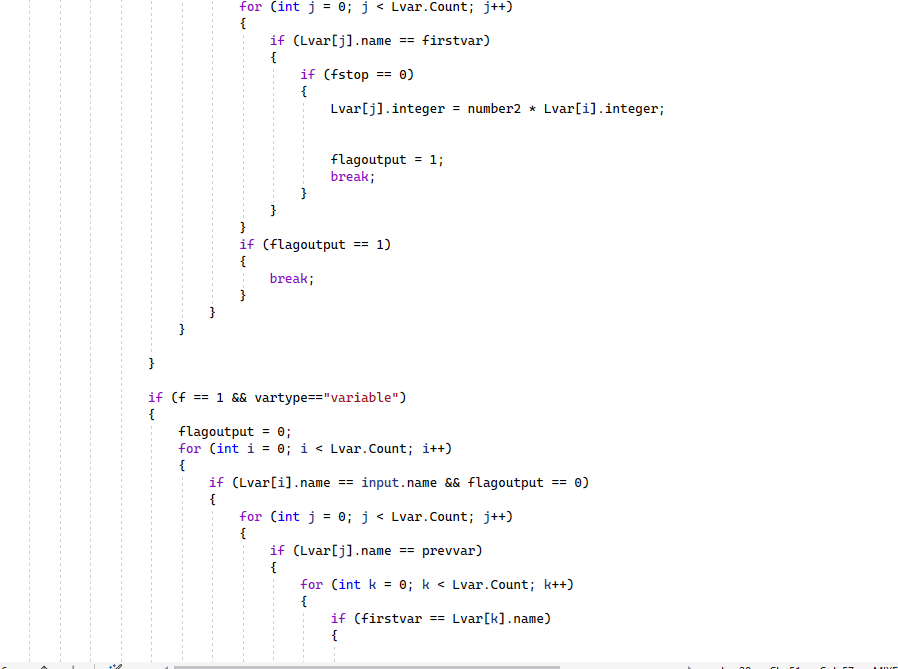
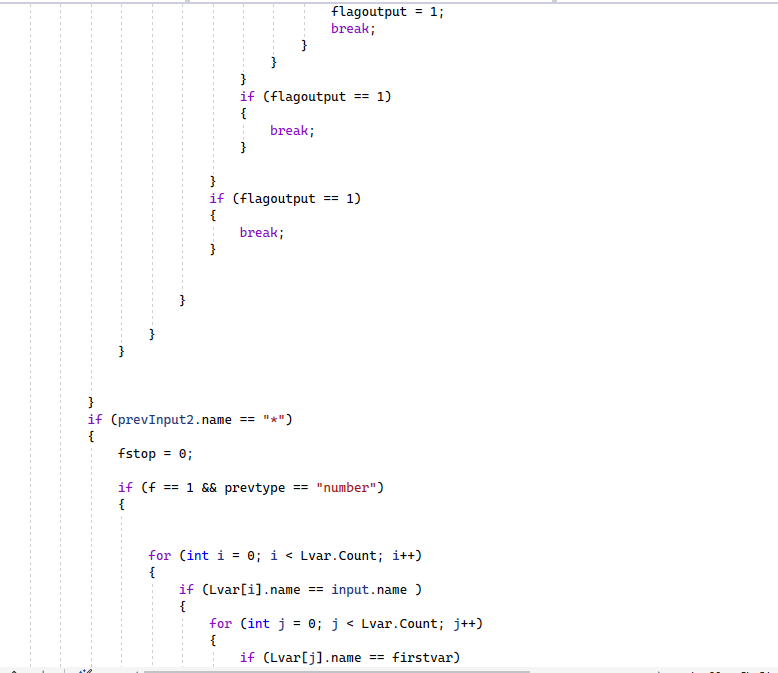


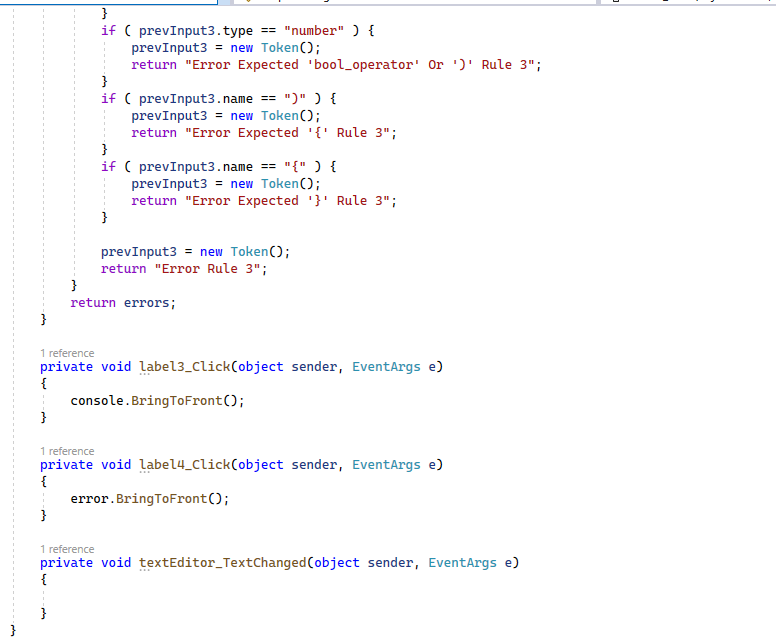




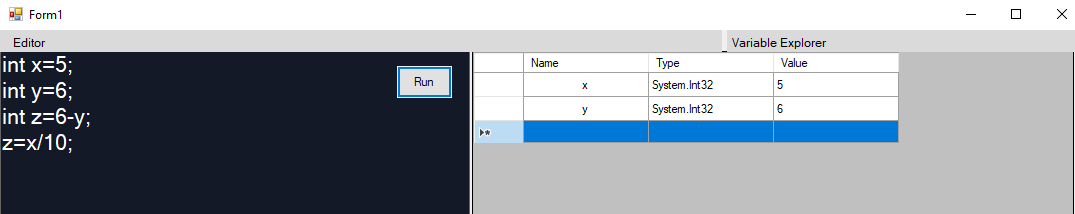








**Test Case:**

****