Programming Assignment V
Evaluation of Arithmetic Expression

Due Date: 2022/11/15 (40) points

1 Description of Assignment

Design and implement efficient data structure and algorithm for the evaluation of arithmetic expressions and assignment statement.

2 Input

For the first part (arithmetic expressions) the input data includes many arithmetic expressions ($\mathbf{E}$), such as "$1+2*3$", "$(1+2)*3$", etc. These expressions are defined by the grammar $G = (V, T, P, \mathbf{E})$, where $V = \{\mathbf{E}, \mathbf{T}, \mathbf{F}\}$, $T = \{\mathbf{a}\}$, and P is the following rules:

1. $\mathbf{E} \rightarrow \mathbf{E} + \mathbf{T}$
2. $\mathbf{E} \rightarrow \mathbf{E} - \mathbf{T}$
3. $\mathbf{E} \rightarrow \mathbf{T}$
4. $\mathbf{T} \rightarrow \mathbf{T} * \mathbf{F}$
5. $\mathbf{T} \rightarrow \mathbf{T}/\mathbf{F}$
6. $\mathbf{T} \rightarrow \mathbf{F}$
7. $\mathbf{F} \rightarrow \mathbf{a}$
8. $\mathbf{F} \rightarrow (\mathbf{E})$

In the above grammar, "**a**" means a constant, such as 3.14, 12, etc. Assume that all data are double in C, C++ and Java.

For the second part, in addition to the arithmetic expressions $\mathbf{E}$ defined above, your program should be able to handle assignment statement, such as $a = (x+3)*c$. The grammar for assignment statement is $G = (V', T', P', \mathbf{A})$, where $V' = \{\mathbf{A}, \mathbf{E}, \mathbf{T}, \mathbf{F}\}$ ad P' is the rules defined above, plus

0. $\mathbf{A} \rightarrow \mathbf{v} = \mathbf{E}$.

Note that the variable $\mathbf{F}$ now includes variables. Rule 7 is replaced by the following 2 rules:

- $\mathbf{F} \to \mathbf{a}$
- $\mathbf{F} \to \mathbf{v}$

For simplicity, a variable **v** can only be one of the 26 lower case letters $a, b, \ldots, z$. The value of a valuable in an expression comes from assignment statement. For example, if $a = 3.1415926$ and $x = a + 2$, then $x = 5.1415926$.

Your program needs to parse the input data in infix format. For example, if the input is $a = 3.1415926$, your program should first divide the input into 3 parts: "a", "=", and "3.1415926", and then recognize that the input is the assignment $a = 3.1415926$. Any parsing method can be used. However, a simple and efficient algorithm, such as Dijkstra's shunting yard algorithm, is preferred for this programming assignment.

Note that each expression or assignment is written in a line. There may be many input lines. Your program should be able to process the input file line by line, and terminates when there are no more input lines.

For the constants, for simplicity, you can assume that they are all positive numbers.

In addition to the above requirements, you may add more features, for example, negative numbers, complex numbers, exponentiation, trigonometric functions, logarithms, etc.

3 Output

For each input line which contains one arithmetic expression, print out the value of the expression. For example, on input

```
(1 + 2) * (3 - 4) / 5
```

print out

```
-0.6
```

in a line. If the input is an assignment, for example,

```
a = 1 + 2 * 3
b = 4 * a + 5
```

print out

```
a = 7
b = 33
```

If a variable is used but not defined in the previous assignment, print out an error message.

Notes

The format of the report of the assignment does not need to be very formal, but it should be close to the format of a research technical report.

1. Title and author.
   Include assignment number, *your name*, *student number* and *email address* on the *first* page of your report.

2. Statement of the problem.
   A "formal" description of the problem in this assignment. In addition to the basic requirements specified in the assignment, emphasize the functions or features you implemented.

3. Main results.
   This section should include at least the following items.

   (a) Description of the design of your program and the data structures used in your program.

   (b) List of your program with comments. If your program is very long, list only the main parts of the program here and the entire program in an appendix. Additional comments can be added manually to explain the design of the program.

   (c) Outputs of the compilation and the executions of your program.

4. Conclusions
   Give a brief summary of what you did, and interesting thing you learned from this assignment.

Additional notes:

1. Turn in your report on or before due day.

2. The output of the program execution should indicate the correctness of your program. In other words, a set of comprehensive (but not necessarily exhaustive) annotated test data for the problem should be provided to show that your program is indeed correct. This can be done by carefully selecting a set of test data.

3. Print or write the report on A4 papers. Bind them together in the upper left corner.