Solution to Homework 14

Tasks: Show, step by step:

1. how the stack-based algorithm will transform the expression $(3/1)-(1\cdot 3)$ into a postfix expression, and then
2. how a second stack-based algorithm will compute the value of this postfix expression.

Solution to Task 1. We have an expression

$$(\ 3\ /\ 1\ )\ -\ (\ 1\ \cdot\ 3\ )$$

We start with an empty stack.

• First, we read the first symbol (:

$$\underline{(}\ 3\ /\ 1\ )\ -\ (\ 1\ \cdot\ 3\ )$$

This is an operation symbol, so we push it into the stack. The stack now has the form

$$\boxed{(}$$

• Then, we read the next symbol 3:

$$(\ \underline{3}\ /\ 1\ )\ -\ (\ 1\ \cdot\ 3\ )$$

It is a number, so we copy it to the postfix expression, which now takes the form

$$3$$

The stack remains the same.

• After that, we read the next symbol /:

$$(\ 3\ \underline{/}\ 1\ )\ -\ (\ 1\ \cdot\ 3\ )$$

It is an operation symbol, so we push it into the stack. Any operation inside the parentheses has to be performed before anything else. In priority terms, this means that any operation has higher priority than the opening parenthesis. Thus, we do not pop (from the stack. The postfix expression remains the same, and the stack takes the form

$$\begin{array}{|c|}\hline / \\ \hline (\\ \hline\end{array}$$

• Next, we read the symbol 1:

$$(\ 3\ /\ \underline{1}\ )\ -\ (\ 1\ \cdot\ 3\ )$$

This symbol is a number, so we add it to the postfix expression, which now takes the form

3 1

The stack remains the same.

• Next, we read the closing parenthesis):

$$(\ 3\ /\ 1\ \underline{)}\ -\ (\ 1\ \cdot\ 3\ )$$

This is an operation symbol, so we push it into the stack. Every operation in parentheses has to be performed before everything else. In priority terms, this means that any operation has higher priority than). So, before pushing, we pop the top symbol / from the stack into the postfix expression, which will now have the form

3 1 /

Now, the stack has the following form:

$$\begin{array}{|c|}\hline) \\ \hline (\\ \hline\end{array}$$

This means, in effect, that we have an expression (), i.e., we have nothing to do. Thus, if we have these two symbols on top of the stack, they can both be canceled. When we cancel them, the stack will be empty.

• Next, we read the minus:

$$(\ 3\ /\ 1\ )\ \underline{-}\ (\ 1\ \cdot\ 3\ )$$

This is an operation symbol, so we push it into the stack. The stack was empty, so there is nothing to pop. Thus, the stack has the following form:

$$\boxed{-}$$

• Next, we read the second opening parenthesis:

$$(\ 3\ /\ 1\ )\ -\ \underline{(}\ 1\ \cdot\ 3\ )$$

This is an operation symbol, so we push it into the stack. Everything inside the parentheses is performed first, so this means that (has higher priority that any symbol before it. So, we do not pop anything from the stack, and the stack now has the form

$$\begin{array}{|c|}\hline (\\ \hline - \\ \hline\end{array}$$

• Next, we read the next symbol 1:

$$(\ 3\ /\ 1\ )\ -\ (\ \underline{1}\ \cdot\ 3\ )$$

This symbol is a number, so we add it to the forming postfix expression:

$$3\ 1\ /\ 1$$

The stack remains the same.

• Then, we read the next symbol $\cdot$:

$$(\ 3\ /\ 1\ )\ -\ (\ 1\ \underline{\cdot}\ 3\ )$$

It is an operation symbol, so we push it into the stack. Any operation inside the parentheses has to be performed before anything else. In priority terms, this means that any operation has higher priority than the opening parenthesis. Thus, we do not pop (from the stack. The postfix expression remains the same, and the stack takes the form

$$\begin{array}{|c|}\hline \cdot \\ \hline (\\ \hline - \\ \hline\end{array}$$

• Next, we read the symbol 3:

$$(\ 3\ /\ 1\ )\ -\ (\ 1\ \cdot\ \underline{3}\ )$$

This symbol is a number, so we add it to the postfix expression, which now takes the form

$$3\ 1\ /\ 1\ 3$$

The stack remains the same.

• Finally, we read the second closing parenthesis):

$$(\ 3\ /\ 1\ )\ -\ (\ 1\ \cdot\ 3\ \underline{)}$$

This is an operation symbol, so we push it into the stack. Every operation in parentheses has to be performed before everything else. In priority terms, this means that any operation has higher priority than). So, before pushing, we pop the top symbol $\cdot$ from the stack into the postfix expression, which will now have the form

$$3\ 1\ /\ 1\ 3\ \cdot$$

Now, the stack has the following form:

$$\begin{array}{|c|}\hline) \\ \hline (\\ \hline - \\ \hline\end{array}$$

This means, in effect, that we have an expression (), i.e., we have nothing to do. Thus, if we have these two symbols on top of the stack, they can both be canceled. So, the stack takes the form:

$$\boxed{-}$$

• We have finished reading the expression, we are at end-of-line, and everything before that has to be performed first. So, the subtraction operation has higher priority and thus, will be popped and added to the postfix expression. So, the stack is empty, and we get the desired postfix expression:

$$3\ 1\ /\ 1\ 3\ \cdot\ -$$

This is our solution to Task 1:

Final solution to Task 1: $3\ 1\ /\ 1\ 3\ \cdot -$

Solution to Task 2. Let us show how a stack-based algorithm enables us to compute the value of this postfix expression

$$3\ 1\ +\ 1\ 3\ -\ \cdot$$

• First, we read the first symbol 3 of this expression $\underline{3}\ 1\ /\ 1\ 3\ \cdot -$. This symbol is a number, so we push it into the stack. Now, the stack takes the form

$$\boxed{3}$$

• Next, we read the next symbol 1 of this expression $3\ \underline{1}\ /\ 1\ 3\ \cdot -$. This symbol is also a number, so we also push it into the stack. Now, the stack takes the form

$$\begin{array}{|c|}\hline 1 \\ \hline 3 \\ \hline\end{array}$$

• The next symbol that we read is the division symbol $3\ 1\ \underline{/}\ 1\ 3\ \cdot -$So:

- we pop the two top numbers from the stack: 1 and 3,
- we apply the division operation to the numbers 3 and 1, i.e., compute $3/1 = 3$, and
- we push 3 into the stack.

Now, the stack takes the form

$$\boxed{3}$$

• Then, we read the next symbol 1 of this expression $3\ 1\ /\ \underline{1}\ 3\ \cdot -$. This symbol is a number, so we also push it into the stack. Now, the stack takes the form

$$\begin{array}{|c|}\hline 1\\ \hline 3\\ \hline\end{array}$$

• Next, we read the next symbol 3 of this expression $3\ 1\ /\ 1\ \underline{3}\ \cdot -$. This symbol is a number, so we also push it into the stack. Now, the stack takes the form

$$\begin{array}{|c|}\hline 3\\ \hline 1\\ \hline 3\\ \hline\end{array}$$

• Next, we read the multiplication symbol $3\ 1\ /\ 1\ 3\ \underline{\cdot} -$ So:

- we pop the two top numbers from the stack: 1 and 3,
- we apply the multiplication operation to the numbers 1 and 3, i.e., compute $1 \cdot 3 = 3$, and
- we push the resulting number 3 into the stack.

Now, the stack takes the form

$$\begin{array}{|c|}\hline 3\\ \hline 3\\ \hline\end{array}$$

• Finally, we read the minus: $3\ 1\ /\ 1\ 3\ \cdot \underline{-}$. So:

- we pop the two top numbers from the stack: 3 and 3,
- we apply the subtraction operation to the numbers 3 and 3, i.e., compute $3 - 3 = 0$, and
- we push the resulting number 0 into the stack.

Now, the stack takes the form

$$\boxed{0}$$

• We have read all the symbols, so the number 0 that we have in the stack is the result of our computation of the original $(3/1) - (1 \cdot 3)$ for which $3\ 1\ /\ 1\ 3\ \cdot\ -$ is the corresponding postfix form.