Solution to Homework 9

**Background.** In Problem 7, we considered a grammar with rules $N \rightarrow L, N \rightarrow NL, N \rightarrow ND, L \rightarrow a, D \rightarrow 0$, and $D \rightarrow 1$.

**Tasks:**

1. Use a general algorithm to construct a (non-deterministic) pushdown automaton that corresponds to context-free grammar described in Problem 7.

2. Show, step by step, how the word $a01$ will be accepted by this automaton.

**Solution to Task 1.** By using the general algorithm, we get the following pushdown automaton:
Solution to Task 2. Let us show how this is done on the example of the word +110 generated by the above automaton:

$$N \rightarrow ND \rightarrow NDD \rightarrow aDD \rightarrow a0D \rightarrow a01.$$  

To make this derivation clearer, let us mark the variables corresponding to different transitions by subscripts:

$$N_1 \rightarrow N_2D_1 \rightarrow N_3D_1D_2 \rightarrow aD_1D_2 \rightarrow a0D_2 \rightarrow a01.$$  

Let us now trace what our pushdown automaton will do.
We start in the state $s$ with an empty stack:
The only thing we can do when in the state $s$ is push the dollar sign into the stack and get to the intermediate state $i$:

The contents of the stack is as follows:

$\$
When we are in the state $i$, the only thing we can do is push the starting variable $N$ into the stack and go into the working state $w$;

Now, the stack contains the starting variable on top of the dollar sign:
Now that we are in the working state, we can start following the rules that were used to derive the word $a01$. The first rule was $N \rightarrow ND$, or, to be precise, $N_1 \rightarrow N_2D_1$. As we have mentioned, this rule is implemented in two steps:

- first, we pop $N$ and push the last symbol of the right-hand side – in this cases, the symbol $D$ into the stack, getting into the auxiliary state $a_2$;
- then, we push $N$ into the stack, and go back to the working state $w$.

Let us illustrate this step by step.

First, we pop $N$, push $D$, and go into the state $a_2$:

The stack will now have $D$ instead of the original $N$:
Then, we push $N$ into the stack and go back to working state $w$:

The stack will now have $N$ on top of its previous contents:

$$\begin{array}{|c|}
\hline
N \\
D \\
\$ \\
\hline
\end{array}$$
Now, the symbol $N$ is on top of the stack, so we again use the rule $N \rightarrow ND$: first, we replace $N$ by $D$ and go to the state $a_2$, then push $N$ and go back to the state $w$:

Now, the stack will have $D$ instead of $N$:

$$
\begin{array}{c}
D \\
D \\
\$ \\
\end{array}
$$
and

Now, the stack will have \( N \) on top:

\[
\begin{array}{c}
N \\
D \\
D \\
$ 
\end{array}
\]
Next, we use the rule $N \rightarrow L$, i.e., replace $N$ with $L$ on top of the stack:

Now, the stack will have $L$ on top:
Next, we use the rule $L \rightarrow a$, i.e., replace $L$ with $a$ on top of the stack:

Now, the stack will have $a$ on top:
Now, we have a terminal symbol $a$ on top of the stack. In this case, the only thing we can do is use the rule $a, a \rightarrow \varepsilon$: read $a$ and pop $a$ from the top of the stack:

Now, the stack will have the following form:

```
D
D
D
$`
Next, we use the rule $D \rightarrow 0$, i.e., replace $D$ with 0 on top of the stack:

Now, the stack will have 0 on top:

$$
\begin{array}{c}
0 \\
D \\
$ \\
\end{array}
$$
Now, we have a terminal symbol 0 on top of the stack, so the only thing we can do is to use the rule $0, 0 \rightarrow \varepsilon$, i.e., read 0 and pop 0 from the stack:

Now, the stack will have the following form:

$$
\begin{array}{c}
D \\
\$ \\
\end{array}
$$
Then, we use the rule $D \rightarrow 1$, i.e., we replace $D$ with 1 on top of the stack:

Now, the stack will have 1 on top:
Now, there is a terminal symbol 1 on top of the stack, so we have no other choice but to use the rule $1, 1 \rightarrow \varepsilon$:

Now, the stack will only have the dollar sign:

$$\$"
We read all the symbols, and the only symbol in the stack is the dollar sign. We can thus go to the final state:

The stack is now empty. We are in the final state with the empty stack, so the word $a01$ is accepted.
A graphical description of the transitions.

<table>
<thead>
<tr>
<th>state</th>
<th>s</th>
<th>i</th>
<th>w</th>
<th>a_2</th>
<th>w</th>
<th>a_2</th>
<th>w</th>
<th>w</th>
<th>w</th>
</tr>
</thead>
<tbody>
<tr>
<td>stack</td>
<td>$</td>
<td>N</td>
<td>D</td>
<td>N</td>
<td>D</td>
<td>N</td>
<td>L</td>
<td>a</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>$</td>
<td>$</td>
<td>$</td>
<td>$</td>
<td>$</td>
<td>$</td>
<td>$</td>
<td>$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>read</th>
<th>a</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>w</td>
<td>w</td>
<td>w</td>
</tr>
<tr>
<td>stack</td>
<td>D</td>
<td>D</td>
<td>1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>read</th>
<th>a</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>state</td>
<td>w</td>
<td>w</td>
<td>w</td>
</tr>
<tr>
<td>stack</td>
<td>D</td>
<td>D</td>
<td>$</td>
</tr>
</tbody>
</table>