МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет ИТМО»

Факультет программной инженерии и компьютерной техники

Домашняя работа №3

Вариант 10

Выполнил:

Круглов Егор Ильич, P3324

Преподаватель:

Лаздин Артур Вячеславович

Санкт-Петербург

2025

Выполнение

1. S → abAC | acAB | aa

2. B → BBb | Bbb | b

3. A → aAA | aCC | bB | b

4. C → cc | cC

Убираем левую рекурсию для 2 правила (для B):

$B \to bB_1$

$B_1 \to BbB_1 \mid bbB_1 \mid \varepsilon$

Сделаем левую факторизацию для S:

$S \to aS_1$

$S_1 \to bAC \mid cAB \mid a$

Сделаем левую факторизацию для A:

$A \to aA_1 \mid bA_2$

$A_1 \to AA \mid CC$

$A_2 \to B \mid \varepsilon$

Сделаем левую факторизацию для C:

$C \to cC_1$

$C_1 \to c \mid C$

По итогу получаем:

1. $S \to aS_1$

2. $S_1 \to bAC \mid cAB \mid a$

3. $B \to bB_1$

4. $B_1 \to BbB_1 \mid bbB_1 \mid \varepsilon$

5. $A \to aA_1 \mid bA_2$

6. $A_1 \to AA \mid CC$

7. $A_2 \to B \mid \varepsilon$

8. $C \to cC_1$

9. $C_1 \to c \mid C$

Символ	nullable	FIRST	FOLLOW
S	false	$\{a\}$	$\{\$\}$
S_1	false	$\{a, b, c\}$	$\{\$\}$
B	false	$\{b\}$	$\{a, b, c, \$\}$
B_1	true	$\{b\}$	$\{a, b, c, \$\}$
A	false	$\{a, b\}$	$\{a, b, c\}$
A_1	false	$\{a, b, c\}$	$\{a, b, c\}$
A_2	true	$\{b\}$	$\{a, b, c\}$
C	false	$\{c\}$	$\{a, b, c, \$\}$
C_1	false	$\{c\}$	$\{a, b, c, \$\}$

	a	b	c	\$
S	$S \to aS_1$			
S_1	$S_1 \to a$	$S_1 \to bAC$	$S_1 \to cAB$	
B		$B \to bB_1$		
B_1	$B_1 \to \varepsilon$	$B_1 \to BbB_1 \mid bbB_1 \mid \varepsilon$	$B_1 \to \varepsilon$	$B_1 \to \varepsilon$
A	$A \to aA_1$	$A \to bA_2$		
A_1	$A_1 \to AA$	$A_1 \to AA$	$A_1 \to CC$	
A_2	$A_2 \to \varepsilon$	$A_2 \to B \mid \varepsilon$	$A_2 \to \varepsilon$	
C			$C \to cC_1$	
C_1			$C_1 \to c \mid C$	

Видим, что у нас есть 3 конфликта, недающих грамматике быть LL(1). Изменим грамматику, чтобы их избежать:

1. $S \to aS_1$

2. $S_1 \to bAC \mid cAB \mid a$

3. $B \to bB_1$

4. $B_1 \to BbB_1 \mid abB_1 \mid c$

5. $A \to aA_1 \mid bA_2$

6. $A_1 \to AA \mid CC$

7. $A_2 \to B$

8. $C \to cC_1$

9. $C_1 \to a \mid C$

Символ	nullable	FIRST	FOLLOW
S	false	$\{a\}$	$\{\$\}$
S_1	false	$\{a, b, c\}$	$\{\$\}$
B	false	$\{b\}$	$\{a, b, c, \$\}$
B_1	false	$\{a, b, c\}$	$\{a, b, c, \$\}$
A	false	$\{a, b\}$	$\{a, b, c\}$
A_1	false	$\{a, b, c\}$	$\{a, b, c\}$
A_2	false	$\{b\}$	$\{a, b, c\}$
C	false	$\{c\}$	$\{a, b, c, \$\}$
C_1	false	$\{a, c\}$	$\{a, b, c, \$\}$

	a	b	c	\$
S	$S \to aS_1$			
S_1	$S_1 \to a$	$S_1 \to bAC$	$S_1 \to cAB$	
B		$B \to bB_1$		
B_1	$B_1 \to abB_1$	$B_1 \to BbB_1$	$B_1 \to c$	
A	$A \to aA_1$	$A \to bA_2$		
A_1	$A_1 \to AA$	$A_1 \to AA$	$A_1 \to CC$	
A_2		$A_2 \to B$		
C			$C \to cC_1$	
C_1	$C_1 \to a$		$C_1 \to C$	

Мы убрали эпсилон правила, а также заменили:

1) $B_1 \to BbB_1 \mid bbB_1 \mid \varepsilon$ на $B_1 \to BbB_1 \mid abB_1 \mid c$
2) $C_1 \to c \mid C$ на $C_1 \to a \mid C$

Также заметим, что у нас образовалось $A_2 \to B$. Уберем A_2, просто заменив его везде на B.

1. $S \to aS_1$

2. $S_1 \to bAC \mid cAB \mid a$

3. $B \to bB_1$

4. $B_1 \to BbB_1 \mid abB_1 \mid c$

5. $A \to aA_1 \mid bB$

6. $A_1 \to AA \mid CC$

7. $C \to cC_1$

8. $C_1 \to a \mid C$

Символ	nullable	FIRST	FOLLOW
S	false	$\{a\}$	$\{\$\}$
S_1	false	$\{a, b, c\}$	$\{\$\}$
B	false	$\{b\}$	$\{a, b, c, \$\}$
B_1	false	$\{a, b, c\}$	$\{a, b, c, \$\}$
A	false	$\{a, b\}$	$\{a, b, c\}$
A_1	false	$\{a, b, c\}$	$\{a, b, c\}$
C	false	$\{c\}$	$\{a, b, c, \$\}$
C_1	false	$\{a, c\}$	$\{a, b, c, \$\}$

	a	b	c	$\$$
S	$S \to aS_1$			
S_1	$S_1 \to a$	$S_1 \to bAC$	$S_1 \to cAB$	
B		$B \to bB_1$		
B_1	$B_1 \to abB_1$	$B_1 \to BbB_1$	$B_1 \to c$	
A	$A \to aA_1$	$A \to bB$		
A_1	$A_1 \to AA$	$A_1 \to AA$	$A_1 \to CC$	
C			$C \to cC_1$	
C_1	$C_1 \to a$		$C_1 \to C$	

Теперь у нас действительно таблица LL(1) анализатора.

Код:

Ссылка на github: https://github.com/KruglovEgor/Compilers/blob/main/lab3/main.py

```
def ll1_parser(input_string):
    parse_table = {
        'S': {'a': ['a', 'S_1']},
        'S_1': {'a': ['a'], 'b': ['b', 'A', 'C'], 'c': ['c', 'A', 'B']},
        'B': {'b': ['b', 'B_1']},
        'B_1': {'a': ['a', 'b', 'B_1'], 'b': ['B', 'b', 'B_1'], 'c': ['c']},
        'A': {'a': ['a', 'A_1'], 'b': ['b', 'B']},
        'A_1': {'a': ['A', 'A'], 'b': ['A', 'A'], 'c': ['C', 'C']},
        'C': {'c': ['c', 'C_1']},
        'C_1': {'a': ['a'], 'c': ['C']}
    }

    stack = ['$', 'S']
    input_string = list(input_string) + ['$']
    pointer = 0

    print(f"Parsing '{''.join(input_string[:-1])}':")

    while stack:
        top = stack[-1]
        current_input = input_string[pointer]

        print(f"Stack: {stack}, Input: {input_string[pointer:]}")

        # Если верхний символ — терминал или конец строки
        if top in {'a', 'b', 'c', '$'}:
            if top == current_input:
```

```
                if top == '$':
                    print("Success! Input accepted.")
                    return True
                stack.pop()
                pointer += 1
            else:
                # Ошибка: промах по терминалу
                print(f"Error: Expected '{top}', but found '{current_input}' at
position {pointer}")
                return False

        # Если верхний символ — нетерминал
        elif top in parse_table:
            if current_input in parse_table[top]:
                stack.pop()
                production = parse_table[top][current_input]
                # Добавляем правило в стек в обратном порядке
                for symbol in reversed(production):
                    stack.append(symbol)
            else:
                # Ошибка: промах по нетерминалу
                expected = list(parse_table[top].keys())
                print(
                    f"Error: No rule for '{top}' with input '{current_input}' at
position {pointer}. Expected one of {expected}")
                return False
        else:
            print(f"Error: Unknown symbol '{top}' in stack")
            return False

    print("Error: Stack not empty after input exhaustion")
    return False


# Тестирование
positive_cases = [
    "aa",
    "abacaccaca",
    "acbbcbabc"
]

negative_cases = [
    "abcc",
    "acaab",
    "abc"
]

print("Testing positive cases:")
for test in positive_cases:
    ll1_parser(test)
    print("---------------------------------")

print("\nTesting negative cases:")
for test in negative_cases:
    ll1_parser(test)
    print("---------------------------------")
```

Вывод:

```
"E:\Program Files\pythonProject\venv\Scripts\python.exe" E:\GitHub\Compilers\lab3\main.py
Testing positive cases:
Parsing 'aa':
Stack: ['$', 'S'], Input: ['a', 'a', '$']
Stack: ['$', 'S_1', 'a'], Input: ['a', 'a', '$']
Stack: ['$', 'S_1'], Input: ['a', '$']
Stack: ['$', 'a'], Input: ['a', '$']
Stack: ['$'], Input: ['$']
Success! Input accepted.
--------------------------------
Parsing 'abacaccaca':
Stack: ['$', 'S'], Input: ['a', 'b', 'a', 'c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'S_1', 'a'], Input: ['a', 'b', 'a', 'c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'S_1'], Input: ['b', 'a', 'c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'A', 'b'], Input: ['b', 'a', 'c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'A'], Input: ['a', 'c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'A_1', 'a'], Input: ['a', 'c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'A_1'], Input: ['c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C', 'C'], Input: ['c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C', 'C_1', 'c'], Input: ['c', 'a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C', 'C_1'], Input: ['a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C', 'a'], Input: ['a', 'c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C'], Input: ['c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C_1', 'c'], Input: ['c', 'c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C_1'], Input: ['c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C'], Input: ['c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C_1', 'c'], Input: ['c', 'a', 'c', 'a', '$']
Stack: ['$', 'C', 'C_1'], Input: ['a', 'c', 'a', '$']
Stack: ['$', 'C', 'a'], Input: ['a', 'c', 'a', '$']
Stack: ['$', 'C'], Input: ['c', 'a', '$']
Stack: ['$', 'C_1', 'c'], Input: ['c', 'a', '$']
Stack: ['$', 'C_1'], Input: ['a', '$']
Stack: ['$', 'a'], Input: ['a', '$']
Stack: ['$'], Input: ['$']
Success! Input accepted.
```

```
Parsing 'acbbcbabc':
Stack: ['$', 'S'], Input: ['a', 'c', 'b', 'b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'S_1', 'a'], Input: ['a', 'c', 'b', 'b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'S_1'], Input: ['c', 'b', 'b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B', 'A', 'c'], Input: ['c', 'b', 'b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B', 'A'], Input: ['b', 'b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B', 'B', 'b'], Input: ['b', 'b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B', 'B'], Input: ['b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B', 'B_1', 'b'], Input: ['b', 'c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B', 'B_1'], Input: ['c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B', 'c'], Input: ['c', 'b', 'a', 'b', 'c', '$']
Stack: ['$', 'B'], Input: ['b', 'a', 'b', 'c', '$']
Stack: ['$', 'B_1', 'b'], Input: ['b', 'a', 'b', 'c', '$']
Stack: ['$', 'B_1'], Input: ['a', 'b', 'c', '$']
Stack: ['$', 'B_1', 'b', 'a'], Input: ['a', 'b', 'c', '$']
Stack: ['$', 'B_1', 'b'], Input: ['b', 'c', '$']
Stack: ['$', 'B_1'], Input: ['c', '$']
Stack: ['$', 'c'], Input: ['c', '$']
Stack: ['$'], Input: ['$']
Success! Input accepted.
--------------------------------

Testing negative cases:
Parsing 'abcc':
Stack: ['$', 'S'], Input: ['a', 'b', 'c', 'c', '$']
Stack: ['$', 'S_1', 'a'], Input: ['a', 'b', 'c', 'c', '$']
Stack: ['$', 'S_1'], Input: ['b', 'c', 'c', '$']
Stack: ['$', 'C', 'A', 'b'], Input: ['b', 'c', 'c', '$']
Stack: ['$', 'C', 'A'], Input: ['c', 'c', '$']
Error: No rule for 'A' with input 'c' at position 2. Expected one of ['a', 'b']
```

```
Parsing 'acaab':
Stack: ['$', 'S'], Input: ['a', 'c', 'a', 'a', 'b', '$']
Stack: ['$', 'S_1', 'a'], Input: ['a', 'c', 'a', 'a', 'b', '$']
Stack: ['$', 'S_1'], Input: ['c', 'a', 'a', 'b', '$']
Stack: ['$', 'B', 'A', 'c'], Input: ['c', 'a', 'a', 'b', '$']
Stack: ['$', 'B', 'A'], Input: ['a', 'a', 'b', '$']
Stack: ['$', 'B', 'A_1', 'a'], Input: ['a', 'a', 'b', '$']
Stack: ['$', 'B', 'A_1'], Input: ['a', 'b', '$']
Stack: ['$', 'B', 'A', 'A'], Input: ['a', 'b', '$']
Stack: ['$', 'B', 'A', 'A_1', 'a'], Input: ['a', 'b', '$']
Stack: ['$', 'B', 'A', 'A_1'], Input: ['b', '$']
Stack: ['$', 'B', 'A', 'A', 'A'], Input: ['b', '$']
Stack: ['$', 'B', 'A', 'A', 'B', 'b'], Input: ['b', '$']
Stack: ['$', 'B', 'A', 'A', 'B'], Input: ['$']
Error: No rule for 'B' with input '$' at position 5. Expected one of ['b']
--------------------------------
Parsing 'abc':
Stack: ['$', 'S'], Input: ['a', 'b', 'c', '$']
Stack: ['$', 'S_1', 'a'], Input: ['a', 'b', 'c', '$']
Stack: ['$', 'S_1'], Input: ['b', 'c', '$']
Stack: ['$', 'C', 'A', 'b'], Input: ['b', 'c', '$']
Stack: ['$', 'C', 'A'], Input: ['c', '$']
Error: No rule for 'A' with input 'c' at position 2. Expected one of ['a', 'b']
--------------------------------
```