МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«МИРЭА – Российский технологический университет»
РТУ МИРЭА

Институт Информационных Технологий
Кафедра Вычислительной техники

ОТЧЕТ О ВЫПОЛНЕНИИ ПРАКТИЧЕСКОЙ РАБОТЫ №5

«Реализовать простой анализатор JSON файла на Python»

по дисциплине
«Теория формальных языков»

Выполнил студент группы ИКБО-15-22 *Оганнисян Г.А.*

Принял старший преподаватель *Боронников А.С.*

Практическая работа выполнена «__»_____2023 г.

«Зачтено» «__»_____2023 г.

Москва 2021

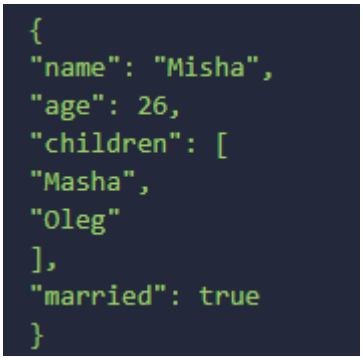

Рисунок 1 – Ввод данных в программу на Flex

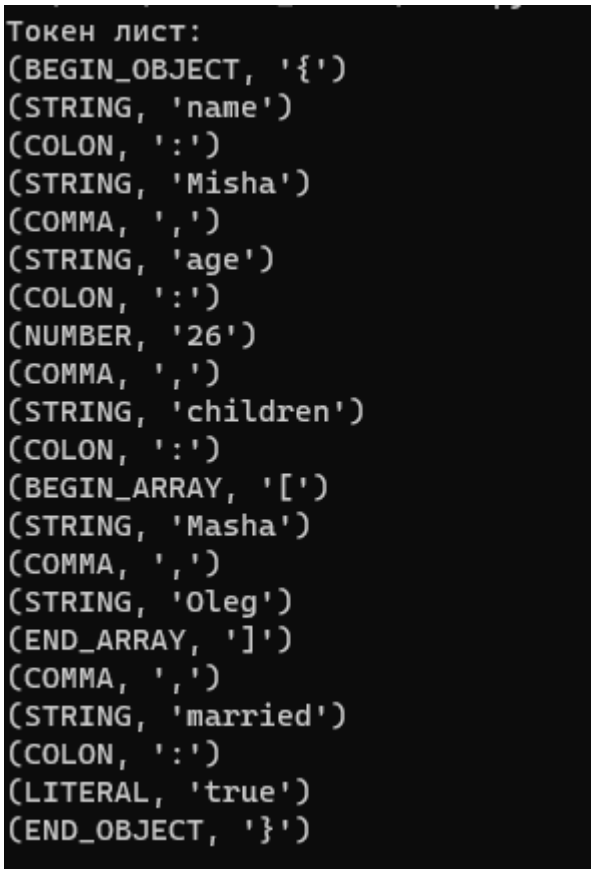

Рисунок 2 – Вывод программы

Этот код представляет собой простой лексический анализатор, написанный на языке Flex. Его основная цель - разбор входного потока символов и выделение лексем (токенов) в соответствии с определенными правилами. Каждый токен представляет собой часть входных данных, такую как строка, число, символы массива, и т. д.

Программа начинает с описания различных токенов и их шаблонов, используя регулярные выражения. Например, **{BEGIN_OBJECT}** соответствует открывающей фигурной скобке **{**, и так далее. Когда лексический анализатор обнаруживает соответствие шаблону, он выполняет соответствующее

действие, такое как вывод сообщения о найденном токене.

Затем идет основная часть программы, в которой задаются правила для обработки каждого типа токена. Например, при обнаружении строки в кавычках, программа удаляет кавычки и выводит токен типа **STRING** без них.

Наконец, в функции **main** вызывается **yylex()**, что инициирует процесс лексического анализа входных данных. Результатом работы программы является вывод токенов с указанием их типа и значения. Если встречается неизвестный символ, программа выводит сообщение об этом.

В целом, этот код представляет собой пример простого лексического анализатора, способного обрабатывать базовые элементы JSON-подобного синтаксиса.

Листинг кода

```
from ply import lex

tokens = (
    'BEGIN_OBJECT',
    'END_OBJECT',
    'BEGIN_ARRAY',
    'END_ARRAY',
    'COMMA',
    'COLON',
    'LITERAL',
    'STRING',
    'NUMBER',
)

t_BEGIN_OBJECT = r'\{'
t_END_OBJECT = r'\}'
t_BEGIN_ARRAY = r'\['
t_END_ARRAY = r'\]'
t_COMMA = r','
t_COLON = r':'
t_LITERAL = r'true|false|null'


# Регулярное выражение для строк (захватывает символы в двойных кавычках)
def t_STRING(t):
    r'"([^"\\]|\\.)*"'
    t.value = t.value[1:-1]  # Убираем двойные кавычки
    return t
```

```
# Регулярное выражение для чисел (целые и с плавающей запятой), ? - 0/1, + - 1/inf, * - 0/inf
def t_NUMBER(t):
    r'[+-]?[0-9]+(\.[0-9]+([eE][+-]?[0-9]+)?)?'
    t.value = float(t.value) if '.' in t.value or 'e' in t.value or 'E' in t.value else int(t.value)
    return t


# Пропуск пробелов и переводов строк
t_ignore = ' \t\n'


# Обработка ошибок
def t_error(t):
    print(f"Ошибка: {t.value[0]}")
    t.lexer.skip(1)


# Создание лексического анализатора
lexer = lex.lex()

if __name__ == "__main__":
    data = '''
    {
    "name": "Misha",
    "age": 26,
    "children": [
    "Masha",
    "Oleg"
    ],
    "married": true
    }
    '''

    lexer.input(data)

    tokenlist = []

    while True:
        token = lexer.token()
        if not token:
            break
        tokenlist.append((token.type, token.value))

    print("Токен лист:")
    for token in tokenlist:
        print(f"({token[0]}, '{token[1]}')")
```