

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Космических и информационных технологий

институт

Кафедра «Информатика»

кафедра

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

Преподаватель

Студент

КИ19–04–1М, 031943354

номер группы, зачетной книжки

подпись, дата

подпись, дата

А.С. Кузнецов

инициалы, фамилия

Т.В. Радионов

инициалы, фамилия

Красноярск 2020

1 Цель и задачи

Цель: изучение основных методов организации таблиц идентификаторов в компиляторах языков программирования и методов синтаксического и контекстного анализа с их программной реализацией.

Задачи:

- изучение теоретического материала об основных методах организации таблиц имен;
- изучение теоретического материала по организации синтаксического и контекстного анализа языков программирования;
- составление формального описания процессов синтаксического и контекстного анализа;
- программная реализация по формальному описанию.

2 Классификация ошибок

Лексические – `LexicalException`. Ошибки написания языковых конструкций.

Синтаксические – `SyntaxException`. Ошибки последовательности языковых конструкций.

Семантические – `SemanticException`. Логические ошибки языковых конструкций.

3 Описание семантики основных конструкций

Функции: являются именованным контейнером, в которых объявляются операторы или не объявляется ничего, также может вызываться внутренняя функция. Функция объявляется с ключевого слова “`def`”, далее задается название функции (именование соответствует правилам именования переменных), затем в скобках перечисляются аргументы функции, далее в фигурных скобках описываются операторы. Приоритетов нет.

Операторы: логические конструкции, используемые для действий с выражениями и также вызывающие функции. Есть несколько типов: условные (`if...else`), циклические (`do...while`), возвращаемые (`return`), выводющие (`print`), присваивающие (`=`), вызывающие (`function()`). Приоритетов нет, используются только внутри функций.

Выражения: представляют из себя арифметические и логические операции, а также операции над строками. Приоритеты согласно общепринятым приоритетам операций в математике.

4 Формальное описание синтаксического и контекстного анализатора

Описание представлено на таблице 1

Таблица 1 – Формальное описание парсера

Порождающее	Порождаемое
F	def VAR (VAR*) { S* }
S	VAR=E VAR(VAR*) print (E) if (E) { S+ } if (E) { S+ } else { S+ } while (E) { S+ } do { S+ } while (E) return E
E	E E E&&E (E) INT BOOL STRING VAR
VAR	[a-zA-Z][0-9]*
INT	[0-9]+
BOOL	true false
STRING	“[^“]*”

5 Тестовые примеры

Тест 0

```
def test00(x y z)
```

```
{
```

```
    // test function
```

```
    x = x + 1
```

```
    y = 1
```

```
    return y
```

```
}
```

```
def test0()
```

```
{
```

```
    x = 2 * 2
```

```
    z = 3
```

```
    test00(10 9 8)
```

```
    return 5 + x
```

```
}
```

```
F0 -> def test00 ( E0 E1 E2 ) { S0 S1 S2 }
```

```
    E0 -> null
```

```
    E1 -> null
```

```
    E2 -> null
```

```
    S0 -> x=E4 = null
```

```
        E0 -> null
```

```
        E3 -> 1
```

```
        E4 -> E0+E3 = null+1 = null
```

```
    S1 -> y=E5 = 1
```

```
        E5 -> 1
```

```
    S2 -> return E5
```

```
        E5 -> 1
```

```
F1 -> def test0 ( ) { S3 S4 S8 S9 }
```

```
    S3 -> x=E8 = 4
```

```
        E6 -> 2
```

```
        E7 -> 2
```

```
        E8 -> E6*E7 = 2*2 = 4
```

```
    S4 -> z=E9 = 3
```

```
        E9 -> 3
```

```
    S8 -> F0 = test00 ( E14 E15 E12 ) { S5 S6 S7 }
```

```
        E14 -> E10+E13 = 10+1 = 11
```

```

E15 -> 1
E12 -> 8
S5 -> x=E14 = 11
E10 -> 10
E13 -> 1
E14 -> E10+E13 = 10+1 = 11
S6 -> y=E15 = 1
E15 -> 1
S7 -> return E15
E15 -> 1
S9 -> return E17
E16 -> 5
E6 -> 2
E7 -> 2
E8 -> E6*E7 = 2*2 = 4
E17 -> E16+E8 = 5+4 = 9

```

Тест 1

```

def test1()
{
    x = 2 + 3 * (4 - 5) - (2 + 3)
}

```

```

F0 -> def test1 ( ) { S0 }
S0 -> x=E12 = -6
E0 -> 2
E1 -> 3
E2 -> 4
E3 -> 5
E4 -> E2-E3 = 4-5 = -1
E5 -> (E4) = -1
E6 -> E1*E5 = 3*-1 = -3
E7 -> E0+E6 = 2+-3 = -1
E8 -> 2
E9 -> 3
E10 -> E8+E9 = 2+3 = 5
E11 -> (E10) = 5
E12 -> E7-E11 = -1-5 = -6

```

Тест 2

```

def test2()
{
    a = (2 > 3) && (3 < 1) || true || !false
}

```

```

F0 -> def test2 ( ) { S0 }
S0 -> a=E13 = true
E0 -> 2
E1 -> 3
E2 -> E0>E1 = 2>3 = false
E3 -> (E2) = false

```

```

E4 -> 3
E5 -> 1
E6 -> E4<E5 = 3<1 = false
E7 -> (E6) = false
E8 -> E3&&E7 = false&&>false = false
E9 -> true
E10 -> E8||E9 = false||true = true
E11 -> false
E12 -> !E11 = true
E13 -> E10||E12 = true||true = true

```

Тест 3

```
def test3()
```

```

{
    x = (2 + 2) * 2
    y = 7
    if (x == y)
    {
        if (2 * 4 > y)
        {
            print("y < 8")
            y = y + 1
        }
        else
        {
            x = -5
        }
    }
}

```

```
F0 -> def test3 ( ) { S0 S1 S6 }
```

```
S0 -> x=E5 = 8
```

```
E0 -> 2
```

```
E1 -> 2
```

```
E2 -> E0+E1 = 2+2 = 4
```

```
E3 -> (E2) = 4
```

```
E4 -> 2
```

```
E5 -> E3*E4 = 4*2 = 8
```

```
S1 -> y=E6 = 7
```

```
E6 -> 7
```

```
S6 -> if ( E7 ) { S5 }
```

```
E0 -> 2
```

```
E1 -> 2
```

```
E2 -> E0+E1 = 2+2 = 4
```

```
E3 -> (E2) = 4
```

```
E4 -> 2
```

```
E5 -> E3*E4 = 4*2 = 8
```

```
E6 -> 7
```

```
E7 -> E5==E6 = 8==7 = false
```

```
S2 -> print E12
```

```

E12 -> "y < 8"
S3 -> y=E14 = 8
E6 -> 7
E13 -> 1
E14 -> E6+E13 = 7+1 = 8
S4 -> x=E16 = -5
E15 -> 5
E16 -> -E15 = -5
S5 -> if ( E11 ) { S2 S3 } else { S4 }
E8 -> 2
E9 -> 4
E10 -> E8*E9 = 2*4 = 8
E6 -> 7
E11 -> E10>E6 = 8>7 = true

```

Тест 4

```

def test4()
{
    w = 1
    while (w < 3)
    {
        w = w + 1
    }
}
F0 -> def test4 ( ) { S0 S4 }
S0 -> w=E0 = 1
E0 -> 1
S4 -> while ( E2 ) { S1 S2 S3 }
E0 -> 1
E1 -> 3
E2 -> E0<E1 = 1<3 = true
S1 -> w=E4 = 2
E0 -> 1
E3 -> 1
E4 -> E0+E3 = 1+1 = 2
S2 -> w=E8 = 3
E0 -> 1
E3 -> 1
E4 -> E0+E3 = 1+1 = 2
E7 -> 1
E8 -> E4+E7 = 2+1 = 3
S3 -> w=E12 = 4
E0 -> 1
E3 -> 1
E4 -> E0+E3 = 1+1 = 2
E7 -> 1
E8 -> E4+E7 = 2+1 = 3
E11 -> 1
E12 -> E8+E11 = 3+1 = 4

```

Тест 5

```
def test5()
```

```
{
    y = 3
    do
    {
        y = y - 1
    } while (y > 1)
}
```

```
F0 -> def test5 ( ) { S0 S3 }
```

```
S0 -> y=E0 = 3
```

```
E0 -> 3
```

```
S3 -> do { S1 S2 } while ( E4 )
```

```
E0 -> 3
```

```
E1 -> 1
```

```
E2 -> E0-E1 = 3-1 = 2
```

```
E3 -> 1
```

```
E4 -> E2>E3 = 2>1 = true
```

```
S1 -> y=E2 = 2
```

```
E0 -> 3
```

```
E1 -> 1
```

```
E2 -> E0-E1 = 3-1 = 2
```

```
S2 -> y=E6 = 1
```

```
E0 -> 3
```

```
E1 -> 1
```

```
E2 -> E0-E1 = 3-1 = 2
```

```
E5 -> 1
```

```
E6 -> E2-E5 = 2-1 = 1
```

Тест 6

```
def test6() {
```

```
}
```

```
F0 -> def test6 ( ) { }
```

Тест 7

```
def test7() {
```

```
    test7()
```

```
}
```

app.classes.exceptions.SemanticException: Функция "test7" не
объявлена: строка 2, позиция 14

Тест 8

```
def test8(x=3)
```

```
{
```

```
    x = 1
```

```
}
```

app.classes.exceptions.SyntaxException: Ожидался "VAR" или ")":
строка 1, позиция 15

```
Тест 9
def test9()
{
    if()
    {
        x = 1
    }
}
```

app.classes.exceptions.SyntaxException: Неизвестное выражение ")":
строка 3, позиция 9

```
Тест 10
def test10()
{
    if(x > 1)
    {
        x = x + 1
    }
}
```

app.classes.exceptions.SemanticException: Переменная "x" не
инициализирована: строка 3, позиция 9

```
Тест 11
def test11()
{
    x = 5
    if (x - 3 = 2)
    {
        print("x = 2")
    }
}
```

app.classes.exceptions.SemanticException: Ожидался класс "Boolean"
вместо "class java.lang.Integer" результата выражения "3": строка 4,
позиция 14

```
Тест 12
def test12()
{
    x = true
    if (x)
    {
        x = false
    }
    else
        x = true
}
```


app.classes.exceptions.SyntaxException: Ожидалась "{" после "else":
строка 8, позиция 7

```
Тест 13
def test13()
{
    x = 5 / 0
}
```

app.classes.exceptions.SemanticException: / by zero: строка 3,
позиция 12

```
Тест 14
def test14()
{
    x = true
    x = x + 1
}
```

app.classes.exceptions.SemanticException: Не определен класс
результата выражения "true+1": строка 4, позиция 12

```
Тест 15
def test15()
{
    test150(10)
}
def test150(x)
{
    y = x
}
```

app.classes.exceptions.SemanticException: Функция "test150" не
объявлена: строка 3, позиция 14

```
Тест 16
def test160(x)
{
    print(x)
}
def test16()
{
    test160(10)
}
```

```
F0 -> def test160 ( E0 ) { S0 }
```

```
E0 -> null
```

```
S0 -> print E0
```

```
E0 -> null
```

```
F1 -> def test16 ( ) { S2 }
```

```
S2 -> F0 = test160 ( E1 ) { S1 }
```

```
E1 -> 10
```

```
S1 -> print E1
```

E1 -> 10

Тест 17
def test170(x)

```
{
    print(x)
}
```

```
def test17()
{
    test170(10 20)
}
```

app.classes.exceptions.SyntaxException: Ожидалась ")" после "10":
строка 7, позиция 19

Тест 18
def test18()

```
{
    return
}
```

app.classes.exceptions.SyntaxException: Неизвестное выражение "}:
строка 4, позиция 2

Тест 19
def test19()

```
{
    return x
}
```

app.classes.exceptions.SemanticException: Переменная "x" не
инициализирована: строка 3, позиция 11

Тест 20
def test20()

```
{
    while (5 + 1)
    {
        x = 1
    }
}
```

app.classes.exceptions.SemanticException: Ожидался класс "Boolean"
вместо "class java.lang.Integer" результата выражения "1": строка 3,
позиция 15