Министерство науки и высшего образования Российской Федерации

Санкт-Петербургский Политехнический Университет Петра Великого

—

Институт прикладной математики и механики

**Высшая школа кибербезопасности и защиты информации**

**Курсовая работа**

«**Реализация грамматики языка программирования Java**»

по дисциплине «Формальные грамматики и теория компиляторов»

Выполнили

студенты гр. 3651001/70201 К. О. Лукин

Н. В. Соболев

Проверил

Преподаватель А. В. Мясников

Санкт-Петербург

2020

**Цель работы**

Реализовать грамматику языка программирования Java с помощью средств bison и flex.

**Формулировка задания**

Необходимо разработать для утилиты flex код, который содержит в себе ключевые слова, строки, числа, идентификаторы, языка Java, которые в последствии разбиваются на лексемы. Для утилиты bison разработать код, который описывает конструкции языка Java.

**Ход работы**

В ходе работы были рассмотрены основные конструкции языка Java. Для облегчения задачи были исключены некоторые конструкции языка:

* Анонимные классы;

На официальном сайте Oracle был найден список ключевых слов языка Java, который был описан в файле для утилиты flex (рисунок 1).

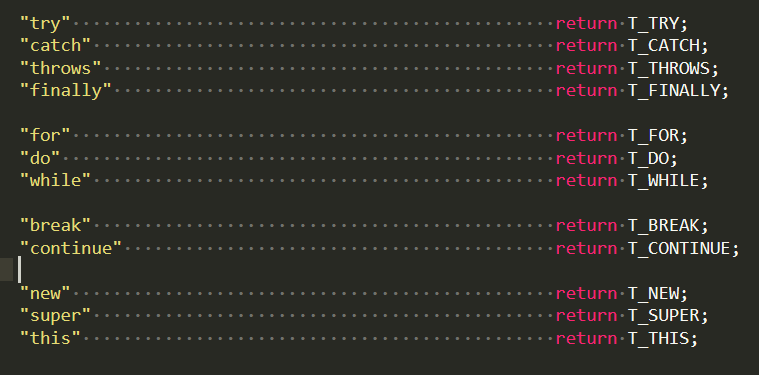


Рисунок 1 - Листинг файла для flex.

Так как язык программирования Java является объектно-ориентированным языком, то у него присутствует наследование и инкапсуляция. Классы в Java могут наследоваться от одного класса и реализовывать множество интерфейсов, для этого были описаны правила “R\_extends” и “R\_implements”, которые содержат перечисление, через запятую, интерфейсов. Также у классов, методов и полей присутствуют различные модификаторы (например, модификаторы доступа, модификаторы static и final и т.д.).

В грамматике было реализовано объявление классов, методов и полей с модификаторами в различном порядке. Для того, чтобы множество first не пересекалось, были описаны правила для разных случаев.

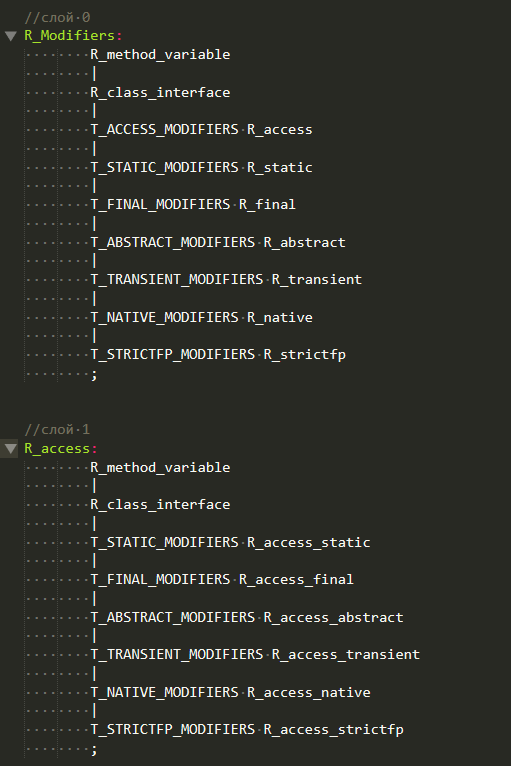


Рисунок 2 - Описание модификаторов.

Далее были описаны конструкции импортирования библиотек, и объединения в пакет. Данные конструкции были объединены в одно правило “R\_Directives”, они могут использоваться только в начале Java-кода.

Затем было описано тело класса, которое может содержать в себе другие классы, поля и методы. Для полей был описан тип данных, инициализация переменной при объявлении, и объявление через запятую нескольких переменных одного типа данных.

Затем было описано рекурсивное тело метода, которое может содержать локальные переменные и внутренние классы.

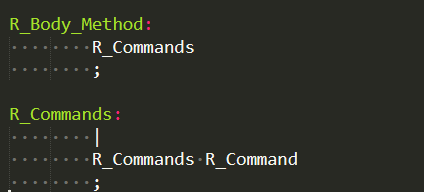


Рисунок 3 - Правила для описания тела метода.

Основное правило “R\_Command” содержит описание базовых конструкций языка Java: циклы, условия, выражения, исключения.

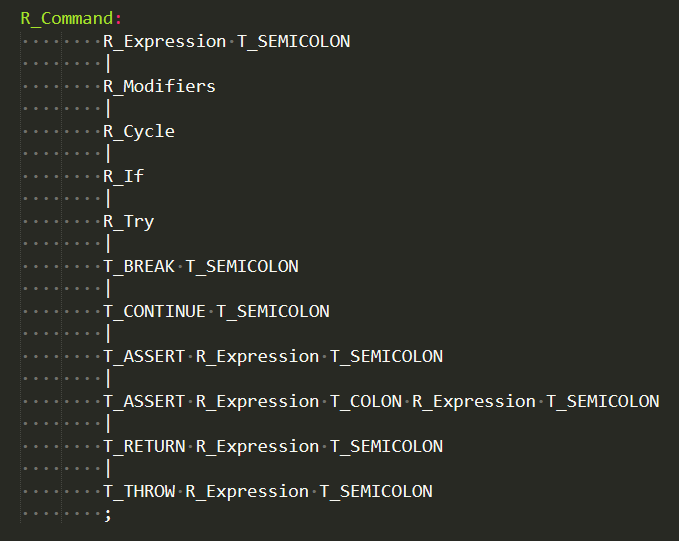


Рисунок 4 - Правило “R\_command”.

Последним рубежом в описании грамматики стало описание выражений. Для этого было создано правило “R\_Expression”, в котором множество first не пересекается, то есть по первому токену выбирается нужное правило, тем самым уменьшено количество ошибок “shift\reduce”. Выражения описывают унарные символы, вызов функций, обращения к элементам массивов, обращения к полям классов.

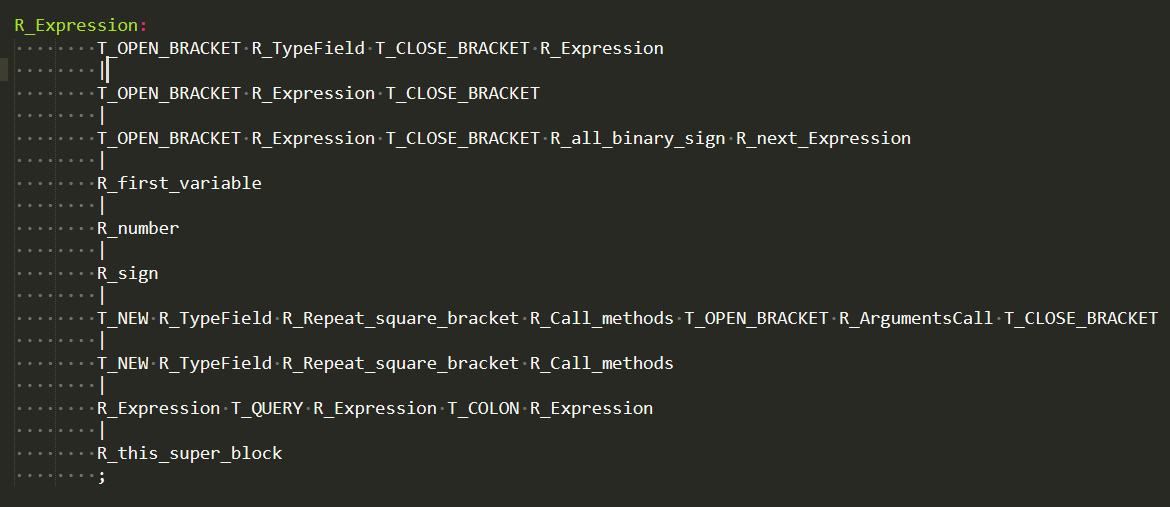


Рисунок 5 - Правило для описания выражений.

После разработки грамматики был создан файл test.z, с помощью которого было проведено тестирование. Также был написан makefile, которых содержит 3 цели:

* all - стандартная компиляция;
* debug - компиляция с флагом для отладки;
* run - запуск полученного транслятора с файлом test.z.

**Вывод**

В результате выполнения данной курсовой работы были рассмотрены базовые конструкции языка Java, получен навык описания грамматики для языка высокого уровня.

**Приложение А**

Исходный код для flex:

%{

#include <stdio.h>

#include "y.tab.h"

extern int number\_str;

%}

%%

"import" return T\_IMPORT;

"package" return T\_PACKAGE;

"class" return T\_CLASS;

"interface" return T\_INTERFACE;

"byte"|"boolean"|"char"|"double"|"float"|"int"|"long"|"short"|"String" return T\_DATA\_TYPE;

"var" return T\_DATA\_TYPE\_VARIABLE;

"void" return T\_DATA\_TYPE\_METHOD;

"enum" return T\_ENUM;

"public"|"private"|"protected" return T\_ACCESS\_MODIFIERS;

"static" return T\_STATIC\_MODIFIERS;

"final" return T\_FINAL\_MODIFIERS;

"abstract" return T\_ABSTRACT\_MODIFIERS;

"transient" return T\_TRANSIENT\_MODIFIERS;

"native" return T\_NATIVE\_MODIFIERS;

"strictfp" return T\_STRICTFP\_MODIFIERS;

"synchronized"|"volatile" return T\_STREAM\_MODIFIERS;

"extends" return T\_EXTENDS;

"implements" return T\_IMPLEMENTS;

"throw" return T\_THROW;

"true"|"false" return T\_TRUE\_FALSE;

"if" return T\_IF;

"else" return T\_ELSE;

"switch" return T\_SWITCH;

"case" return T\_CASE;

"default" return T\_DEFAULT;

"try" return T\_TRY;

"catch" return T\_CATCH;

"throws" return T\_THROWS;

"finally" return T\_FINALLY;

"for" return T\_FOR;

"do" return T\_DO;

"while" return T\_WHILE;

"break" return T\_BREAK;

"continue" return T\_CONTINUE;

"new" return T\_NEW;

"super" return T\_SUPER;

"this" return T\_THIS;

"assert" return T\_ASSERT;

"instaceof" return T\_INSTACEOF;

"return" return T\_RETURN;

"++"|"--" return T\_INCREMENT\_DECREMENT\_SIGN;

"~"|"!" return T\_UNARY\_SIGN\_ONLY;

"-" return T\_MINUS;

"+" return T\_PLUS;

"=="|"!="|">="|"<="|">"|"<" return T\_COMPARISON\_SIGN;

"/"|"%"|"&"|"|"|"^"|"<<"|">>"|">>>"|"&&"|"||" return T\_BINARY\_SIGN;

"\*" return T\_STAR;

"=" return T\_ASSIGNMENT;

"+="|"-="|"\*="|"/="|"%="|"&="|"|="|">>="|"<<=" return T\_ASSIGNMENT2;

"(" return T\_OPEN\_BRACKET;

")" return T\_CLOSE\_BRACKET;

"[" return T\_OPEN\_SQUARE\_BRACKET;

"]" return T\_CLOSE\_SQUARE\_BRACKET;

"{" return T\_OPEN\_BRACE;

"}" return T\_CLOSE\_BRACE;

";" return T\_SEMICOLON;

"." return T\_DOT;

"," return T\_COMMA;

"?" return T\_QUERY;

":" return T\_COLON;

\"([^"]|\\.)\*\" return T\_STRING;

[0-9]+"."[0-9]+"f" return T\_FLOAT;

[0-9]+"."[0-9]+ return T\_FLOAT;

[0-9]+ return T\_NUMBER;

[\_a-zA-Z][\_a-zA-Z0-9]\* return T\_IDENTIFIER;

\n ++number\_str;

[ \t]+ ;

%%

**Приложение Б**

Исходный код для bison:

%{

#include <stdio.h>

#include <string.h>

extern int yylex();

extern int yylval;

int yydebug=1;

int number\_str=1;

void yyerror(const char \*str){

fprintf(stderr,"error: %s\n", str);

fprintf(stderr,"str: %d\n", number\_str);

}

int yywrap(){

return 1;

}

%}

%start R\_Programm

%token T\_STAR T\_IMPORT T\_PACKAGE T\_CLASS T\_INTERFACE T\_DATA\_TYPE T\_DATA\_TYPE\_VARIABLE T\_DATA\_TYPE\_METHOD T\_ENUM T\_ACCESS\_MODIFIERS T\_STATIC\_MODIFIERS T\_FINAL\_MODIFIERS T\_ABSTRACT\_MODIFIERS T\_TRANSIENT\_MODIFIERS T\_NATIVE\_MODIFIERS T\_STRICTFP\_MODIFIERS T\_STREAM\_MODIFIERS T\_EXTENDS T\_IMPLEMENTS T\_THROW T\_TRUE\_FALSE T\_IF T\_ELSE T\_SWITCH T\_CASE T\_DEFAULT T\_TRY T\_CATCH T\_THROWS T\_FINALLY T\_FOR T\_DO T\_WHILE T\_BREAK T\_CONTINUE T\_NEW T\_SUPER T\_THIS T\_ASSERT T\_INSTACEOF T\_RETURN T\_INCREMENT\_DECREMENT\_SIGN T\_UNARY\_SIGN\_ONLY T\_COMPARISON\_SIGN T\_BINARY\_SIGN T\_ASSIGNMENT T\_OPEN\_BRACKET T\_CLOSE\_BRACKET T\_OPEN\_SQUARE\_BRACKET T\_CLOSE\_SQUARE\_BRACKET T\_OPEN\_BRACE T\_CLOSE\_BRACE T\_SEMICOLON T\_DOT T\_COMMA T\_QUERY T\_COLON T\_FLOAT T\_STRING T\_NUMBER T\_IDENTIFIER T\_MINUS T\_PLUS T\_ASSIGNMENT2

%%

R\_Programm:

R\_Directives R\_Classes

;

R\_Classes:

|

R\_Classes R\_Modifiers

;

R\_Directives:

|

R\_Directives R\_Package T\_SEMICOLON

|

R\_Directives R\_Import T\_SEMICOLON

;

R\_Package:

T\_PACKAGE T\_IDENTIFIER

;

R\_NamePacket:

|

R\_NamePacket T\_DOT T\_IDENTIFIER

;

R\_Import:

T\_IMPORT T\_IDENTIFIER R\_NamePacket T\_DOT T\_STAR

|

T\_IMPORT T\_IDENTIFIER R\_NamePacket

;

R\_EXTENDS://наследование

|

T\_EXTENDS T\_IDENTIFIER

;

R\_IMPLEMENTS://имплементация

|

T\_IMPLEMENTS T\_IDENTIFIER R\_comma\_next\_IMPLEMENTS

;

R\_comma\_next\_IMPLEMENTS://множественная имплементация

|

T\_COMMA T\_IDENTIFIER R\_comma\_next\_IMPLEMENTS

;

R\_class\_interface:

T\_CLASS T\_IDENTIFIER R\_EXTENDS R\_IMPLEMENTS R\_BodyClass

|

T\_INTERFACE T\_IDENTIFIER R\_BodyClass

;

//слой 0

R\_Modifiers:

R\_method\_variable

|

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access

|

T\_STATIC\_MODIFIERS R\_static

|

T\_FINAL\_MODIFIERS R\_final

|

T\_ABSTRACT\_MODIFIERS R\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_transient

|

T\_NATIVE\_MODIFIERS R\_native

|

T\_STRICTFP\_MODIFIERS R\_strictfp

;

//слой 1

R\_access:

R\_method\_variable

|

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_access\_static

|

T\_FINAL\_MODIFIERS R\_access\_final

|

T\_ABSTRACT\_MODIFIERS R\_access\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_access\_transient

|

T\_NATIVE\_MODIFIERS R\_access\_native

|

T\_STRICTFP\_MODIFIERS R\_access\_strictfp

;

R\_static:

R\_method\_variable

|

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_static

|

T\_FINAL\_MODIFIERS R\_static\_final

|

T\_ABSTRACT\_MODIFIERS R\_static\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_static\_transient

|

T\_NATIVE\_MODIFIERS R\_static\_native

|

T\_STRICTFP\_MODIFIERS R\_static\_strictfp

;

R\_final:

R\_method\_variable

|

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_final

|

T\_STATIC\_MODIFIERS R\_static\_final

|

T\_ABSTRACT\_MODIFIERS R\_final\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_final\_transient

|

T\_NATIVE\_MODIFIERS R\_final\_native

|

T\_STRICTFP\_MODIFIERS R\_final\_strictfp

;

R\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_abstract

|

T\_STATIC\_MODIFIERS R\_static\_abstract

|

T\_FINAL\_MODIFIERS R\_final\_abstract

|

T\_NATIVE\_MODIFIERS R\_abstract\_native

|

T\_STRICTFP\_MODIFIERS R\_abstract\_strictfp

;

R\_transient:

R\_defin\_var1

|

T\_ACCESS\_MODIFIERS R\_access\_transient

|

T\_STATIC\_MODIFIERS R\_static\_transient

|

T\_FINAL\_MODIFIERS R\_final\_transient

;

R\_native:

R\_method\_abstract//абстраткт так как нативный метод не имеет тела

|

T\_ACCESS\_MODIFIERS R\_access\_native

|

T\_STATIC\_MODIFIERS R\_static\_native

|

T\_FINAL\_MODIFIERS R\_final\_native

;

R\_strictfp:

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_strictfp

|

T\_STATIC\_MODIFIERS R\_static\_strictfp

|

T\_FINAL\_MODIFIERS R\_final\_strictfp

|

T\_ABSTRACT\_MODIFIERS R\_abstract\_strictfp

;

//слой 2

R\_access\_static:

R\_method\_variable

|

R\_class\_interface

|

T\_FINAL\_MODIFIERS R\_access\_static\_final

|

T\_ABSTRACT\_MODIFIERS R\_access\_static\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_access\_static\_transient

|

T\_NATIVE\_MODIFIERS R\_access\_static\_native

|

T\_STRICTFP\_MODIFIERS R\_access\_static\_strictfp

;

R\_access\_final:

R\_method\_variable

|

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_access\_static\_final

|

T\_ABSTRACT\_MODIFIERS R\_access\_final\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_access\_final\_transient

|

T\_NATIVE\_MODIFIERS R\_access\_final\_native

|

T\_STRICTFP\_MODIFIERS R\_access\_final\_strictfp

;

R\_access\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_access\_static\_abstract

|

T\_FINAL\_MODIFIERS R\_access\_final\_abstract

|

T\_NATIVE\_MODIFIERS R\_access\_abstract\_native

|

T\_STRICTFP\_MODIFIERS R\_access\_abstract\_strictfp

;

R\_access\_transient:

R\_defin\_var1

|

T\_STATIC\_MODIFIERS R\_defin\_var1

|

T\_FINAL\_MODIFIERS R\_defin\_var1

|

T\_STATIC\_MODIFIERS T\_FINAL\_MODIFIERS R\_defin\_var1

|

T\_FINAL\_MODIFIERS T\_STATIC\_MODIFIERS R\_defin\_var1

;

R\_access\_native:

R\_method\_abstract//абстраткт так как нативный метод не имеет тела

|

T\_STATIC\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_STATIC\_MODIFIERS T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS T\_STATIC\_MODIFIERS R\_method\_abstract

;

R\_access\_strictfp:

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_access\_static\_strictfp

|

T\_FINAL\_MODIFIERS R\_access\_final\_strictfp

|

T\_ABSTRACT\_MODIFIERS R\_access\_abstract\_strictfp

;

R\_static\_final:

R\_method\_variable

|

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_static\_final

|

T\_ABSTRACT\_MODIFIERS R\_static\_final\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_static\_final\_transient

|

T\_NATIVE\_MODIFIERS R\_static\_final\_native

|

T\_STRICTFP\_MODIFIERS R\_static\_final\_strictfp

;

R\_static\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_static\_abstract

|

T\_FINAL\_MODIFIERS R\_static\_final\_abstract

|

T\_NATIVE\_MODIFIERS R\_static\_abstract\_native

|

T\_STRICTFP\_MODIFIERS R\_static\_abstract\_strictfp

;

R\_static\_transient:

R\_defin\_var1

|

T\_FINAL\_MODIFIERS R\_defin\_var1

|

T\_ACCESS\_MODIFIERS R\_defin\_var1

|

T\_FINAL\_MODIFIERS T\_ACCESS\_MODIFIERS R\_defin\_var1

|

T\_ACCESS\_MODIFIERS T\_FINAL\_MODIFIERS R\_defin\_var1

;

R\_static\_native:

R\_method\_abstract//абстраткт так как нативный метод не имеет тела

|

T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS T\_FINAL\_MODIFIERS R\_method\_abstract

;

R\_static\_strictfp:

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_static\_strictfp

|

T\_FINAL\_MODIFIERS R\_static\_final\_strictfp

|

T\_ABSTRACT\_MODIFIERS R\_static\_abstract\_strictfp

;

R\_final\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_static\_final\_abstract

|

T\_ACCESS\_MODIFIERS R\_access\_final\_abstract

|

T\_NATIVE\_MODIFIERS R\_final\_abstract\_native

|

T\_STRICTFP\_MODIFIERS R\_final\_abstract\_strictfp

;

R\_final\_transient:

R\_defin\_var1

|

T\_STATIC\_MODIFIERS R\_defin\_var1

|

T\_ACCESS\_MODIFIERS R\_defin\_var1

|

T\_STATIC\_MODIFIERS T\_ACCESS\_MODIFIERS R\_defin\_var1

|

T\_ACCESS\_MODIFIERS T\_STATIC\_MODIFIERS R\_defin\_var1

;

R\_final\_native:

R\_method\_abstract//абстраткт так как нативный метод не имеет тела

|

T\_STATIC\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_STATIC\_MODIFIERS T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS T\_STATIC\_MODIFIERS R\_method\_abstract

;

R\_final\_strictfp:

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_final\_strictfp

|

T\_STATIC\_MODIFIERS R\_static\_final\_strictfp

|

T\_ABSTRACT\_MODIFIERS R\_final\_abstract\_strictfp

;

R\_abstract\_native:

R\_method\_abstract

|

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_static\_abstract\_native

|

T\_FINAL\_MODIFIERS R\_final\_abstract\_native

|

T\_ACCESS\_MODIFIERS R\_access\_abstract\_native

;

R\_abstract\_strictfp:

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_static\_abstract\_strictfp

|

T\_FINAL\_MODIFIERS R\_final\_abstract\_strictfp

|

T\_ACCESS\_MODIFIERS R\_access\_abstract\_strictfp

;

//слой 3

R\_access\_static\_final:

R\_method\_variable

|

R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS R\_access\_static\_final\_abstract

|

T\_TRANSIENT\_MODIFIERS R\_defin\_var1

|

T\_NATIVE\_MODIFIERS R\_access\_static\_final\_native

|

T\_STRICTFP\_MODIFIERS R\_access\_static\_final\_strictfp

;

R\_access\_static\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_FINAL\_MODIFIERS R\_access\_static\_final\_abstract

|

T\_NATIVE\_MODIFIERS R\_access\_static\_abstract\_native

|

T\_STRICTFP\_MODIFIERS R\_access\_static\_abstract\_strictfp

;

R\_access\_static\_transient:

R\_defin\_var1

|

T\_FINAL\_MODIFIERS R\_defin\_var1

;

R\_access\_static\_native:

R\_method\_abstract

|

T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_ABSTRACT\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS T\_ABSTRACT\_MODIFIERS R\_method\_abstract

|

T\_ABSTRACT\_MODIFIERS T\_FINAL\_MODIFIERS R\_method\_abstract

;

R\_access\_static\_strictfp:

R\_class\_interface

|

T\_FINAL\_MODIFIERS R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS R\_class\_interface

|

T\_FINAL\_MODIFIERS T\_ABSTRACT\_MODIFIERS R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS T\_FINAL\_MODIFIERS R\_class\_interface

;

R\_access\_final\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_access\_static\_final\_abstract

|

T\_NATIVE\_MODIFIERS R\_access\_final\_abstract\_native

|

T\_STRICTFP\_MODIFIERS R\_access\_final\_abstract\_strictfp

;

R\_access\_final\_transient:

R\_defin\_var1

|

T\_STATIC\_MODIFIERS R\_defin\_var1

;

R\_access\_final\_native:

R\_method\_abstract

|

T\_STATIC\_MODIFIERS R\_method\_abstract

|

T\_ABSTRACT\_MODIFIERS R\_method\_abstract

|

T\_STATIC\_MODIFIERS T\_ABSTRACT\_MODIFIERS R\_method\_abstract

|

T\_ABSTRACT\_MODIFIERS T\_STATIC\_MODIFIERS R\_method\_abstract

;

R\_access\_final\_strictfp:

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS R\_class\_interface

|

T\_STATIC\_MODIFIERS T\_ABSTRACT\_MODIFIERS R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS T\_STATIC\_MODIFIERS R\_class\_interface

;

R\_access\_abstract\_native:

R\_method\_abstract

|

T\_STATIC\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_STATIC\_MODIFIERS T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS T\_STATIC\_MODIFIERS R\_method\_abstract

;

R\_access\_abstract\_strictfp:

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_class\_interface

|

T\_FINAL\_MODIFIERS R\_class\_interface

|

T\_STATIC\_MODIFIERS T\_FINAL\_MODIFIERS R\_class\_interface

|

T\_FINAL\_MODIFIERS T\_STATIC\_MODIFIERS R\_class\_interface

;

R\_static\_final\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_access\_static\_final\_abstract

|

T\_NATIVE\_MODIFIERS R\_static\_final\_abstract\_native

|

T\_STRICTFP\_MODIFIERS R\_static\_final\_abstract\_strictfp

;

R\_static\_final\_transient:

R\_defin\_var1

|

T\_ACCESS\_MODIFIERS R\_defin\_var1

;

R\_static\_final\_native:

R\_method\_abstract

|

T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_ABSTRACT\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS T\_ABSTRACT\_MODIFIERS R\_method\_abstract

|

T\_ABSTRACT\_MODIFIERS T\_ACCESS\_MODIFIERS R\_method\_abstract

;

R\_static\_final\_strictfp:

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS R\_class\_interface

|

T\_ACCESS\_MODIFIERS T\_ABSTRACT\_MODIFIERS R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS T\_ACCESS\_MODIFIERS R\_class\_interface

;

R\_static\_abstract\_native:

R\_method\_abstract

|

T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS T\_FINAL\_MODIFIERS R\_method\_abstract

|

T\_FINAL\_MODIFIERS T\_ACCESS\_MODIFIERS R\_method\_abstract

;

R\_static\_abstract\_strictfp:

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_class\_interface

|

T\_FINAL\_MODIFIERS R\_class\_interface

|

T\_ACCESS\_MODIFIERS T\_FINAL\_MODIFIERS R\_class\_interface

|

T\_FINAL\_MODIFIERS T\_ACCESS\_MODIFIERS R\_class\_interface

;

R\_final\_abstract\_native:

R\_method\_abstract

|

T\_STATIC\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_STATIC\_MODIFIERS T\_ACCESS\_MODIFIERS R\_method\_abstract

|

T\_ACCESS\_MODIFIERS T\_STATIC\_MODIFIERS R\_method\_abstract

;

R\_final\_abstract\_strictfp:

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_class\_interface

|

T\_STATIC\_MODIFIERS T\_ACCESS\_MODIFIERS R\_class\_interface

|

T\_ACCESS\_MODIFIERS T\_STATIC\_MODIFIERS R\_class\_interface

;

//слой4

R\_access\_static\_final\_abstract:

R\_method\_abstract

|

R\_class\_interface

|

T\_NATIVE\_MODIFIERS R\_method\_abstract

|

T\_STRICTFP\_MODIFIERS R\_class\_interface

;

R\_access\_static\_final\_native:

R\_method\_abstract

|

T\_ABSTRACT\_MODIFIERS R\_method\_abstract

;

R\_access\_static\_final\_strictfp:

R\_class\_interface

|

T\_ABSTRACT\_MODIFIERS R\_class\_interface

;

R\_access\_static\_abstract\_native:

R\_method\_abstract

|

T\_FINAL\_MODIFIERS R\_method\_abstract

;

R\_access\_static\_abstract\_strictfp:

R\_class\_interface

|

T\_FINAL\_MODIFIERS R\_class\_interface

;

R\_access\_final\_abstract\_native:

R\_method\_abstract

|

T\_STATIC\_MODIFIERS R\_method\_abstract

;

R\_access\_final\_abstract\_strictfp:

R\_class\_interface

|

T\_STATIC\_MODIFIERS R\_class\_interface

;

R\_static\_final\_abstract\_native:

R\_method\_abstract

|

T\_ACCESS\_MODIFIERS R\_method\_abstract

;

R\_static\_final\_abstract\_strictfp:

R\_class\_interface

|

T\_ACCESS\_MODIFIERS R\_class\_interface

;

R\_Repeat\_square\_bracket:

|

T\_OPEN\_SQUARE\_BRACKET R\_Expression T\_CLOSE\_SQUARE\_BRACKET R\_Repeat\_square\_bracket

|

T\_OPEN\_SQUARE\_BRACKET T\_CLOSE\_SQUARE\_BRACKET R\_Repeat\_square\_bracket

;

R\_method\_abstract://разные типы

T\_IDENTIFIER R\_Repeat\_square\_bracket T\_IDENTIFIER R\_defin\_method\_abstract

|

T\_DATA\_TYPE R\_Repeat\_square\_bracket T\_IDENTIFIER R\_defin\_method\_abstract

|

T\_DATA\_TYPE\_METHOD T\_IDENTIFIER R\_defin\_method\_abstract

;

R\_defin\_var1:

T\_IDENTIFIER R\_Repeat\_square\_bracket T\_IDENTIFIER R\_defin\_var//всё ещё не понятно переменная или метод

|

T\_DATA\_TYPE R\_Repeat\_square\_bracket T\_IDENTIFIER R\_defin\_var//всё ещё не понятно переменная или метод

|

T\_DATA\_TYPE\_VARIABLE R\_Repeat\_square\_bracket T\_IDENTIFIER R\_defin\_var//точно переменная

;

R\_method\_variable://пеменная или метод

T\_IDENTIFIER R\_Repeat\_square\_bracket T\_IDENTIFIER R\_method\_or\_var//всё ещё не понятно переменная или метод

|

T\_DATA\_TYPE R\_Repeat\_square\_bracket T\_IDENTIFIER R\_method\_or\_var//всё ещё не понятно переменная или метод

|

T\_DATA\_TYPE\_VARIABLE R\_Repeat\_square\_bracket T\_IDENTIFIER R\_defin\_var//точно переменная

|

T\_DATA\_TYPE\_METHOD T\_IDENTIFIER R\_defin\_method//точно метод

|

T\_IDENTIFIER R\_defin\_method

;

R\_method\_or\_var:

R\_defin\_method

|

R\_defin\_var//точно переменная

;

R\_defin\_method\_abstract:

T\_OPEN\_BRACKET R\_Arguments T\_CLOSE\_BRACKET T\_SEMICOLON

;

R\_defin\_method:

T\_OPEN\_BRACKET R\_Arguments T\_CLOSE\_BRACKET T\_OPEN\_BRACE R\_Body\_Method T\_CLOSE\_BRACE//точно метод

|

T\_OPEN\_BRACKET R\_Arguments T\_CLOSE\_BRACKET T\_SEMICOLON

|

T\_OPEN\_BRACKET R\_Arguments T\_CLOSE\_BRACKET T\_THROWS R\_ArgumentsCall T\_OPEN\_BRACE R\_Body\_Method T\_CLOSE\_BRACE

;

R\_defin\_var:

R\_comma\_next\_var2 R\_comma\_next\_var T\_SEMICOLON//возможно несолько переменных одного типа

|

R\_comma\_next\_var2 R\_comma\_next\_var

;

R\_comma\_next\_var://несолько переменных одного типа через запятую

|

T\_COMMA R\_TypeField T\_IDENTIFIER R\_comma\_next\_var2 R\_comma\_next\_var

;

R\_comma\_next\_var2://возможно присвоение

|

T\_ASSIGNMENT R\_Expression

|

T\_ASSIGNMENT T\_OPEN\_BRACE R\_ArgumentsCall T\_CLOSE\_BRACE

;

R\_unary\_sign:

T\_MINUS

|

T\_PLUS

|

T\_UNARY\_SIGN\_ONLY

;

R\_Strings:

T\_STRING R\_Repeat\_square\_bracket

;

R\_Value:

T\_NUMBER

|

R\_Strings

|

T\_FLOAT

|

T\_TRUE\_FALSE

;

R\_TypeField:

T\_DATA\_TYPE

|

T\_DATA\_TYPE\_VARIABLE

|

T\_IDENTIFIER

;

R\_Arguments:

|

R\_Arguments T\_COMMA R\_TypeField R\_Repeat\_square\_bracket T\_IDENTIFIER

|

R\_TypeField R\_Repeat\_square\_bracket T\_IDENTIFIER

;

R\_Body\_Method:

R\_Commands

;

R\_Commands:

|

R\_Commands R\_Command

;

R\_ArgumentsCall:

|

R\_ArgumentsCall T\_COMMA R\_Expression

|

R\_Expression

;

R\_Expression:

T\_OPEN\_BRACKET R\_TypeField T\_CLOSE\_BRACKET R\_Expression

|

T\_OPEN\_BRACKET R\_Expression T\_CLOSE\_BRACKET

|

T\_OPEN\_BRACKET R\_Expression T\_CLOSE\_BRACKET R\_all\_binary\_sign R\_next\_Expression

|

R\_first\_variable

|

R\_number

|

R\_sign

|

T\_NEW R\_TypeField R\_Repeat\_square\_bracket R\_Call\_methods T\_OPEN\_BRACKET R\_ArgumentsCall T\_CLOSE\_BRACKET

|

T\_NEW R\_TypeField R\_Repeat\_square\_bracket R\_Call\_methods

|

R\_Expression T\_QUERY R\_Expression T\_COLON R\_Expression

|

R\_this\_super\_block

;

R\_this\_super\_block:

R\_this\_super R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods

|

R\_this\_super R\_Call\_methods

|

R\_this\_super R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods R\_all\_binary\_sign R\_next\_Expression

|

R\_this\_super R\_Call\_methods R\_all\_binary\_sign R\_next\_Expression

|

R\_this\_super R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods T\_INCREMENT\_DECREMENT\_SIGN

|

R\_this\_super R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods T\_INCREMENT\_DECREMENT\_SIGN R\_all\_binary\_sign R\_next\_Expression

;

R\_this\_super:

T\_THIS

|

T\_SUPER

;

R\_assignment\_sign:

T\_ASSIGNMENT

|

T\_ASSIGNMENT2

;

R\_first\_variable:

R\_variable

|

T\_IDENTIFIER R\_Repeat\_square\_bracket R\_assignment\_sign R\_Expression

;

R\_Call\_methods:

|

T\_DOT T\_IDENTIFIER R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods

|

T\_DOT T\_IDENTIFIER R\_Repeat\_square\_bracket R\_Call\_methods

|

T\_DOT R\_this\_super R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods

|

T\_DOT R\_this\_super R\_Call\_methods

;

R\_variable:

T\_IDENTIFIER R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods

|

T\_IDENTIFIER R\_Repeat\_square\_bracket R\_Call\_methods

|

T\_IDENTIFIER R\_Repeat\_bracket R\_Repeat\_square\_bracket R\_Call\_methods R\_all\_binary\_sign R\_next\_Expression

|

T\_IDENTIFIER R\_Repeat\_square\_bracket R\_Call\_methods R\_all\_binary\_sign R\_next\_Expression

|

T\_IDENTIFIER R\_Repeat\_square\_bracket T\_INCREMENT\_DECREMENT\_SIGN

|

T\_IDENTIFIER R\_Repeat\_square\_bracket T\_INCREMENT\_DECREMENT\_SIGN R\_all\_binary\_sign R\_next\_Expression

;

R\_Repeat\_bracket:

T\_OPEN\_BRACKET R\_ArgumentsCall T\_CLOSE\_BRACKET

;

R\_number:

R\_Value

|

R\_Value R\_all\_binary\_sign R\_next\_Expression

;

R\_sign:

R\_unary\_sign R\_next\_Expression

|

T\_INCREMENT\_DECREMENT\_SIGN T\_IDENTIFIER R\_Repeat\_square\_bracket

|

T\_INCREMENT\_DECREMENT\_SIGN T\_IDENTIFIER R\_Repeat\_square\_bracket R\_all\_binary\_sign R\_next\_Expression

|

T\_INCREMENT\_DECREMENT\_SIGN R\_this\_super\_block

|

T\_INCREMENT\_DECREMENT\_SIGN R\_this\_super\_block R\_all\_binary\_sign R\_next\_Expression

;

R\_next\_Expression:

R\_variable

|

R\_number

|

R\_sign

|

T\_OPEN\_BRACKET R\_TypeField T\_CLOSE\_BRACKET R\_next\_Expression

|

T\_OPEN\_BRACKET R\_Expression T\_CLOSE\_BRACKET

|

T\_OPEN\_BRACKET R\_Expression T\_CLOSE\_BRACKET R\_all\_binary\_sign R\_next\_Expression

;

R\_Command:

R\_Expression T\_SEMICOLON

|

R\_Modifiers

|

R\_Cycle

|

R\_If

|

R\_Try

|

T\_BREAK T\_SEMICOLON

|

T\_CONTINUE T\_SEMICOLON

|

T\_ASSERT R\_Expression T\_SEMICOLON

|

T\_ASSERT R\_Expression T\_COLON R\_Expression T\_SEMICOLON

|

T\_RETURN R\_Expression T\_SEMICOLON

|

T\_THROW R\_Expression T\_SEMICOLON

;

R\_Try:

T\_TRY R\_Body R\_Catch R\_Finaly

;

R\_Catch:

T\_CATCH T\_OPEN\_BRACKET R\_Arguments T\_CLOSE\_BRACKET R\_Body

|

R\_Catch T\_CATCH T\_OPEN\_BRACKET R\_Arguments T\_CLOSE\_BRACKET R\_Body

;

R\_Finaly:

|

T\_FINALLY R\_Body

;

R\_If:

T\_IF T\_OPEN\_BRACKET R\_Cond T\_CLOSE\_BRACKET R\_Body

|

T\_IF T\_OPEN\_BRACKET R\_Cond T\_CLOSE\_BRACKET R\_Body T\_ELSE R\_Body

;

R\_Cycle:

T\_FOR T\_OPEN\_BRACKET R\_Initialization\_for R\_Cond\_for T\_SEMICOLON R\_Cond\_for T\_CLOSE\_BRACKET R\_Body

|

T\_FOR T\_OPEN\_BRACKET R\_Initialization\_for T\_COLON R\_Expression T\_CLOSE\_BRACKET R\_Body

|

T\_WHILE T\_OPEN\_BRACKET R\_Cond T\_CLOSE\_BRACKET R\_Body

|

T\_DO R\_Body T\_WHILE T\_OPEN\_BRACKET R\_Cond T\_CLOSE\_BRACKET T\_SEMICOLON

;

R\_Initialization\_for:

|

R\_Modifiers

;

R\_Cond\_for:

|

R\_Expression

;

R\_Cond:

R\_Expression

;

R\_Body:

T\_OPEN\_BRACE R\_Commands T\_CLOSE\_BRACE

|

R\_Command

;

R\_all\_binary\_sign:

R\_unary\_sign

|

R\_binary\_sign

|

R\_comparison\_sign

;

R\_binary\_sign:

T\_BINARY\_SIGN

|

T\_STAR

;

R\_comparison\_sign:

T\_COMPARISON\_SIGN

;

R\_BodyClass:

T\_OPEN\_BRACE R\_BodyClass2 T\_CLOSE\_BRACE

;

R\_BodyClass2:

|

R\_BodyClass2 R\_Modifiers

;

%%

int main(int argc, char const \*argv[]){

yylval=1;

return yyparse();

}