Обфускатор

Максим Крючков, M3339

28 ноября 2018 г.

Грамматика

Построим грамматику
Она будет разбирать функци написанные на C у которых в теле возможны 5 операций:

1. Объявление переменной
2. Присваивание
3. Объявление и присваивание
4. Вызов функции
5. Возвращение значения

- input -> fun

  Будем обрабатывать функции на C.

- fun -> head body RFPAREN

  Функция состоит из верхушки, тела и закрывающей скобки

- head -> type fname LPAREN hargs RPAREN LFPAREN

  Верхушка состоит из типа возвращаемого значения, имени функции, объявлений аргументов в круглых скобках и открывающей фигурной скобки

- body ->

  1. eps
     Пустое тело
  2. body instr SEMICOLON
     Непустое тело представляется в виде последовательности инструкций, разделенных точкой с запятой.

- instr ->

  1. declaration -> type name
     Объявление переменной - это тип и имя переменной
  2. declare_assign -> type name EQUALS rvalue
     Объявление и присваивание
  3. assignment -> name EQUALS rvalue
     Присваивание
  4. return -> RETURN rvalue
     Возвращение значения

5. fcall ->fname LPAREN args RPAREN
   Вызов функции

- hargs ->

  1. %empty
     Либо у функции нет аргументов (отдельный нетерминал)
  2. type name
     Один аргумент
  3. hargs COMA type name
     Много аргументов

  Аналогично с args, только там не указывается тип

- type ->

  1. TYPE
     Обычный тип
  2. type PTR
     Указатель

- name -> NAME
  Имя переменной
- fname -> NAME
  Имя функции
- rvalue ->

  1. name
  2. fcall
  3. NUMBER
  4. STRING

Лексический анализатор

Построим класс Token для хранения токенов.

Терминал	**Token**
Тип без *	TYPE
Строка	STRING
Число	NUMBER
Последовательность пустых символов	BLANKS
Открывающие, закрывающие скобки	..
return	RETURN
=	EQUALS
*	PTR
;	SEMICOLON
,	COMA
Идентификатор, не являющийся ключевым словом	NAME
$	END