

ЛОИ

Лексический анализ

# Введение

Задача: по данной КС-грамматике  $G$  и слову  $w \in \Sigma^*$  определить  $w \in L(G)$ ?

Задача синтаксического анализа:

Если  $w \in L(G)$ , то выдать информацию о выводе  $w$  (н-р, дерево вывода);

Если  $w \notin L(G)$ , то найти ошибку;

*Компилятор* - программа, которая переводит слова (цепочку), принадлежащую одному языку (исходному) в семантике эквивалентную цепочку другого языка (целового).

Исходный - язык программирования

Целевой - язык машинных команд

# Этапы компиляции и роль лексического анализа

Этапы компиляции:

- 1) Лексический анализ - основа компилятор
- 2) Синтаксический анализ
- 3) Семантический анализ
- 4) Генерация кода

*Лексический анализ* - преобразование исходной программы в цепочку токенов, либо сообщение об ошибке.

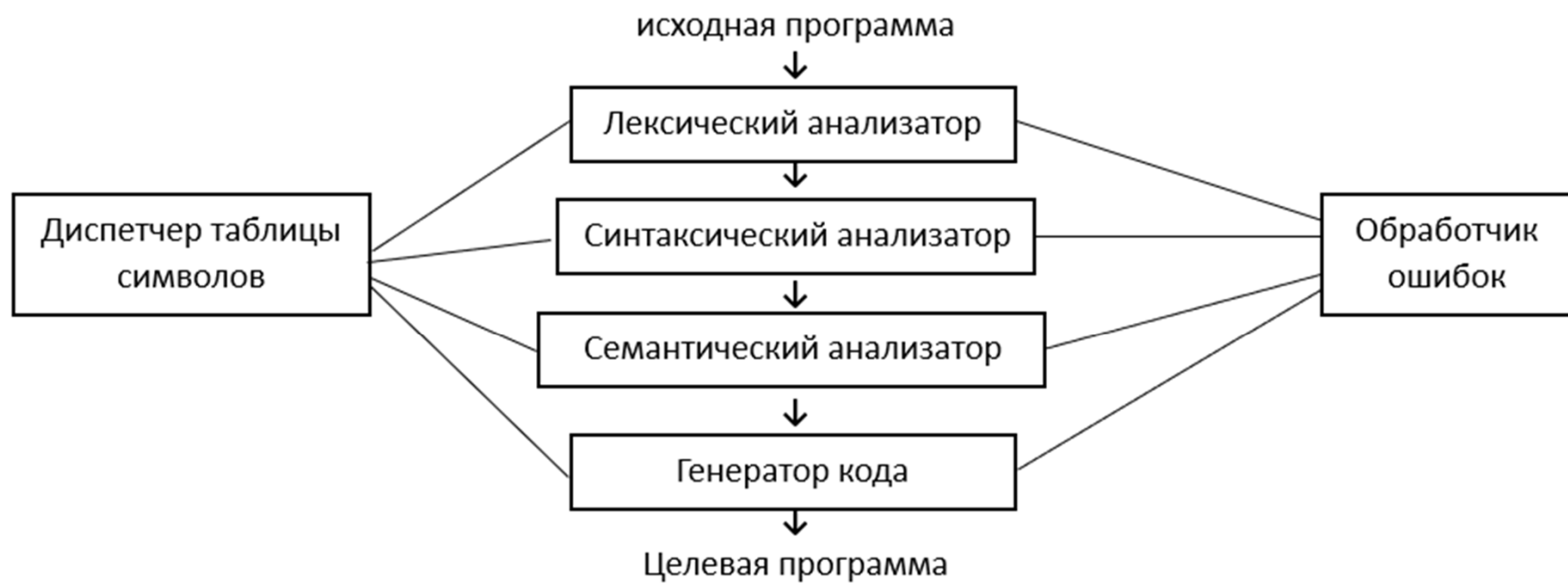


Рисунок 1: упрощенная модель компилятора

# Лексемы и токены

*Задача лексического анализа* - выделение во входной цепочке (т.е. также программы) минимальных смысловых единиц - **лексем**, и определение соответствующих лексемам синтаксических категорий языка - **токенов**. Попутно:

- Удаляются комментарии и пробелы
- Обнаруживаются ошибки некоторых видов

*Атрибут токена* - переменная, приписанная токену, значение которой указывает на некоторую характеристику токена.

*Шаблон* - регулярное выражение для данного токена.

## Пример исходного кода на Pascal

```
function grac_sum (n : integer) : real;
```

```
{вычисление суммы  $1 + 1/2 + 1/3 + \dots + 1/n$ }
```

```
begin
```

```
    grac_sum := 0;
```

```
    while n > 0 do
```

```
        begin
```

```
            grac_sum := grac_sum + 1/n;
```

```
            n := n - 1;
```

```
        end;
```

```
end;
```

## Таблица лексем и токенов для примера

№	Лексема	Токен	Смысл токена	Атрибут	Смысл атрибута
1	function	function	ключевое слово	function	ключевое слово
2	grac_sum	id	идентификатор	id_funct	идентификатор функции
3	(	sps	специальный символ	sps_(	специальный символ "открытая скобка"
4	n	id	идентификатор	id_var	идентификатор для переменной
5	:	sps	специальный символ	sps_:	специальный символ "двоеточие"
6	integer	type	тип	type	тип
7	)	sps	специальный символ	sps_)	специальный символ "закрытая скобка"
8	:	sps	специальный символ	sps_:	специальный символ ":"
9	real	type	тип	type	тип
10	;	sps	специальный символ	sps_;	специальный символ ";"

№	Лексема	Токен	Смысл токена	Атрибут	Смысл атрибута
11	begin	begin	ключевое слово	begin	ключевое слово
12	grac_sum	id	идентификатор	id_funct	идентификатор для функции
13	:=	assign	присваивание	:=	присваивание
14	0	const	константа	const_int	целая константа
15	;	Sps	специальный символ	sps_;	специальный символ ";"
16	while	while	ключевое слово	while	ключевое слово
17	n	id	идентификатор	id_var	идентификатор переменной
18	>	rel	отношение	rel_gr	отношение "больше"
19	0	const	константа	const_int	целая константа
20	do	do	ключевое слово	do	ключевое слово
21	begin	begin	ключевое слово	begin	ключевое слово



№	Лексема	Токен	Смысл токена	Атрибут	Смысл атрибута
22	grac_sum	id	идентификатор	id_funct	идентификатор функции
23	:=	assign	присваивание	:=	присваивание
24	grac_sum	id	идентификатор	id_funct	идентификатор для функции
25	+	op	операция	aop	арифметическая операция
26	1	const	константа	const_int	целая константа
27	/	op	операция	map	мультипликативная операция
28	n	id	идентификатор	id_var	идентификатор переменной
29	;	sps	специальный символ	sps_;	специальный символ ";"
30	n	id	идентификатор	id_var	идентификатор переменной
31	:=	assign	присваивание	:=	присваивание
32	n	id	идентификатор	id_var	идентификатор переменной

№	Лексема	Токен	Смысл токена	Атрибут	Смысл атрибута
33	_	op	операция	aop	арифметическая операция
34	1	const	константа	const_int	целая константа
35	end	end	ключевое слово	end	ключевое слово
36	;	;	специальный символ	sps_;	специальный символ ";"
37	end	end	ключевое слово	end	ключевое слово
38	;	sps	специальный символ	sps_;	специальный символ ";"

## Последовательность токенов

function	id	sps	id	sps	type	sps	sps	type	begin	id
	↓	↓	↓	↓	↓	↓	↓	↓		↓
	ptr <sub>1</sub>	ptr <sub>2</sub>	ptr <sub>3</sub>	ptr <sub>4</sub>	ptr <sub>5</sub>	ptr <sub>6</sub>	ptr <sub>7</sub>	ptr <sub>8</sub>		ptr <sub>9</sub>

assign ...	count	sps	while	id	rel	const	do	begin	id	sps	id	op	const
	↓												
	ptr <sub>10</sub>												

op id sps id assign id op const end sps end sps

# Автоматы для шаблонов

reg – принадлежит лексеме

reg\_ – последний прочитанный символ и принадлежащий лексеме, по его нужно прочитать, чтобы обнаружить эту лексему

other – переход по любому другому символу

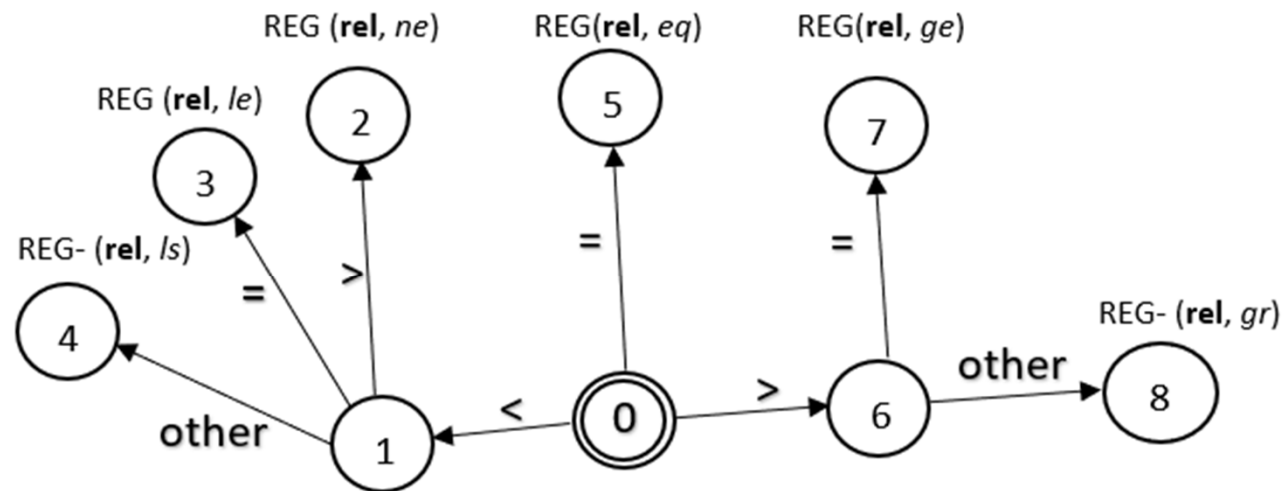


Рисунок 2: автомат для шаблона *rel*

## Автоматы для шаблонов

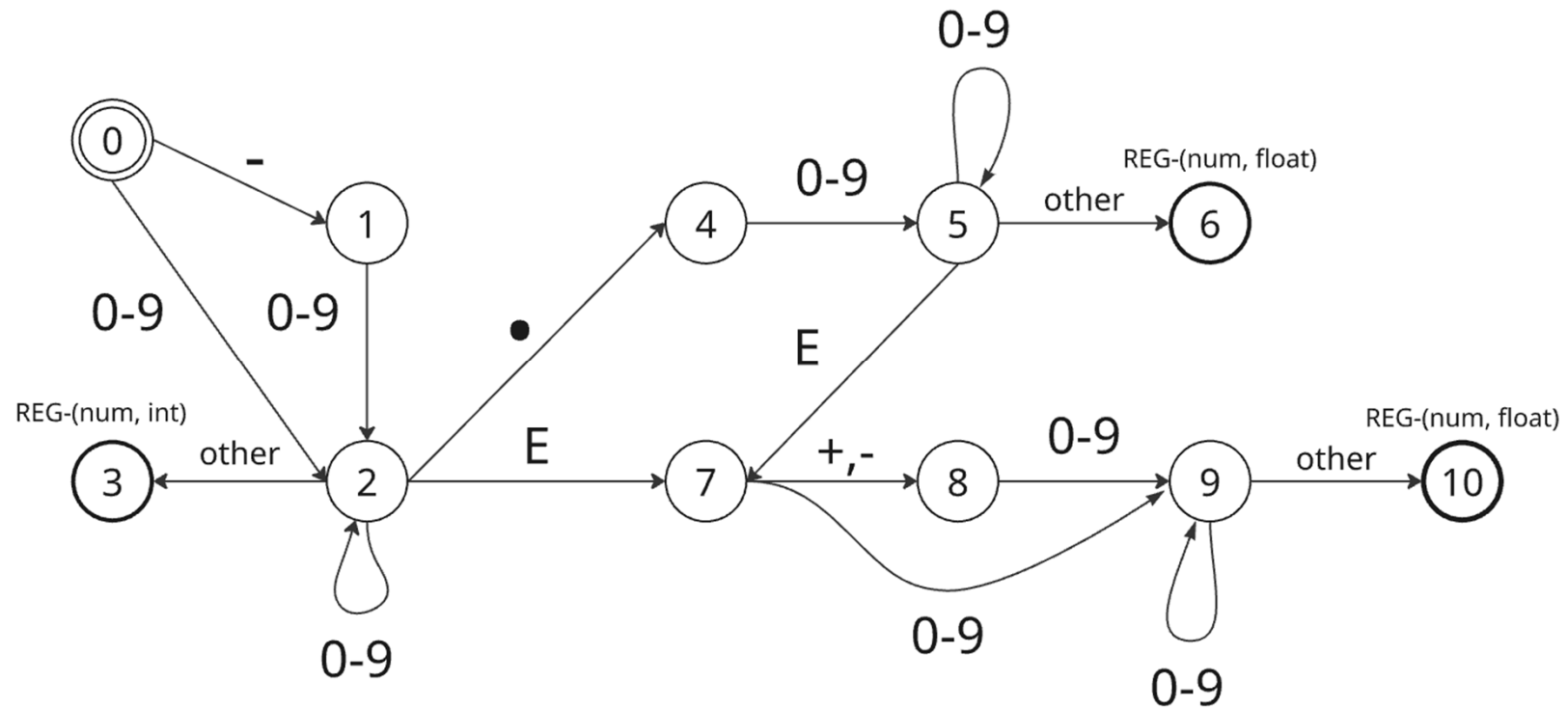


Рисунок 3: автомат для шаблона *num*

## Автоматы для шаблонов

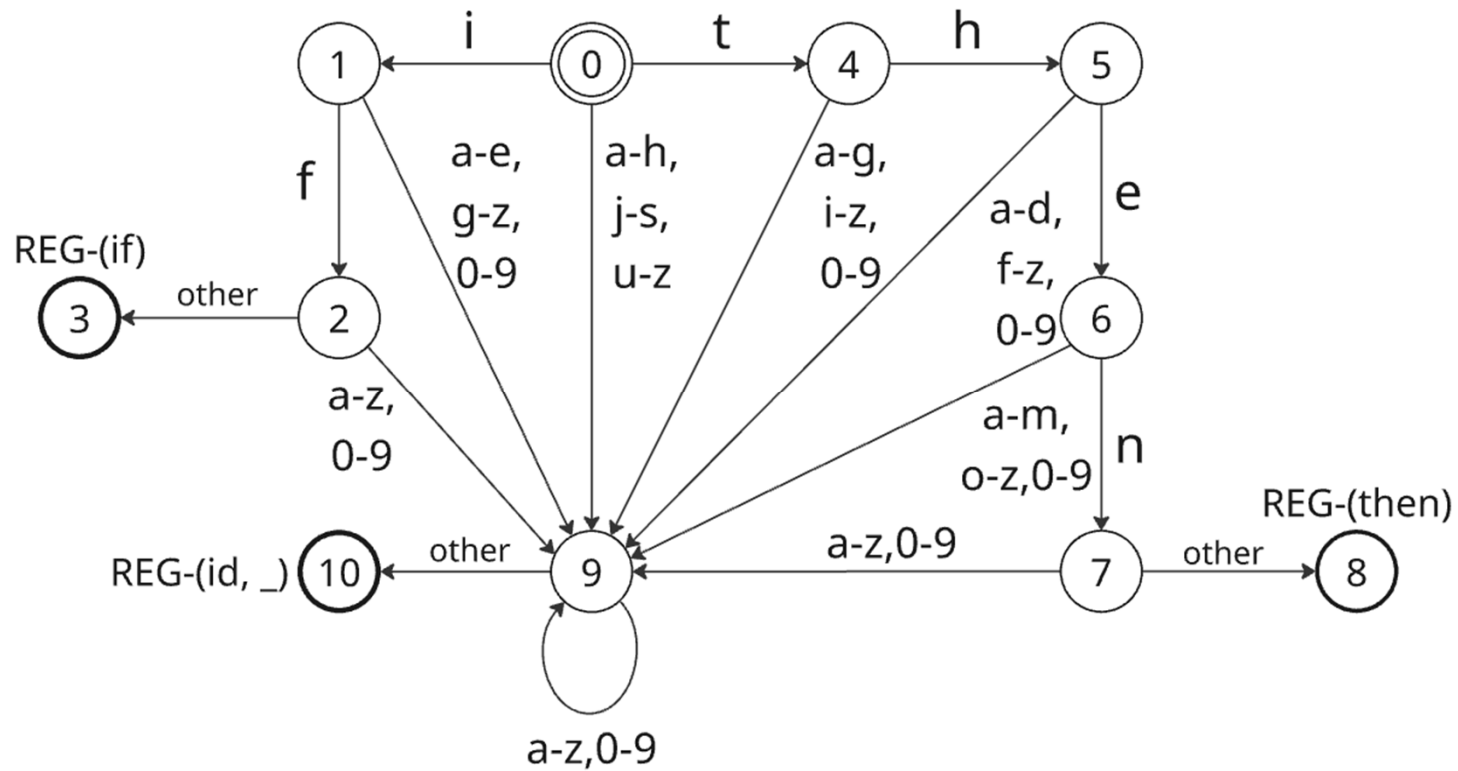


Рисунок 4: автомат для шаблона *if, then* и *id*. Не изображены дуги из вершин 1, 4, 5, 6 в вершину 10 с меткой *other*