Содержание

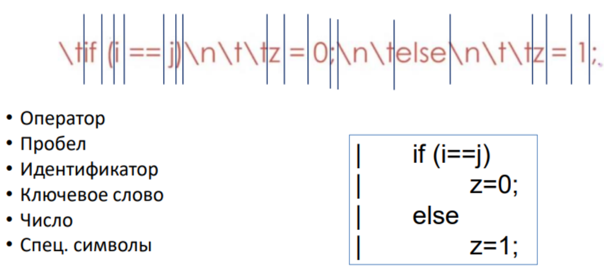
1 Билет 1

Общая структура компиляторов

Лексический анализ

Лексичсекий анализ — деление текста на слова, выделение токенов.

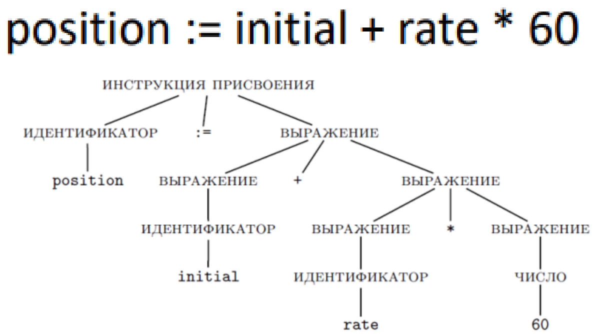
Задача ликсического анализа — выделить лексемы, классифицировать их и передать их на стадию синтаксического анализа.



Синтаксический анализ

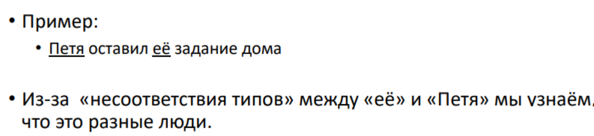
Синтаксический анализ — определение структуры предложения.

Задача: иерархическая группировка в соответсвии с грамматикой языка программирования.



Семантический анализ

Семантический анализ — устранение неоднозначностей.



Оптимизация кода

Оптимизация кода:

1. На естественном языке отимизация не имеет строгих правил и сводится к редактированию;
2. Для программ она предполагает следующее:
   (a) Увеличение скорости работы программы;
   (b) Уменьшение объёма используемой памяти;
   (c) И так далее.

Генерация кода

Генерация кода — трансляция исходного кода на другой язык программирования.

Обычно результатом является ассемблерный код.

Дополнительную информацию смотри на слайдах 1-39 первой половины лекций.

2 Билет 2

3 Билет 3

Определение алфавита

Алфавит — это конечное множество символов.

Опредделение слова в $\sum$

Словом в алфавите $\sum$ называется любая конечная последовательность символов этого алфавита.

Операции над цепочками символов

1. Конкатенация

   Опр. Если a и b — цепочки, то цепочка ab (результат приписывания цепочки b в конец цепочки a), называется *конкатенацией* (или *сцеплением*) цепочек a и b. Конкатенацию можно считать двуместной операцией над цепочками: a×b = ab.
   Например, если w = *ab* и z = *cd*, то w×z = *abcd*.
   Для любой цепочки a : aε = εa = a.
   Для любых цепочек a, b, g справедливо свойство ассоциативности операции конкатенации (ab)g = a(bg) = abg.

2. Обращение

   - **Опр.** *Обращением* (или ***реверсом***) цепочки α называется цепочка, символы которой записаны в обратном порядке.

   Обращение цепочки α будем обозначать α^R.
   Например, если α = *abcdef*, то α^R = *fedcba*.
   Для пустой цепочки: $\varepsilon^R = \varepsilon$.

3. Возведение в степень

   - **Опр.** *n*-ой степенью цепочки α (будем обозначать α^n) называется конкатенация *n* цепочек α:
   $\alpha^n = \underbrace{\alpha\alpha \ldots \alpha\alpha\alpha}_{n}$.

   Свойства степени: $\alpha^0 = \varepsilon$; $\alpha^n = \alpha\alpha^{n-1} = \alpha^{n-1}\alpha$

Дополнительную информацию смотрите на слайдах 40-42 первой презентации.

4 Билет 4

• **Опр.** Обозначим через Σ^* множество, содержащее все цепочки в алфавите Σ, включая пустую цепочку ε.
Например, если $\Sigma = \{0, 1\}$, то $\Sigma^* = \{\varepsilon, 0, 1, 00, 11, 01, 10, 000, 001, 011, \ldots\}$.

Формальное определение языка

• **Опр.** *Язык* в алфавите Σ — это подмножество множества всех цепочек в этом алфавите. Для любого языка L справедливо $L \subseteq \Sigma^*$.

Операции над языками

1. Контатенация

• **Опр.** *Конкатенацией* двух языков L_1 и L_2 называется язык
$L_3 = \{\alpha\beta \mid \alpha \in L_1,\ \beta \in L_2\}$.
Обозн.: $L_3 = L_1 L_2$
Пример: {01,111,10} {00, 01} = {0100, 0101, 11100, 11101, 1000, 1001}.

2. Объединение

• **Опр.** *Объединением* двух языков L_1 и L_2 называется язык
$L = L_1 \cup L_2 = \{\alpha \mid \alpha \in L_1 \text{ или } \alpha \in L_2\}$.
Пример: $\{01, 111, 10\} \cup \{00, 01\} = \{01, 111, 10, 00\}$.

3. Степень

• **Опр.** *Степень* языка L:
1. $L^0 = \{\varepsilon\}$
2. $L^1 = L$
3. $L^k = L^{k-1} L$

4. Итерация

- **Опр**. *Итерация* языка L:

$$L^* = \bigcup_{k=0}^{\infty} L^k \qquad L^+ = \bigcup_{k=1}^{\infty} L^k \qquad L^+ = L^* L$$

$L^* = \{\varepsilon\} \cup L \cup LL \cup LLL \cup \ldots$

Пример: $\{0,10\}^* = \{\varepsilon, 0, 10, 00, 010, 100, 1010, \ldots\}$

5 Билет 5

Способы описания языков

- Конечный язык можно описать простым перечислением его цепочек.
- Как представлять бесконечные языки?
  - спецификация (описание)
  - механизм распознавания
  - механизм порождения (генерации).
- Не каждый формальный язык можно задать с помощью конечного описания.

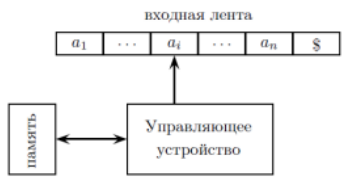
6 Билет 6

- *Спецификация* – Описание языка, как множества слов, удовлетворяющих некоторому условию. (Для регулярных языков – это регулярное выражение_.

7 Билет 7

- Механизм распознавания (*распознаватель*), по сути, является процедурой специального вида, которая по заданной цепочке определяет, принадлежит ли она языку.
- Если принадлежит, то процедура останавливается с ответом «да», т. е. *допускает* цепочку; иначе — останавливается с ответом «нет» или зацикливается.
- *Язык, определяемый распознавателем* — это множество всех цепочек, которые он допускает.

- Механизм, который является процедурой специального вида, которая по заданной цепочке определяет, принадлежит ли она языку.
- Если принадлежит, то процедура останавливается с ответом «да», т. е. *допускает* цепочку; иначе — останавливается с ответом «нет» или зацикливается.
- ***Язык, определяемый распознавателем*** — это множество всех цепочек, которые он допускает.



8 Билет 8

Опр. ***Порождающая грамматика*** G — это четверка $\langle T, N, P, S \rangle$,
где
T — алфавит терминальных символов (терминалов);
N — алфавит нетерминальных символов (нетерминалов), $T \cap N = \varnothing$;
P — конечное подмножество множества $(T \cup N)^+ \times (T \cup N)^*$;
где элемент (α, β) записывается в виде $\alpha \to \beta$ и называется *правилом вывода*;
α называется *левой частью* правила, β — *правой частью* правила.
Левая часть любого правила из P обязана содержать хотя бы один нетерминал;
S — начальный символ грамматики, $S \in N$.

Для записи правил вывода с одинаковыми левыми частями

$$\alpha \to \beta_1 \quad \alpha \to \beta_2 \quad \ldots \quad \alpha \to \beta_n$$

используют сокращенную запись $\alpha \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$.

Опр. Цепочка $\beta \in (T \cup N)^*$ *непосредственно выводима* из цепочки $\alpha \in (T \cup N)^+$ в грамматике $G = \langle T, N, P, S \rangle$ (обозначается $\alpha \to_G \beta$), если

$$\alpha = \xi_1 \gamma \xi_2, \ \beta = \xi_1 \delta \xi_2,$$

где

$$\xi_1, \xi_2, \delta \in (T \cup N)^*, \gamma \in (T \cup N)^+$$

и правило вывода $\gamma \to \delta$ содержится в P.

Опр. Цепочка $\beta \in (T \cup N)^*$ *выводима* из цепочки $\alpha \in (T \cup N)^+$ в грамматике $G = \langle T, N, P, S \rangle$ (обозначается), если существуют цепочки $\gamma_0, \gamma_1, \ldots, \gamma_n$ ($n \geq 0$), такие, что

$$\alpha = \gamma_0 \to \gamma_1 \to \ldots \to \gamma_n = \beta.$$

Последовательность $\gamma_0, \gamma_1, \ldots, \gamma_n$ называется *выводом длины n*.
Длину вывода n показывают обозначением:
Вывод за некоторое число шагов (м.б. 0 шагов) обозначается:

Индекс G в обозначении $\Rightarrow_G$ опускают, если понятно, какая грамматика подразумевается.

Опр. ***Языком, порождаемым грамматикой*** $G = \langle T, N, P, S \rangle$, называется множество

$$L(G) = \{\alpha \in T^* \mid S \Rightarrow \alpha\}.$$

Другими словами, $L(G)$ — это все цепочки в алфавите T, которые выводимы из S с помощью правил P.
Например, $L(G_{example}) = \{0^n 1^n \mid n > 0\}$.

Опр. Цепочка $\alpha \in (T \cup N)^*$, для которой $S \Rightarrow \alpha$, называется ***сентенциальной формой*** в грамматике $G = \langle T, N, P, S \rangle$.

Таким образом, ***язык, порождаемый грамматикой***, можно определить как ***множество терминальных сентенциальных форм***.

Дополнительную информацию смотри на слайдах 54-60 первой презентации.

9 Билет 9

Классификация грамматик и языков по Хомскому

- Тип грамматики определяется типом ограничений на вид правил вывода.
- Всего определено четыре типа грамматик:

  тип 0, тип 1, тип 2, тип 3.
- Каждому типу грамматик соответствует свой класс языков.
- Если язык порождается грамматикой типа i (для i = 0, 1, 2, 3), то он является *языком типа i*.

1. Тип 0

Тип 0

Любая порождающая грамматика является грамматикой типа 0.

На вид правил грамматик этого типа не накладывается никаких дополнительных ограничений.

Класс языков типа 0 совпадает с классом рекурсивно перечислимых языков.

2. Тип 1

Классификация грамматик и языков по Хомскому: **Тип 1**

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***неукорачивающей***, если правая часть каждого правила из P не короче левой части (т. е. для любого правила $\alpha \to \beta \in P$ выполняется неравенство $|\alpha| \le |\beta|$).

В виде исключения в неукорачивающей грамматике допускается наличие правила $S \to \varepsilon$, при условии, что S (начальный символ) не встречается в правых частях правил.

Грамматикой *типа 1* называют неукорачивающую грамматику.

Классификация грамматик и языков по Хомскому: **Тип 1**

Другое определение:

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***контекстно-зависимой (КЗ)***, если каждое правило из P имеет вид $\alpha \to \beta$,

где $\alpha = \xi_1 A \xi_2$, $\beta = \xi_1 \gamma \xi_2$, $A \in N$, $\gamma \in (T \cup N)^+$, $\xi_1, \xi_2 \in (T \cup N)^*$.

В виде исключения в КЗ-грамматике допускается наличие правила $S \to \varepsilon$, при условии, что S (начальный символ) не встречается в правых частях правил.

Язык, порождаемый контекстно-зависимой грамматикой, называется ***контекстно-зависимым языком***.

3. Тип 2

Классификация грамматик и языков по Хомскому: **Тип 2**

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***контекстно-свободной*** (*КС*), если каждое правило из P имеет вид $A \to \beta$, где $A \in N$, $\beta \in (T \cup N)^*$.
Заметим, что в КС-грамматиках допускаются правила с пустыми правыми частями. Язык, порождаемый контекстно-свободной грамматикой, называется ***контекстно-свободным*** языком.
Грамматикой ***типа 2*** будем называть контекстно-свободную грамматику.

4. Тип 3

Классификация грамматик и языков по Хомскому: **Тип 3**

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***праволинейной***, если каждое правило из P имеет вид $A \to wB$ либо $A \to w$, где $A, B \in N$, $w \in T^*$.

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***леволинейной***, если каждое правило из P имеет вид $A \to Bw$ либо $A \to w$, где $A, B \in N$, $w \in T^*$.

Дополнительная информация

Иерархия грамматик Хомского

Утверждение 5. Справедливы следующие утверждения:
1) любая регулярная грамматика является КС-грамматикой;
2) любая неукорачивающая КС-грамматика является КЗ-грамматикой;
3) любая неукорачивающая грамматика является грамматикой типа 0.

Утверждение 5 следует непосредственно из определений.
Рассматривая только неукорачивающие регулярные и неукорачивающие КС-грамматики, получаем следующую иерархию классов грамматик:

Регулярные неукорачивающие $\subset$ *КС неукорачивающие* $\subset$ *КЗ* $\subset$ *Тип 0*

10 Билет 10

Эквивалентность неукорачивающих и КЗ-грамматик

Утверждение 1. Пусть L — формальный язык. Следующие утверждения эквивалентны.

1) существует контекстно-зависимая грамматика G_1, такая что $L = L(G_1)$;

2) существует неукорачивающая грамматика G_2, такая что $L = L(G_2)$.

Док-во. Очевидно, что (1) $\Rightarrow$ (2): любая контекстно-зависимая грамматика удовлетворяет ограничениям неукорачивающей грамматики (см. определения).

Т.к. каждое неукорачивающее правило можно заменить эквивалентной серией контекстно-зависимых правил, следовательно (2) $\Rightarrow$ (1) .

Т.о., язык, порождаемый неукорачивающей грамматикой, - контекстно-зависимый. ■

Т.е., неукорачивающие и КЗ-грамматики определяют один и тот же класс языков.

11 Билет 11

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***контекстно-свободной*** (*КС*), если каждое правило из P имеет вид $A \to \beta$, где $A \in N$, $\beta \in (T \cup N)^*$.
Заметим, что в КС-грамматиках допускаются правила с пустыми правыми частями. Язык, порождаемый контекстно-свободной грамматикой, называется ***контекстно-свободным*** языком.

Грамматикой ***типа 2*** будем называть контекстно-свободную грамматику.

КС-грамматика может являться неукорачивающей, т.е. удовлетворять ограничениям неукорачивающей грамматики.

Утверждение 2. Для любой КС-грамматики G существует неукорачивающая КС-грамматика G', такая что $L(G) = L(G')$.

12 Билет 12

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***праволинейной***, если каждое правило из P имеет вид $A \to wB$ либо $A \to w$, где $A, B \in N$, $w \in T^*$.

Опр. Грамматика $G = \langle T, N, P, S \rangle$ называется ***леволинейной***, если каждое правило из P имеет вид $A \to Bw$ либо $A \to w$, где $A, B \in N$, $w \in T^*$.

Утверждение 3. Пусть L — формальный язык. Следующие два утверждения эквивалентны:

1) существует праволинейная грамматика G_1, такая что $L = L(G_1)$;
2) существует леволинейная грамматика G_2, такая что $L = L(G_2)$.

Т.е., праволинейные и леволинейные грамматики определяют один и тот же класс языков. Такие языки называются ***регулярными***.

- Право- и леволинейные грамматики также называют регулярными.
- Регулярная грамматика является грамматикой ***типа 3***.

13 Билет 13

Т.е., праволинейные и леволинейные грамматики определяют один и тот же класс языков. Такие языки называются ***регулярными***.

Опр. ***Автоматной*** грамматикой называется праволинейная (леволинейная) грамматика, такая, что каждое правило с непустой правой частью имеет вид:

$A \to a$ либо $A \to aB$ (для леволинейной, соответственно, $A \to a$ либо $A \to Ba$),

где $A, B \in N$, $a \in T$.

Автоматная грамматика является более простой формой регулярной грамматики.

Существует алгоритм, позволяющий по регулярной (право- или леволинейной) грамматике построить соответствующую автоматную грамматику.

Таким образом, любой регулярный язык может быть порожден автоматной грамматикой.

Существует алгоритм, позволяющий устранить из регулярной (автоматной) грамматики все ε-правила (кроме $S \to \varepsilon$ в случае, если пустая цепочка принадлежит языку; при этом S не будет встречаться в правых частях правил).

Утверждение 4. Для любой регулярной (автоматной) грамматики G существует неукорачивающая регулярная (автоматная) грамматика G', такая что $L(G) = L(G')$.

14 Билет 14

Иерархия языков

Утверждение 6. Справедливы следующие утверждения:

1) Каждый регулярный язык является КС-языком, но существуют КС-языки, которые не являются регулярными, например:

$$L = \{a^n b^n \mid n > 0\};$$

2) Каждый КС-язык является КЗ-языком, но существуют КЗ-языки, которые не являются КС-языками, например:

$$L = \{a^n b^n c^n \mid n > 0\};$$

3) Каждый КЗ-язык является языком типа 0 (т. е. рекурсивно перечислимым языком), но существуют языки типа 0, которые не являются КЗ-языками.

Из утверждения 6 следует иерархия классов языков:

$$\textit{Тип 3 (Регулярные)} \subset \textit{Тип 2 (КС)} \subset \textit{Тип 1 (КЗ)} \subset \textit{Тип 0}$$

Для k = 1, 2, 3 язык типа k является также и языком типа $k - 1$ (класс языков типа k является подклассом класса языков типа $k - 1$).

15 Билет 15

- **Утверждение 7.** Проблема «Можно ли язык, описанный грамматикой типа k (k = 0, 1, 2), описать грамматикой типа $k + 1$?» является ***алгоритмически неразрешимой***.
- Т.е., нет алгоритма, позволяющего по заданному описанию языка L (например, по грамматике), определить максимальное k, такое что L является языком типа k