# ГЛАВА 2

## Основы теории формальных языков и грамматик

**2.1 Языки и цепочки символов. Способы задания языков**
- **2.1.1 Цепочки символов. Операции над ними**
- **2.1.2 Формальное определение языка. Понятие языка**
- **2.1.3 Способы задания языка**
- **2.1.4 Синтаксис и семантика**

**2.2 Определение грамматики**
- **2.2.1 Понятие о грамматике языка**
- **2.2.2 Формальное определение грамматики**

**2.3 Способы записи синтаксиса языка**
- **2.3.1 Метаязык Хомского**
- **2.3.2 Бэкуса-Наура формы (БНФ)**
- **2.3.3 РБНФ (расширенная)**
- **2.3.4 Диаграмма Вирта**

**2.4 Классификация языков и грамматик**
- **2.4.1 Классификация грамматик**
- **2.4.2 Классификация языков**
- **2.4.3 Примеры классификации языков и грамматик**

**2.5 Цепочки вывода. Сентенциальная форма**
- **2.5.1 Вывод. Цепочка вывода.**
- **2.5.2 Сентенциальная форма грамматики. Основа**
- **2.5.3 Левосторонний и правосторонний вывод**
- **2.5.4 Дерево вывода**

# ГЛАВА 2. Основы теории формальных языков и грамматик

## *2.1 ЯЗЫКИ И ЦЕПОЧКИ СИМВОЛОВ. СПОСОБЫ ЗАДАНИЯ ЯЗЫКОВ*

# 2.1.1 Цепочки символов. Операции над ними

- **Цепочкой (строкой)** называется последовательность символов записанных один за одним. α β γ ω
- Цепочка – последовательность, в которую могут входить все допустимые символы (не обязательно несущие смысл). abc или call_me_1_02
- Цепочки символов α и β равны (α = β) тогда и только тогда, когда имеют один и тот же состав символов, и одинаковое их количество и их порядок следования.
- Количество символов в цепочке называется длиной цепочки. |α|

  α=abc |α| = 3

  α = β |α| = |β|

# 2.1.1 Цепочки символов. Операции над ними

- Если из цепочки единичной длины $|\alpha|=1$ удаляется этот единственный символ, $\alpha$ по прежнему остается цепочкой, но длина ее равна 0. $|\alpha|=0$
- Цепочку нулевой длины будем обозначать $\varepsilon$.

  $|\varepsilon|=0$

  $\varepsilon d=d\varepsilon$
- Если существует цепочка $\omega = \alpha\beta$, то $\alpha$ – голова цепочки, $\beta$ – хвост цепочки.
- Причем $\alpha$ – правильная голова, если $\beta$ – не пустая цепочка. $|\beta| >0$. $\beta$ –правильный хвост, если $\alpha$ – не пустая цепочка. $|\alpha| > 0$.

  $\alpha = abc$

  $\varepsilon$, a, ab, abc – головы цепочки. $\varepsilon$, a, ab – правильные головы.

# 2.1.1 Цепочки символов. Операции над ними

- Если α и β - цепочки, то цепочка αβ называется **конкатенацией** (или сцеплением) цепочек α и β.

  α = ab и β = cd, αβ = abcd.

  αε = εα = α.

- Коммутативность конкатенации $\alpha\beta \neq \beta\alpha$, ассоциативность $\alpha(\beta\gamma) = (\alpha\beta)\gamma$
- **Обращением (или реверсом)** цепочки α называется цепочка, символы которой записаны в обратном порядке. $\alpha^R$.

  α = abcdef, $\alpha^R$ = fedcba.

  $\varepsilon = \varepsilon^R$.

  $(\alpha\beta)^R = \beta^R\alpha^R$

- **Итерация (повторение, степень)** n-ой степенью цепочки α (будем обозначать $\alpha^n$) называется конкатенация n цепочек α.

  $\alpha^0 = \varepsilon$; $\alpha^n = \alpha\alpha^{n-1} = \alpha^{n-1}\alpha$.

  $\varepsilon^n = \varepsilon$, где $n \in N$, $n >= 0$.

# 2.1.2 Формальное определение языка. Понятие языка

- **Язык** – это заданный набор символов и правил, устанавливающих способы комбинации этих символов между собой для записи осмысленных предложений.
- **Алфавит** – набор допустимых символов языка. Алфавит – счетное, непустое множество символов.
- Цепочка символов α является цепочкой над алфавитом α(V), если в нее входят только символы, принадлежащие алфавиту V.
- Для любого алфавита V пустая цепочка ε может как являться, так и не являться цепочкой над этим алфавитом.
- Если |V|=0 и V – множество, то оно называется пустым множеством и обозначается $.

  | ε |=0

  | {ε} |=1

# 2.1.2 Формальное определение языка. Понятие языка

- V* множество, содержащее все цепочки в алфавите V, включая пустую цепочку ε.
- V* - **итерация** множества V или транзитивное замыкание.
- V+ - множество всех цепочек длиной 1 и более, исключив тем самым цепочку ε.
- V+ - **усечённая итерация** множества V или усеченное транзитивное замыкание.

  $V^* = V^+ \cup \{\varepsilon\}$

  $V = \{a,b,c\}$

  $V^* = \{a, b, c, aa, bb, cc, aab, abc, abbc \ldots \varepsilon\}$

  $V^+ = \{a, b, c, aa, bb, cc, aab, abc, abbc \ldots\}$
- **Декартовым произведением** $A \times B$ множеств A и B называется множество $\{ \alpha\beta \mid \alpha \in A, \beta \in B\}$.

  Если $A= \{a,b\}$ и $B=\{c,d\}$ , то $A \times B = \{ac, ad, bc, bd\}$

# 2.1.2 Формальное определение языка. Понятие языка

- **Языком L** над алфавитом V называют некоторое счетное подмножество цепочек конечной длины из множества всех цепочек над алфавитом V. $L(V) \leq V^*$
  - ✔ множество цепочек языка не обязано быть конечным
  - ✔ хотя каждая цепочка в языке обязана быть конечной длины, эта длина формально ничем не ограничена
- **Предложением языка** называется цепочка символов, принадлежащая этому языку.

# 2.1.2 Формальное определение языка. Понятие языка

- Язык L над алфавитом V включает в себя язык L’ над алфавитом V (L(V) ≤ L’(V)), если справедливо, что любая цепочка α принадлежащая L, принадлежит и L’. α є L(V) и α є L’(V)
- **Два языка** L(V) и L’(V) **равны** или совпадают если справедливо L(V) ≤ L’(V) и L’(V) ≤ L(V).
- **Два языка** L(V) и L’(V) **почти эквиваленты**, если они отличаются на пустую цепочку L(V) =~ L’(V).

  $L(V) \cup \{\varepsilon\} = L'(V) \cup \{\varepsilon\}$ .

# 2.1.3 Способы задания языка

- перечисление всех допустимых цепочек языка
- с помощью указания способа порождения цепочек языка (задать грамматику языка, используемую для порождения языка)

$L(G1)=\{0^n1^n \mid n>0\}$

01

0011

000111

$L(G1)=\{0^n1^m \mid n>0, m>0\}$

01

0000011

00111

- определение метода распознавания цепочек языка

# ***2.1.4 Синтаксис и семантика***

- **Лексема** – это языковая конструкция, которая состоит из элементов алфавита языка и не содержит других конструкций.
- **Синтакис** – набор правил, определяющих допустимые конструкции языка. Синтаксис определяет форму языка.
- **Семантика** – это раздел языка, определяющий значения предложений языка (определяющий содержание, смысл языка).

## ГЛАВА 2. Основы теории формальных языков и грамматик

# 2.2 ОПРЕДЕЛЕНИЕ ГРАММАТИКИ

# 2.2.1 Понятие о грамматике языка

- **Грамматика** – описание способов построения предложений некоторого языка.
- Грамматика — один из основных подходов к описанию бесконечного формального языка конечными средствами.
- **Правило** (**продукция**) – упорядоченная пара цепочек (α β ), которое записывается α −> β (α порождает β ).
- L(G) – язык L, заданный грамматикой G.

# 2.2.1 Понятие о грамматике языка

- Грамматики G1 и G2 называются **эквивалентными**, если L(G1) = L(G2).

  G1 = ({0,1}, {A,S}, P1, S)  G2 = ({0,1}, {S}, P2, S)

  P1: S -> 0A1  P2: S -> 0S1 | 01

  0A -> 00A1

  A -> ε

  $L(G1) = L(G2) = \{0^n1^n \mid n>0\}$.

- Грамматики G1 и G2 **почти эквивалентны**, если $L(G1) \cup \{\varepsilon\} = L(G2) \cup \{\varepsilon\}$.

  G1 = ({0,1}, {A,S}, P1, S)  G2 = ({0,1}, {S}, P2, S)

  P1: S -> 0A1  P2: S -> 0S1 | ε

  0A -> 00A1

  A -> ε

  $L(G1)=\{0^n1^n \mid n>0\}$  $L(G2)=\{0^n1^n \mid n>=0\}$

# 2.2.2 Формальное определение грамматики

По определению Хомского формальная грамматика представляет собой четвёрку:

**G={VT, VN, P, S}**

- VT, T - множество терминальных символов языка,
- VN, N – множество нетерминальных символов (или понятий языка или синтаксических единиц)

  $V = VN \cup VT$

  $VN \cap VT = \varnothing$
- P – множество правил подстановки (продукций), имеющий вид $\alpha \to \beta$, $\alpha \in V^+$, $\beta \in V^*$.

  Знак -> означает "непосредственно порождает "или "есть по определению".
- S – аксиома грамматики или начальный символ грамматики. $S \in VN$.

# ***2.2.2 Формальное определение грамматики***

Грамматика, определяющая целое число без знака:

G={VT,VN,P,S}

VN = {A,B}

VT = {0,1,2,3,4,5,6,7,8,9}

P = {A ->BA, A ->B, B ->0, B ->1, B->9}

S = {A}

A - целое число без знака, B - любая цифра.

# ГЛАВА 2. Основы теории формальных языков и грамматик

## *2.3 СПОСОБЫ ЗАПИСИ СИНТАКСИСА ЯЗЫКА*

*Метаязык - язык, предназначенный для описания другого языка*

# ***2.3.1 Метаязык Хомского***

- -> символ отделяет левую часть правила от правой (читается как "порождает" и "это есть");
- нетерминалы обозначаются буквой A с индексом, указывающим на его номер;
- терминалы - это символы, используемые в описываемом языке;
- Каждое правило определяет порождение одной новой цепочки, причём один и тот же нетерминал может встречаться в нескольких правилах слева.

# 2.3.1 Метаязык Хомского

| | | | |
|---|---|---|---|
| 1. $A_1 \to A$ | 17. $A_1 \to Q$ | 33. $A_1 \to g$ | 49. $A_1 \to w$ |
| 2. $A_1 \to B$ | 18. $A_1 \to R$ | 34. $A_1 \to h$ | 50. $A_1 \to x$ |
| 3. $A_1 \to C$ | 19. $A_1 \to S$ | 35. $A_1 \to i$ | 51. $A_1 \to y$ |
| 4. $A_1 \to D$ | 20. $A_1 \to T$ | 36. $A_1 \to j$ | 52. $A_1 \to z$ |
| 5. $A_1 \to E$ | 21. $A_1 \to U$ | 37. $A_1 \to k$ | 53. $A_2 \to 0$ |
| 6. $A_1 \to F$ | 22. $A_1 \to V$ | 38. $A_1 \to l$ | 54. $A_2 \to 1$ |
| 7. $A_1 \to G$ | 23. $A_1 \to W$ | 39. $A_1 \to m$ | 55. $A_2 \to 2$ |
| 8. $A_1 \to H$ | 24. $A_1 \to X$ | 40. $A_1 \to n$ | 56. $A_2 \to 3$ |
| 9. $A_1 \to I$ | 25. $A_1 \to Y$ | 41. $A_1 \to o$ | 57. $A_2 \to 4$ |
| 10. $A_1 \to J$ | 26. $A_1 \to Z$ | 42. $A_1 \to p$ | 58. $A_2 \to 5$ |
| 11. $A_1 \to K$ | 27. $A_1 \to a$ | 43. $A_1 \to q$ | 59. $A_2 \to 6$ |
| 12. $A_1 \to L$ | 28. $A_1 \to b$ | 44. $A_1 \to r$ | 60. $A_2 \to 7$ |
| 13. $A_1 \to M$ | 29. $A_1 \to c$ | 45. $A_1 \to s$ | 61. $A_2 \to 8$ |
| 14. $A_1 \to N$ | 30. $A_1 \to d$ | 46. $A_1 \to t$ | 62. $A_2 \to 9$ |
| 15. $A_1 \to O$ | 31. $A_1 \to e$ | 47. $A_1 \to u$ | 63. $A_3 \to A_1$ |
| 16. $A_1 \to P$ | 32. $A_1 \to f$ | 48. $A_1 \to v$ | 64. $A_3 \to A_3 A_1$ |
| | | | 65. $A_3 \to A_3\ A_2$ |

Описание идентификатора на метаязыке Хомского

# 2.3.2 Бэкуса-Наура формы (БНФ)

- символ "::=" отделяет левую часть правила от правой;
- нетерминалы обозначаются произвольной символьной строкой, заключённой в угловые скобки "<" и ">";
- терминалы – это символы, используемые в описываемом языке;
- каждое правило определяет порождение нескольких альтернативных цепочек, отделяемых друг от друга символом вертикальной черты “|”.

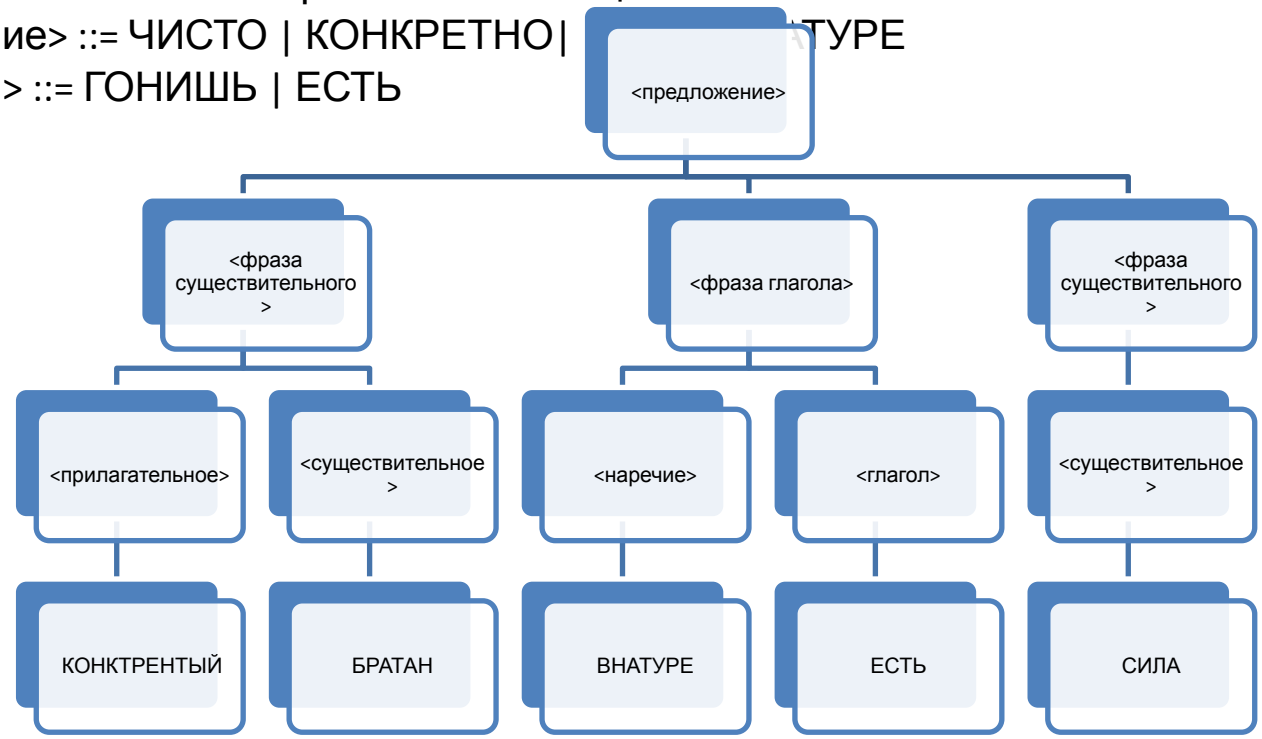
# 2.3.2 Бэкуса-Наура формы (БНФ)

1. <буква> ::= A | B| C …| Z| a| b| c| …| z
2. <цифра> ::= 0| 1| 2 … |9
3. <идентификатор> ::= <буква> | <идентификатор><буква>| <идентификатор><цифра>

Пример описания идентификатора с использованием БНФ

# 2.3.2 Бэкуса-Наура формы (БНФ)

- <предложение> ::= <фраза существительного> <фраза глагола> <фраза существительного>
- < фраза существительного > ::= <прилагательное> <существительное> | <существительное>
- <прилагательное> ::= БЛАТНОЙ| КОНКРЕТНЫЙ
- <существительное> ::= ПАЦАН| БРАТАН| СИЛА
- <фраза глагола> ::= <наречие> <глагол> | <глагол>
- < наречие> ::= ЧИСТО | КОНКРЕТНО| [illegible]ТУРЕ
- <глагол> ::= ГОНИШЬ | ЕСТЬ




Упрощенная грамматика русского языка в терминах БНФ

# ***2.3.3 РБНФ (расширенная)***

- [ ] – синтаксическая конструкция может отсутствовать;
- { } – повторение синтаксической конструкции (возможно 0 раз)
- ( ) – для ограничения альтернативных конструкций
- {\ \} – для обозначения повторения один и более раз.

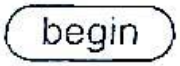
# ***2.3.3 РБНФ***

- <буква> ::= A | B| C …| Z| a| b| c| …| z
- <цифра> ::= 0| 1| 2 … |9
- <идентификатор> ::= <буква> {<буква> | <цифра>}

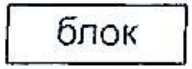
Пример описания идентификатора с использованием РБНФ

# 2.3.4 Диаграмма Вирта

терминальный символ, принадлежащий алфавиту языка

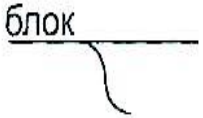



постоянная группа терминальных символов, определяющая название лексемы, ключевое слово и т.д.



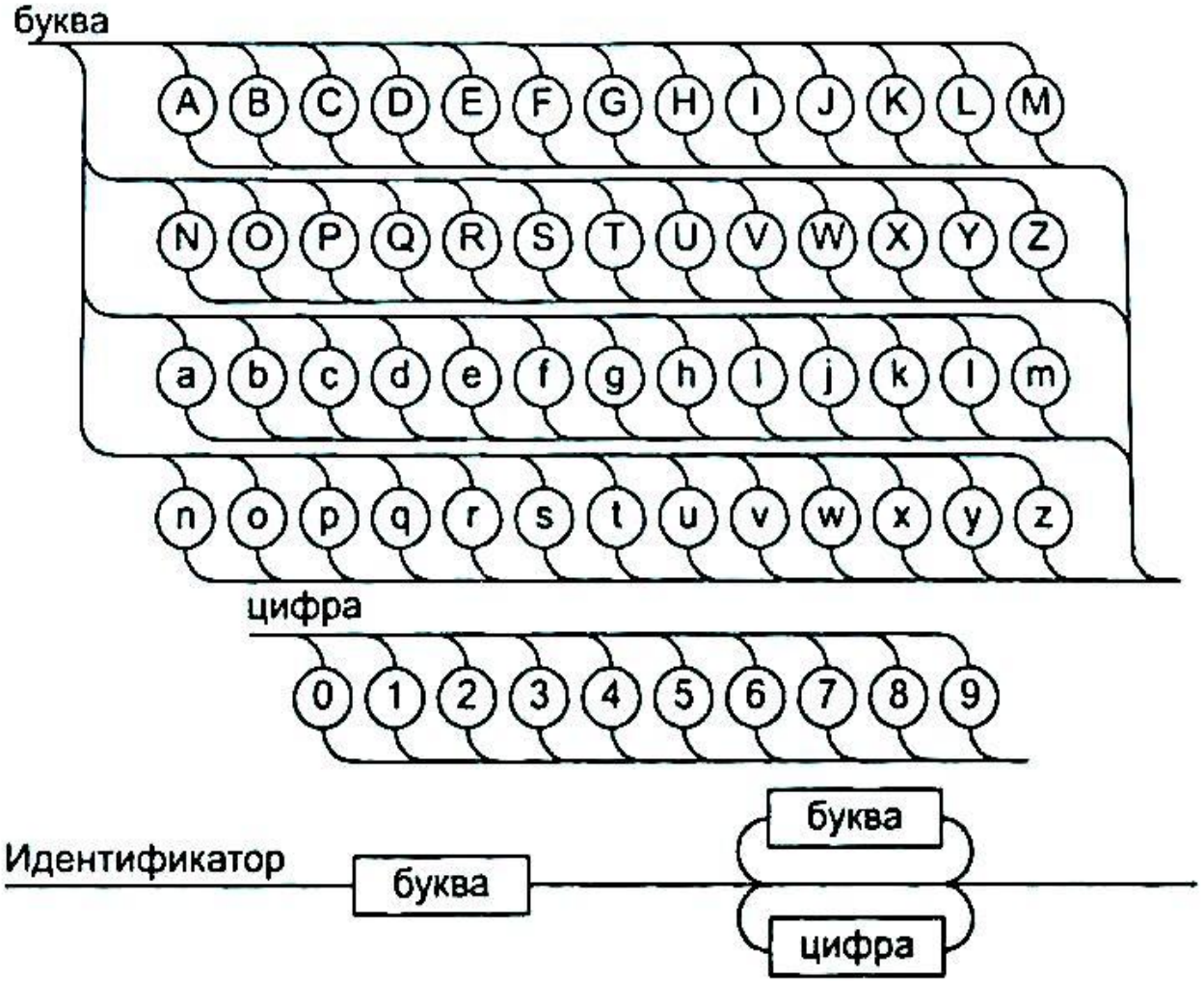

нетерминальный символ определяющий название правила

соединительные линии, обеспечивающие связь между терминальными и нетерминальными символами в правилах, заданных диаграммами Вирта




входная дуга с именем правила, определяющая его начало

# 2.3.4 Диаграмма Вирта




Пример описания идентификатора с использованием диаграмм Вирта

# ГЛАВА 2. Основы теории формальных языков и грамматик

## *2.4 КЛАССИФИКАЦИЯ ЯЗЫКОВ И ГРАММАТИК*

# 2.4.1 Классификация грамматик

| Тип | Название | Ограничение на правила |
|---|---|---|
| 0 | Грамматики с фразовой структурой (грамматики без ограничений) | $\alpha \to \beta$, где $\alpha \in V^+$, $\beta \in V^*$ |
| 1 | Контекстно-зависимые | $\xi_1 A \xi_2 \to \xi_1 \gamma \xi_2$, где $A \in VN$; $\gamma \in V^+$; $\xi_1, \xi_2 \in V^*$. |
| | неукорачивающие | $\alpha \to \beta$, где $\alpha, \beta \in V^+$ и $\lvert \alpha \rvert <= \lvert \beta \rvert$ |

# 2.4.1 Классификация грамматик

| Тип | Название | Ограничение на правила |
|---|---|---|
| 2 | Контекстно-свободные:<br>неукорачивающие<br>укорачивающие | <br><br>$A \to \beta$, где $A \in V_N, \beta \in V^+$<br>$A \to \beta$, где $A \in V_N, \beta \in V^*$ |
| 3 | Регулярные:<br>леволинейные<br>праволинейные<br>Подкласс регулярных – автоматные:<br>леволинейные<br>праволинейные | <br><br>$A \to B\gamma \mid \gamma$, где $A, B \in V_N, \gamma \in V_T^*$<br>$A \to \gamma B \mid \gamma$, где $A, B \in V_N, \gamma \in V_T^*$<br><br><br><br>$A \to Bt \mid t$, где $A, B \in V_N, t \in V_T$<br>$A \to tB \mid t$, где $A, B \in V_N, t \in V_T$ |

# 2.4.1 Классификация грамматик

- Эта иерархия грамматик – включающая.
- Грамматика 2 включает 3, но не наоборот.
- Любая грамматика относится к типу 0.
- Существуют такие УКС грамматики, которые не относятся к КЗ и неукорачивающим, а относятся к типу без ограничений.
- Сложность грамматики обратно пропорциональна тому максимально возможному номеру типа к которому может быть отнесена грамматика.

# 2.4.2 Классификация языков

Языки классифицируются в соответствие с типами грамматик с помощью которых они заданы. Поскольку один и тот же язык в общем случае может быть задан сколь угодным количеством грамматик, которые могут относится к разным типам, то для классификации языка всегда выбирается грамматика с максимальным классификационным типом.

# 2.4.2 Классификация языков

Грамматика 0 G1 = ({0,1}, {A,S}, P1, S) и

P1: S -> 0A1

0A -> 00A1

A -> ε

КС-грамматика G2 = ({0,1}, {S}, P2, S), где

P2: S -> 0S1 | 01

описывают один и тот же язык

$L = L(G1) = L(G2) = \{ 0^n1^n \mid n>0\}$

# 2.4.2 Классификация языков

- Сложность языка убывает с возрастанием классификационного типа языка.
- Тип 0. Язык с фразовой структурой (естественные языки).
- Тип 1. Язык контекстно-зависимый.

В общем случае время на распознавание предложения этого языка экспоненциально зависит от длины входящей цепочки.

- Тип 2. Контекстно-свободный язык.

Время распознавания предложений КС-языка полиномиально зависит от длины входящей цепочки.

- Тип 3. Регулярные.

  Время распознавания предложений языка линейно зависит от длины входящей цепочки.

# 2.4.3 Примеры классификации языков и грамматик

- Язык типа 2: $L(G3) = \{(ac)^n (cb)^n \mid n > 0\}$

G3: S -> aQb | accb

Q -> cSc

- Язык типа 3: $L(G4) = \{\omega \perp \mid \omega \in \{a,b\}^+$, где нет двух рядом стоящих a}

G4: S -> A⊥ | B⊥

A -> a | Ba

B -> b | Bb | Ab

# 2.4.3 Примеры классификации языков и грамматик

Язык типа 0:

L(G1) = $\{a^2 b^{n^2-1} \mid n >= 1\}$

G1: S -> aaCFD

F -> AFB | AB

AB -> bBA

Ab -> bA

AD -> D

Cb -> bC

CB -> C

bCD -> ε

Язык типа 1:

L(G2) = { $a^n b^n c^n$, n >= 1}

G2: S -> aSBC | abC

CB -> BC

bB -> bb

bC -> bc

cC -> cc

# ГЛАВА 2. Основы теории формальных языков и грамматик

## *2.5 ЦЕПОЧКИ ВЫВОДА. СЕНТЕНЦИАЛЬНАЯ ФОРМА*

# 2.5.1 Вывод. Цепочка вывода.

- **Выводом** называется процесс порождения предложений языка на основе правил, определяющих язык.
- Цепочка $\beta \in (VT \cup VN)^*$ **непосредственно выводима** из цепочки $\alpha \in (VT \cup VN)^+$ в грамматике $G = (VT, VN, P, S)$ (обозначим $\alpha \Rightarrow \beta$), если $\alpha = \xi_1\gamma\xi_2$, $\beta = \xi_1\delta\xi_2$, где $\xi_1, \xi_2, \delta \in (VT \cup VN)^*$, $\gamma \in (VT \cup VN)^+$ и правило вывода $\gamma \to \delta$ содержится в P.

# 2.5.1 Вывод. Цепочка вывода.

- Цепочка $\beta \in V^*$ выводима из цепочки $\alpha \in V^+$ в грамматике $G = (VT, VN, P, S)$ (обозначим $\alpha \Rightarrow^* \beta$), если:
  - $\beta$ непосредственно выводима из $\alpha$ ($\alpha \Rightarrow \beta$)
  - существует такая цепочка $\gamma$, что $\beta$ непосредственно выводима из $\gamma$ ($\gamma \Rightarrow \beta$), а $\gamma$ выводима из $\alpha$ ($\alpha \Rightarrow^* \gamma$)
- **$\beta$ выводима из $\alpha$** если $\beta$ непосредственно выводима из $\alpha$ или если можно построить цепочку непосредственных выводов $\alpha \Rightarrow \gamma 0 \Rightarrow \gamma 1 \Rightarrow \ldots \Rightarrow \gamma n \Rightarrow \beta$.
- Такая последовательность непосредственно выводимых цепочек называется **цепочкой вывода**.
- Каждый переход от одной цепочки к другой называется **шагом вывода**.

# 2.5.1 Вывод. Цепочка вывода.

- Если цепочка вывода от α к β содержит одну и более промежуточных цепочек, то такая цепочка обозначается $\alpha \Rightarrow^{+} \beta$ (**β нетривиально выводима из α**).
- Если количество шагов вывода известно, то его можно указать непосредственно над знаком вывода.

  Если $\alpha \Rightarrow \beta$, то один шаг вывода.

  Если $\alpha \Rightarrow^{4} \beta$, то β выводима из α за 4 шага.

  Если $\alpha \Rightarrow^{0} \beta$, то $\alpha = \beta$.

# 2.5.1 Вывод. Цепочка вывода.

G=({A, S) {0, 1}, P, S)

P: S -> 0A1

0A -> 00A1

A -> ε

Приведем вывод для цепочки 0011 є L(G):

- $S \Rightarrow 0A1 \Rightarrow 00A11 \Rightarrow 0011$;
- $S \Rightarrow^{+} 0A1$; $S \Rightarrow + 00A11$; $S \Rightarrow + 0011$;
- $S \Rightarrow^{*} S$; $S \Rightarrow^{m} 0A1$; $S \Rightarrow^{*} 00A11$; $S \Rightarrow^{3} 0011$;

# 2.5.2 Сентенциальная форма грамматики. Основа

- Вывод называется законченным, если на основе цепочки β, полученной в результате вывода нельзя сделать ни одного шага вывода, т.е. цепочка β состоит из терминальных символов.
- Цепочка β, полученная в результате законченного вывода называется **конечной цепочкой** вывода.
- Цепочка $\alpha \in (VT \cup VN)^*$, для которой $S \Rightarrow^* \alpha$ (если α выводима из начального символа грамматики), называется **сентенциальной формой** в грамматике G = (VT, VN, P, S).
- Если $\alpha \in VT^*$, то α называется **конечной сентенциальной формой** или предложением языка.

# 2.5.2 Сентенциальная форма грамматики. Основа

- Пусть G=(VN, VT, P, S) грамматика и цепочка w = $\gamma_1$ β $\gamma_2$ сентенциальная форма $\gamma_1$, $\gamma_2$ є V*, β є V+, тогда β называют **фразой сентенциальной формы** w для нетерминала B, если существуют выводы $S \Rightarrow^* \gamma_1 B \gamma_2$, $B \Rightarrow^+ \beta$.
- β называется **простой фразой**, если существуют выводы $S \Rightarrow^* \gamma_1 B \gamma_2$, $B \Rightarrow \beta$.
- Основой всякой сентенциальной формы называется самая левая простая фраза.

  если $\gamma_1 = \varepsilon$, то B - голова.

  если $\gamma_2 = \varepsilon$, то B - хвост.
- Язык L заданный грамматикой G - это множество всех конечных сентенциальных форм грамматики G.

# 2.5.3 Левосторонний и правосторонний вывод

- Вывод цепочки $\beta \in (VT)^*$ из $S \in VN$ в КС-грамматике $G = (VT, VN, P, S)$, называется левым (левосторонним), если в этом выводе каждая очередная сентенциальная форма получается из предыдущей заменой самого левого нетерминала.
- Вывод цепочки $\beta \in (VT)^*$ из $S \in VN$ в КС-грамматике $G = (VT, VN, P, S)$, называется правым (правосторонним), если в этом выводе каждая очередная сентенциальная форма получается из предыдущей заменой самого правого нетерминала.

# 2.5.3 Левосторонний и правосторонний вывод

Например, для цепочки a+b+a в грамматике

G = ({a,b,+}, {S,T}, {S -> T | T+S; T -> a | b}, S)

можно построить выводы:

(1) S ⇒ T+S ⇒ T+T+S ⇒ T+T+T ⇒ a+T+T ⇒ a+b+T ⇒ a+b+a

(2) S ⇒ T+S ⇒ a+S ⇒ a+T+S ⇒ a+b+S ⇒ a+b+T ⇒ a+b+a

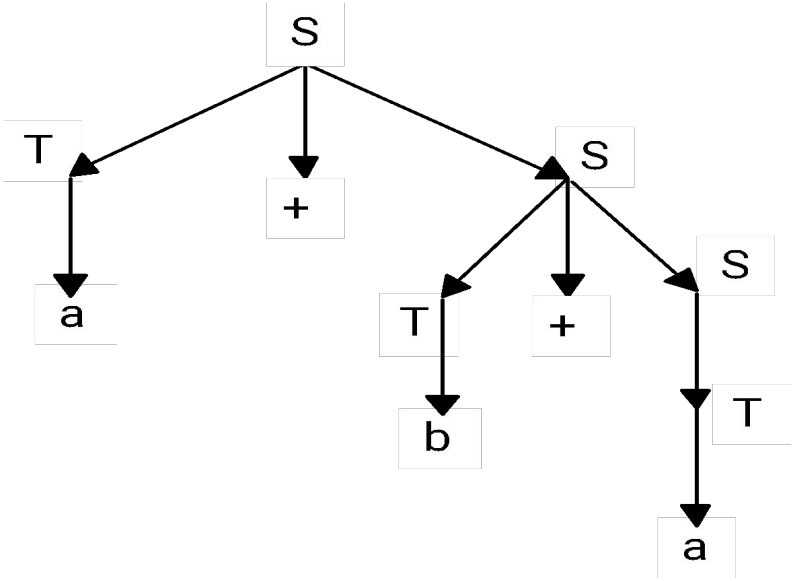
(3) S ⇒ T+S ⇒ T+T+S ⇒ T+T+T ⇒ T+T+a ⇒ T+b+a ⇒ a+b+a

Для грамматик типов 2,3 для любой сентенциальной формы можно построить левый и правый вывод.

Для грамматик типов 0,1 – не всегда.

# 2.5.4 Дерево вывода

- Цепочку вывода можно представить графически в форме дерева (графа).
- Корнем дерева является вершина, обозначенная начальным символом грамматики.
- В узлах дерева находятся нетерминальные символы, в листьях – терминалы.
- Каждая связь соответствует одному шагу вывода.




Процесс распознавания цепочки символов – можно ли построить для данной цепочки дерево назовем синтаксическим анализом, а само дерево – синтаксическим.