فصل ۲

خواص عمومی زبان ها و انواع گرامرها

(یاد آوری نظریه زبان ها)

الفبا و رشته

- **الفبا**: مجمعه متناهی از علائم (نمادها) را الفبا می گویند و با سیگما (Σ) نشان می‌دهند.
- **رشته (String)**: دنباله‌ای متناهی از علائم یک الفبا را رشته می‌گویند. مثال: S = aababbabaab
- **طول رشته**: به تعداد الفبای موجود در یک رشته، طول رشته می‌گویند. طول رشته S را با |S| یا n(S) نمایش می‌دهند.
- هر رشته شامل بخش‌هایی به صورت زیر است:
  - **پیشوند (Prefix)**: رشته حاصل از حذف یک یا چند مورد از علائم الفبا از انتهای رشته باعث ایجاد پیشوند رشته می‌گردد. مثال: رشته sara را در نظر بگیرید. پیشوندهای این رشته عبارتند از: S ،Sa ،Sar ،Sara
  - **پسوند (Suffix)**: با حذف یک یا چند مورد از علائم از ابتدای رشته بدست می‌آید. مثال: Sara، ara، ra، a
  - **زیر رشته (Substring)**: شامل خود رشته S و هر پیشوند و پسوند ممکن از آن است.
- **زبان (Language)**: مجموعه‌ای از رشته‌های ورودی یک الفبا می‌باشد مثل زبان فارسی

عملیات روی زبان ها

فرض کنید که L و D دارای الفبایی به صورت $D = \{0,1,2, \dots ,9\}$ و $L = \{a, \dots ,z, A, \dots ,Z\}$ می باشند. آنگاه عملیات زیر را می‌توان انجام داد:

- $L \cup D = \{a, \dots ,z,A, \dots ,Z,0,1,2, \dots ,9\}$:$\boldsymbol{L \cup D}$ مجموعه‌ای است از حروف و ارقام به صورت
- $L \cdot D = \{a0,a1, \dots ,z0, \dots ,A0, \dots ,Z0, \dots ,Z9\}$:$\mathbf{L \cdot D}$ مجموعه‌ای است که رشته های آن شامل حروفی می باشند که با ارقام ختم می‌گردند.
- $D^2 = \{00,01,02, \dots ,10,11,12, \dots ,90,91,92, \dots ,99\}$:$\mathbf{D^2}$ مجموعه‌ای از هر ترکیب ممکن الفبای D که دارای طول ۲ باشد:
- $D^4 = \{0000,9999,1234,9876,8426,3671, \dots\}$:$\mathbf{D^4}$ مجموعه‌ای از هر ترکیب ممکن الفبای D که دارای طول ۴ باشد:
- $\boldsymbol{D^*}$: شامل مجموعه همه رشته های قابل تولید از الفبای D با هر طولی به همراه رشته با طول صفر (λ یا ε)

$$D^* = \{\lambda, 0, 00, 000, \dots , 1, 1^2, 1^3, \dots , 8^{657}, 6578952136875, \dots\}$$

- $\boldsymbol{D^+}$: شامل مجموعه همه رشته های قابل تولید از الفبای D با هر طولی بجز رشته با طول صفر (λ یا ε)
- با توجه به تعاریف D^* و D^+ خواهیم داشت: $D^* = D^+ + \{\lambda\}$ یا $D^+ = D^* - \{\lambda\}$

عبارات منظم

هر عبارت منظم روی الفبا به صورت زیر تعریف می شود (هر عبارت منظم، زبان منظم L را توصیف می کند):

۱. λ یک عبارت منظم است که زبان $\{\lambda\}$ را توصیف می‌کند.

۲. اگر a عضوی از Σ باشد a یک عبارت منظم است که زبان $\{a\}$ را توصیف می‌کند.

۳. اگر r و s عبارات منظمی باشند که زبان های L(r) و L(s) را تعریف کنند آنگاه:

- (r) | (s) عبارت منظمی است که زبان L(r)U L(s) را تعریف می‌کند.
- (r) . (s) عبارت منظمی است که زبان L (r) . L(s) را تعریف می‌کند.
- (r) * عبارت منظمی است که زبان $L (r)^*$ را تعریف می‌کند.
- دو عبارت منظم زمانی با هم معادل هستند که زبان یکسانی را تعریف کنند. مثال: $\{ a , b\}$ = (b|a) = (a|b)

گرامر

- شامل مجموعه‌ای از قوائد است که ساخت جملات یک زبان را مشخص می‌کند. در واقع گرامر به ما کمک می‌کند که تشخیص دهیم آیا جمله‌ای به یک زبان متعلق است یا خیر.

- **تعریف رسمی:** هر گرامر معرف یک زبان است که شامل چهارتایی $G=<N, T, S, P>$ می باشد که در آن N مجموعه‌ای از غیر پایانه‌ها (non - terminal) و T مجموعه‌ای از پایانه‌ها (terminal) و S علامت شروع گرامر ($S \in N$) و P مجموعه‌ای از قواعد است. مثال:

$N=\{E\}$ $T = \{ (,),*, + , id\}$ $S=\{E\}$

$P : \{E \rightarrow E+E, E \rightarrow E*E, E \rightarrow (E), E \rightarrow id\}$

- نکته : غیر پایانه‌ها (N) را با حروف بزرگ و پایانه‌ها (T) را با حروف کوچک نمایش می‌دهند.

انواع گرامر

چامسکی با در نظر گرفتن محدودیت‌هایی روی گرامرها، آنها را به چهار دسته تقسیم کرد:

۱. **گرامر نوع صفر یا بدون محدودیت:** این گرامر شامل قواعدی به صورت $\alpha \rightarrow \beta$ است که محدودیت خاصی ندارند. فقط سمت چپ این قواعد (α) باید حداقل شامل یک غیر پایانه (non- terminal) باشد.

۲. **گرامر نوع یک یا وابسته به متن:** علاوه بر محدودیت فوق باید طول رشته سمت چپ از طول رشته سمت راست کوچکتر و یا مساوی باشد. یعنی $|\alpha| < |\beta|$

مثال:

$S \rightarrow aAb$ √

$aA \rightarrow aaA$ √

$abaA \rightarrow aB$ × (طول رشته سمت راست کمتر است)

انواع گرامر (ادامه)

۳- گرامر نوع دو یا مستقل از متن: علاوه بر محدودیت‌های فوق (که در گرامر ۱ و ۲ گفته شد) باید سمت چپ قواعد تنها شامل یک غیرپایانی باشد یعنی $|\alpha| = 1$

۴- گرامر نوع سه یا منظم: علاوه بر تمام محدودیت‌هایی که در بالا گفته شد، شامل λ نباشد (بازگشتی چپ یا بازگشتی راست باشد).

$B \rightarrow \lambda$ ×

$A \rightarrow Aa$ √

$A \rightarrow aA$ √

- مثال : گرامری بنویسید که زبان $L(G)=\{a^n b^n , n \geq 0\}$ را تولید کند ؟ حل:

$S \rightarrow aSb \mid \lambda$

- مثال: گرامر زیر چه زبانی تولید می کند؟

$S \rightarrow abS \mid ab$

حل:

$S \Rightarrow ab$

$S \Rightarrow abS \Rightarrow abab$

$S \Rightarrow abS \Rightarrow ababS \Rightarrow abababS \Rightarrow abababab$

$L(G)=\{(ab)^n, n>0\}$

- مثال: گرامر زبان $L(G)=\{a^n b^{2n} , n \geq 0\}$ را بنویسید. حل:

$S \rightarrow aSbb \mid \lambda$

$S \Rightarrow aSbb \Rightarrow aaSbbbb \Rightarrow aaaSbbbbbb$ ...

اشتقاق

با شروع از علامت Start گرامر و با استفاده از قواعد موجود، می‌توان یک سری از جملات را تولید کرد که به آن بسط یا اشتقاق آن گرامر می‌گویند.
دو نوع بسط وجود دارد:

- **بسط چپ:** که در آن همواره سمت چپ‌ترین غیر پایانه را با قاعده معادل آن جایگزین می‌کنیم.
- **بسط راست:** که در آن همواره سمت راست‌ترین غیر پایانه را با قاعده معادل آن جایگزین می کنیم.

مثال : گرامر زیر را در نظر بگیرید، آن را بسط دهید تا به جمله id + id * id برسید.

E → E * E,
E → E + E,
E → id

اشتقاق راست:

$$E \rightarrow E + E \rightarrow E + E * E \rightarrow E + E * id \rightarrow E + id * id \rightarrow id + id * id$$

اشتقاق چپ:

$$E \rightarrow E + E \rightarrow id + E \rightarrow id + E * E \rightarrow id + id * E \rightarrow id + id * id$$

درخت تجزیه

درخت تجزیه نشان دهنده چگونگی اشتقاق رشته ای از زبان از نماد شروع گرامر

ساخت درخت تجزیه:

S ریشه درخت

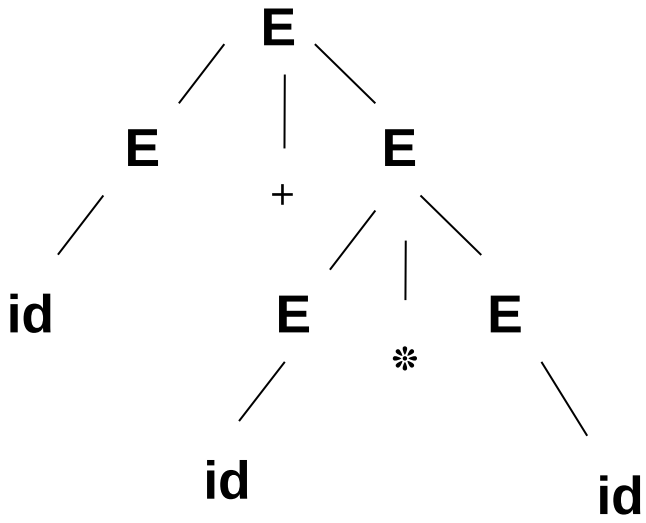
قانون $A \rightarrow XYZ$: A گره ای در درخت و XYZ فرزندان آن

قانون $A \rightarrow Xa$: A گره ای در درخت و X ,a فرزندان آن

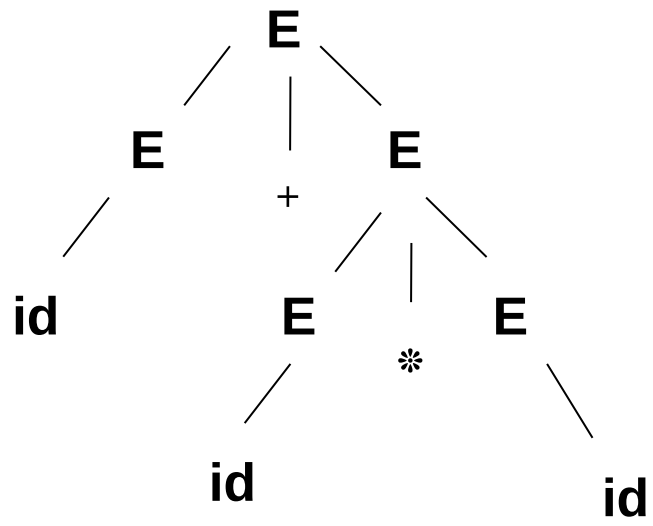
پایانه ها (حروف کوچک) تنها در برگ ها دیده می شوند.

درخت اشتقاق راست و چپ

درخت تجزیه ای نشان دهنده مراحل اشتقاق بکار رفته (راست یا چپ). مثال: **id+id*id**



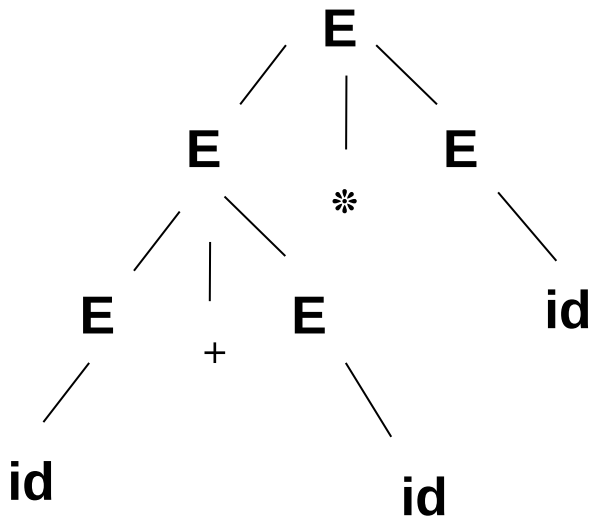
$E \rightarrow E + E \rightarrow E + E * E \rightarrow E + E * id$
$\rightarrow E + id * id \rightarrow id + id * id$

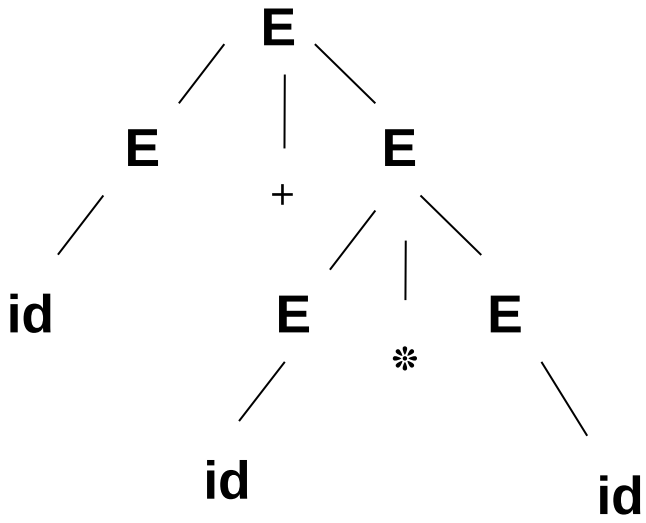


$E \rightarrow E + E \rightarrow id + E \rightarrow id + E * E$
$\rightarrow id + id * E \rightarrow id + id * id$

گرامر مبهم

وجود دو اشتقاق برای یک رشته در گرامر (وجود دو درخت اشتقاق متفاوت)





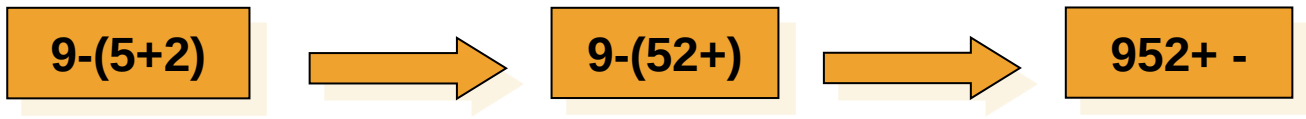
نشان گذاری پسوندی

نشان گذاری یک عبارت مانند E

۱- اگر E متغیر و یا ثابت باشد نشان گذاری آن خودش می شود.

۲- اگر E عبارتی به شکل E1 op E2 باشد که Op عملگر دودویی است نشان گذاری آن عبارتست از F1 F2 Op که F1, F2 نشان گذاری پسوندی E2 , E1 هستند.

۳- اگر E عبارتی بشکل (E1) باشد، نشان گذاری برای E1 همان نشان گذاری برای E می باشد.



تعریف نحو گرا

- کاربرد برای ترجمه ساختارهای زبان.
- ترجمه ساختار بر اساس صفات مربوط به مولفه های نحوی
- تعیین نوع ساختار، مکان اولین دستور تولید شده در برنامه هدف یا تعداد دستورات برای کامپایلر

تعریف نحوی جهت دار

گرامر و مجموعه ای از قواعد معنایی وابسته به آن

صفات و محاسبه آنها

۱. هر نماد در گرامر مجموعه ای از صفات دارد.

۲. در هر مولد یا قانون گرامر مجموعه ای از قواعد معنایی برای محاسبه مقادیر صفات نمادها وجود دارد.

ترجمه

ساختن درخت ترجمه:

الف) ساختن درخت تجزیه برای ورودی

ب) اگر گره **n** در درخت تجزیه با نماد **x** از گرامر متناظر باشد، **x.a** مقدار صفت **a** از نماد **x** در آن گره است.

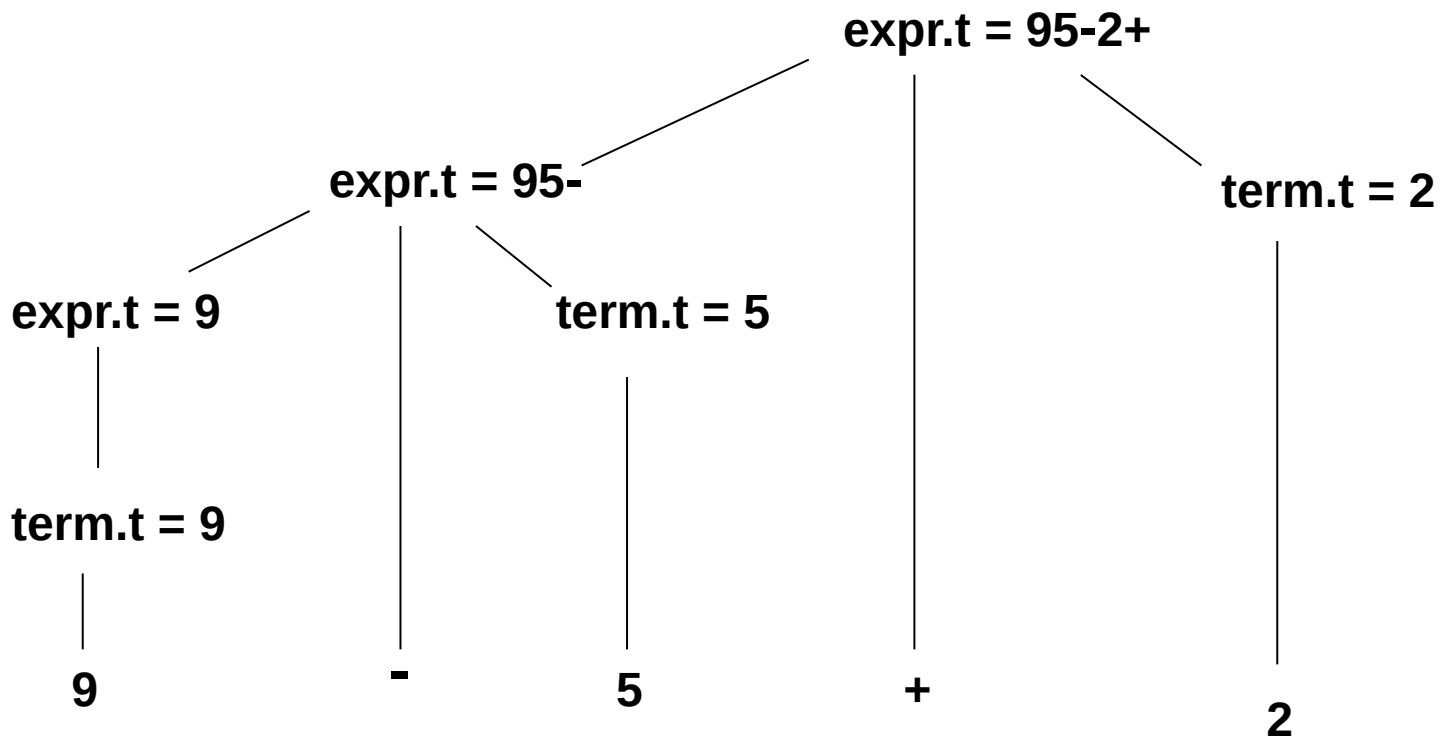
ج) مقدار **x.a** در گره **n** با استفاده از قواعد معنایی برای صفت **a** همراه با قانون گرامری استفاده شده درگره **n** محاسبه می شود.

مثال از ترجمه

قانون		قواعد معنایی
expr	expr1 + term	expr.t := expr1.t // term.t // ,+,
expr	expr1 - term	expr.t := expr1.t // term.t // , -
expr	term	expr.t := term.t
term	0	term.t := , 0,
term	1	term.t := , 1,
.....		.....
term	9	term.t := , 9,

تعریف نحوگرا برای ترجمه عبارات میانوندی به عبارت معادل پسوندی

درخت نحو



انواع درخت نحوی

درخت نحو مجرد

هر گره نماینده یک عملگر و فرزندان آن عملوند آن

درخت نحو واقعی

درخت تجزیه ای که عملگرها خود فرزند محسوب می شوند.

الگوی ترجمه

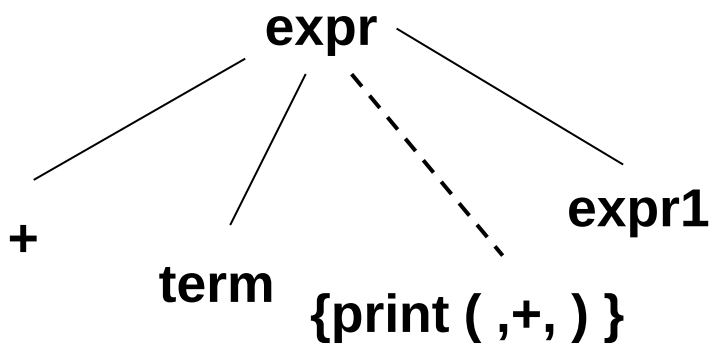
گرامر مستقل از متنی که قطعه برنامه هایی که عملیات معنایی نامیده می شوند در سمت راست قوانین آن اضافه شده اند.

تفاوت با ترجمه نحوگرا

نمایش ترتیب ارزشیابی قوانین بطور صریح.

درخت تولید شده برای الگوی ترجمه

طریقه ساخت درخت



یک برگ اضافی ساخته شده برای عمل معنی

۱- ساختن فرزند اضافی برای درخت

۲- متصل نمودن این فرزند به گره مربوط به قانون خود در گرامر

تجزیه (پارسینگ)

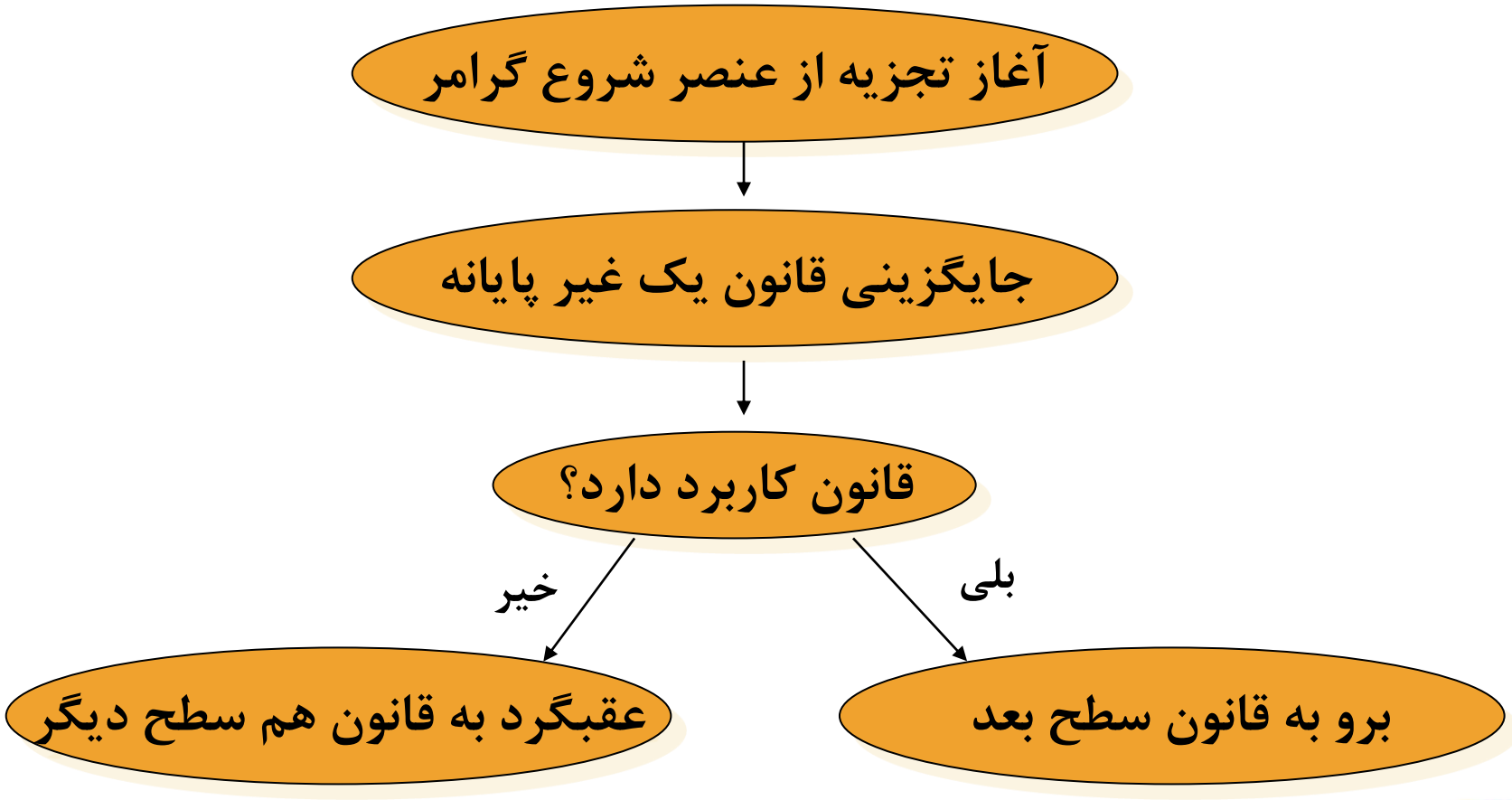
- تجزیه به کمک تحلیلگر نحوی و به نام تحلیل نحوی انجام می گیرد.
- در تجزیه تعلق رشته ورودی به زبان مبدا بررسی می شود.
- بررسی طبق ساختار و نحو دستورات زبان مبدا انجام می گیرد.

تجزیه- دسته بندی روشها

روش بالا به پایین :
ساخته شدن درخت تجزیه از
بالا به پایین. مانند LL(1)

روش پایین به بالا:
ساخته شدن درخت تجزیه از پایین
مانند تجزیه رو به بالا
عملگراولویت و
LR

تجزیه کننده بالا به پایین

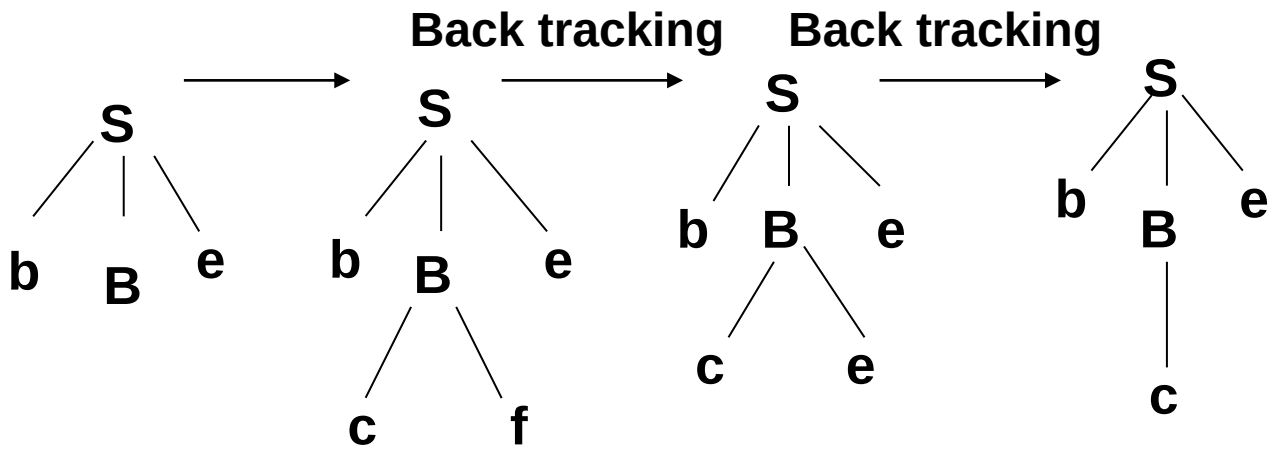


مثال از تجزیه بالا به پایین

$S \rightarrow bBe$

$B \rightarrow cf/\ ce/c$

تجزیه برای تولید رشته bce:



تجزیه بالا به پایین پیشگویانه

روش تجزیه بالا به پایین بدون ویژگی عقبگرد

با نگاه به نماد پیش نگر در مورد استفاده از هر قانون درتصمیم گیری

نماد پیش نگر

مجموعه تمام پایانه هایی است که در قوانین مربوط به غیر پایانه در سمت چپ قرار می گیرند

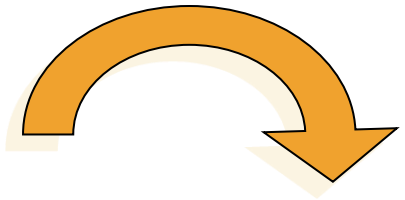
مثال

نمادهای پیش نگر یا مجموعه First

$A \rightarrow aB / cC / e$

$B \rightarrow dA / c$

$C \rightarrow cB / a$



First (A):{ a, c, e }

First (B): { d, c }

First (C) : {c, a }

بازگشتی چپ

ظاهر شدن غیر پایانه سمت چپ در سمت راست قانون به عنوان اولین عنصر

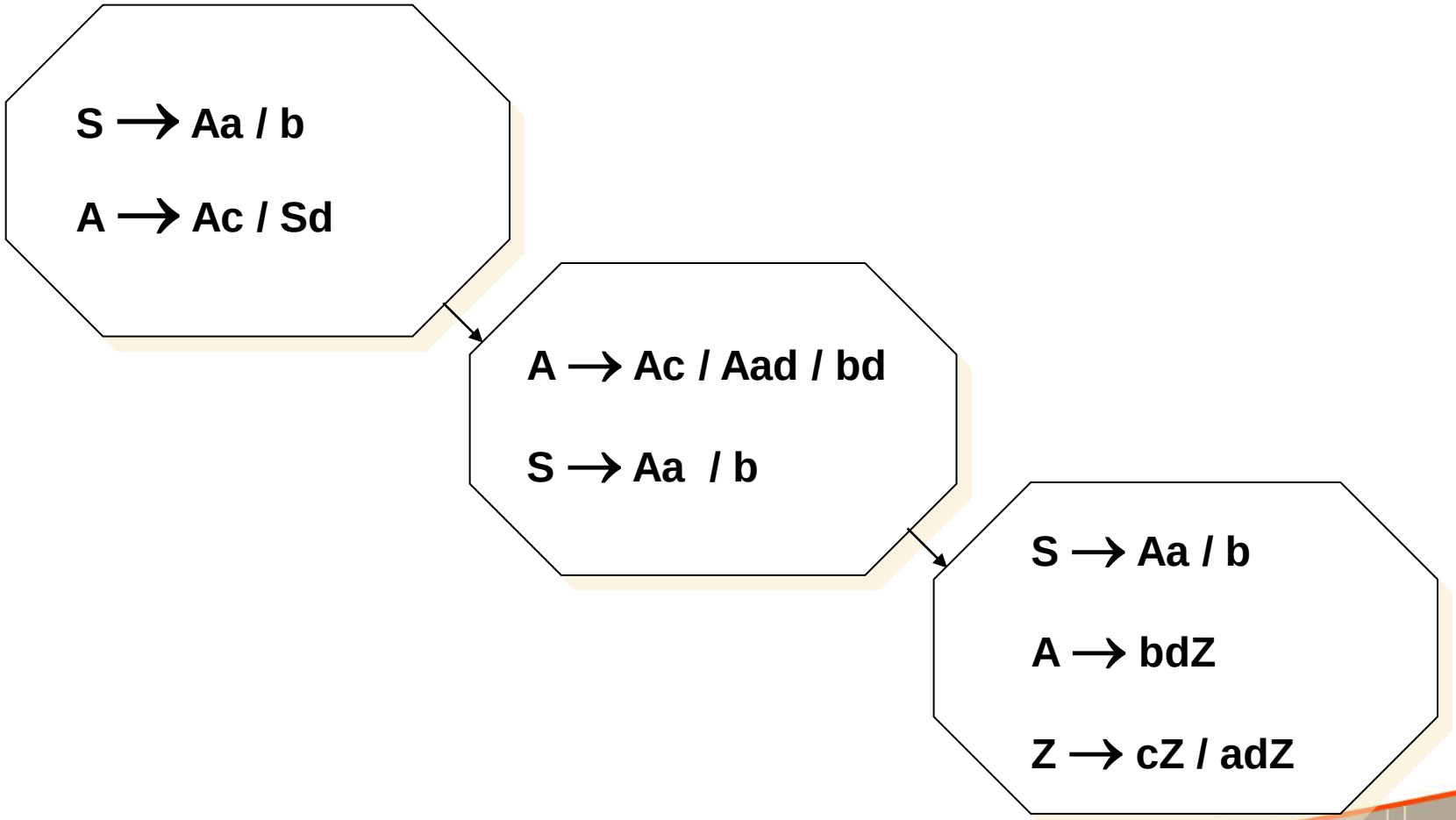
حذف بازگشتی چپ پیش از تجزیه

$A \rightarrow Aa / Ab / Ac / x / y$ ⟹

$A \rightarrow xZ / yZ$

$Z \rightarrow aZ / bZ / cZ$

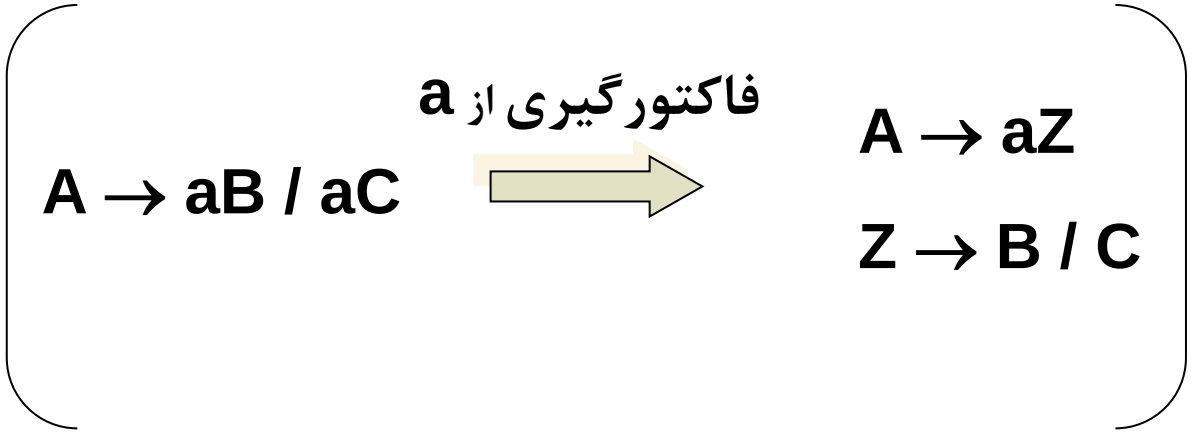
مثال حذف بازگشتی چپ



فاکتور چپ

یکسان بودن عنصر سمت چپ در حداقل دو قانون گرامر

فاکتورگیری چپ



مثال فاکتورگیری چپ

$S \rightarrow iEtS$

$S \rightarrow iEtSeS \mid a$

$E \rightarrow b$

$S \rightarrow iEtS \mid iEtSes$

$S \rightarrow a$

$E \rightarrow b$

$S \rightarrow iEtSZ \mid a$

$Z \rightarrow \lambda \mid eS$

$E \rightarrow b$

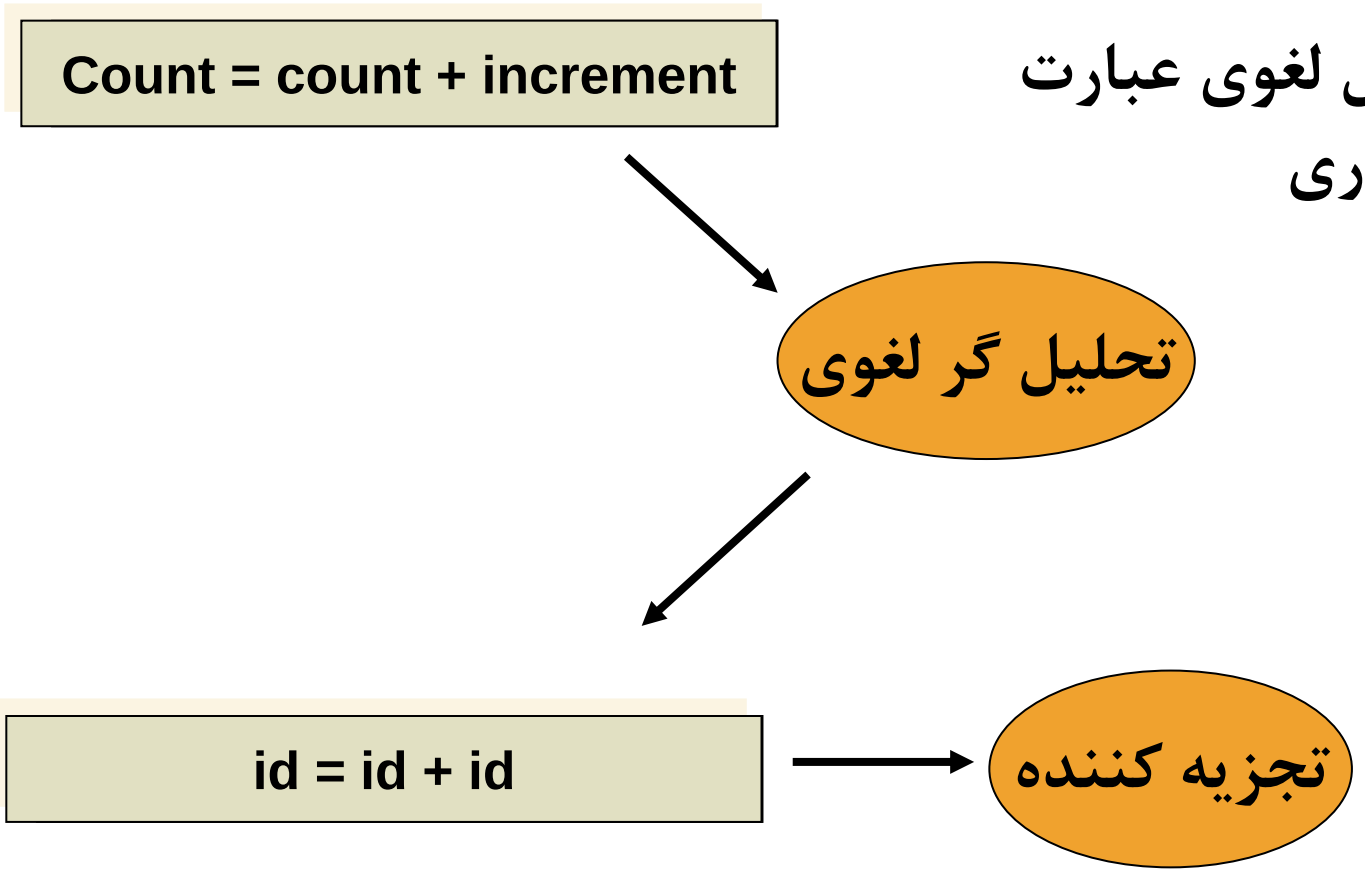
تحلیل لغوی

- دریافت یک رشته کاراکتری از ورودی
- استخراج نشانه ها از آن
- تحویل نشانه ها به تجزیه کننده
- ارتباط دادن پیام های خطای تولید شده کامپایلر با برنامه مبدا

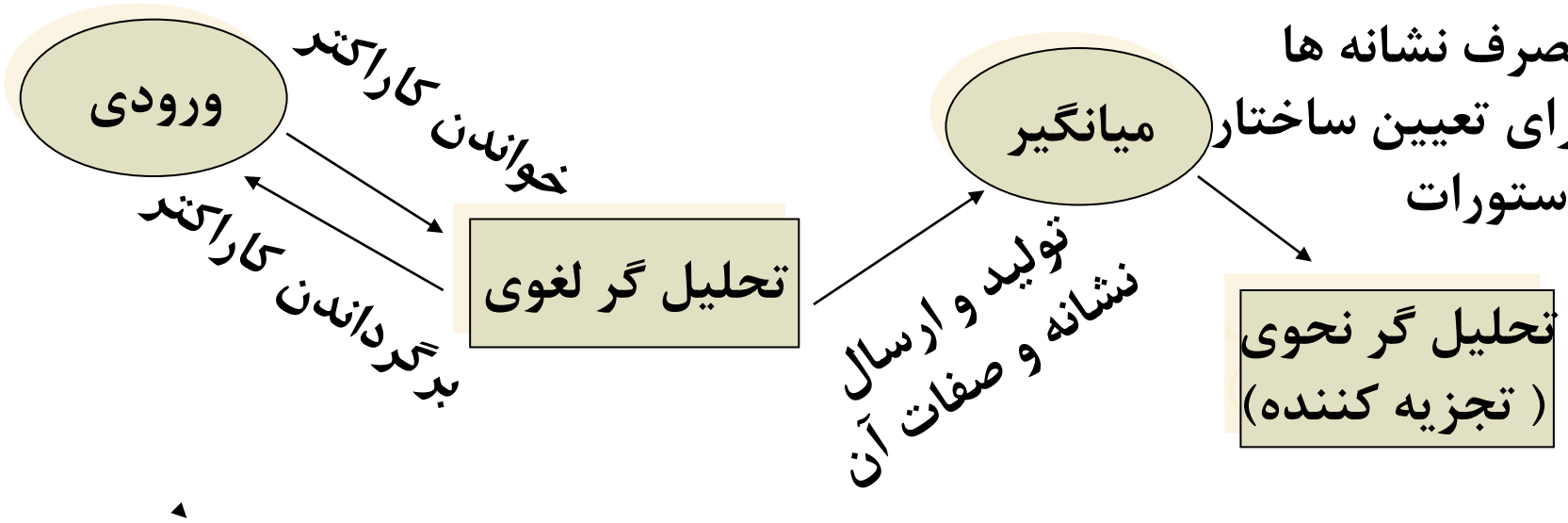
- حذف فضاهای خالی بین عبارات و توضیحات برنامه
- تشخیص شناسه ها و کلمات کلیدی زبان

مثال

تحلیل لغوی عبارت دستوری



رابط تحلیلگر لغوی



میانگیر

۱- در هر زمان حاوی یک بلوک از کاراکترها

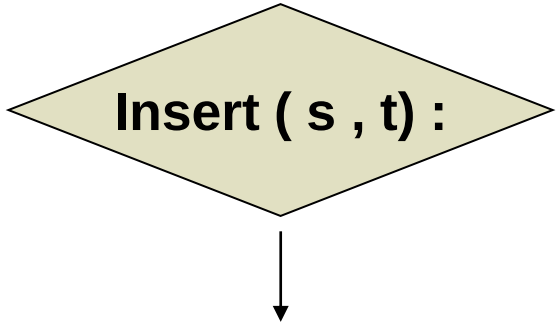
۲- اشاره گری به مکان نشانه بعدی پردازش نشده

تشکیل جدول نماد

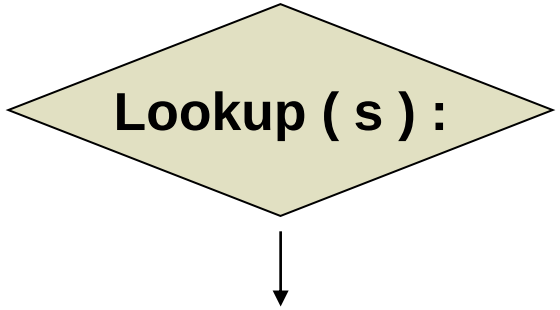
جدولی ساخته شونده توسط فازهای تحلیل، مورد استفاده فازهای تولید کد

- **فاز تحلیل لغوی** ← ذخیره رشته کاراکتری تشکیل دهنده شناسه در جدول.
- **فاز تحلیل نحوی** ← اضافه کردن نوع شناسه، مورد استفاده(رویه، متغیر و..)
- **فاز تحلیل معنایی** ← درج مکان شناسه در حافظه در جدول
- **فاز تولید کد** ← استفاده از اطلاعات جدول برای دسترسی به متغیر و تولید کد

جدول نماد- روال ها



اندیس وارده جدید مربوط به رشته s نشانه t را برمی گرداند.

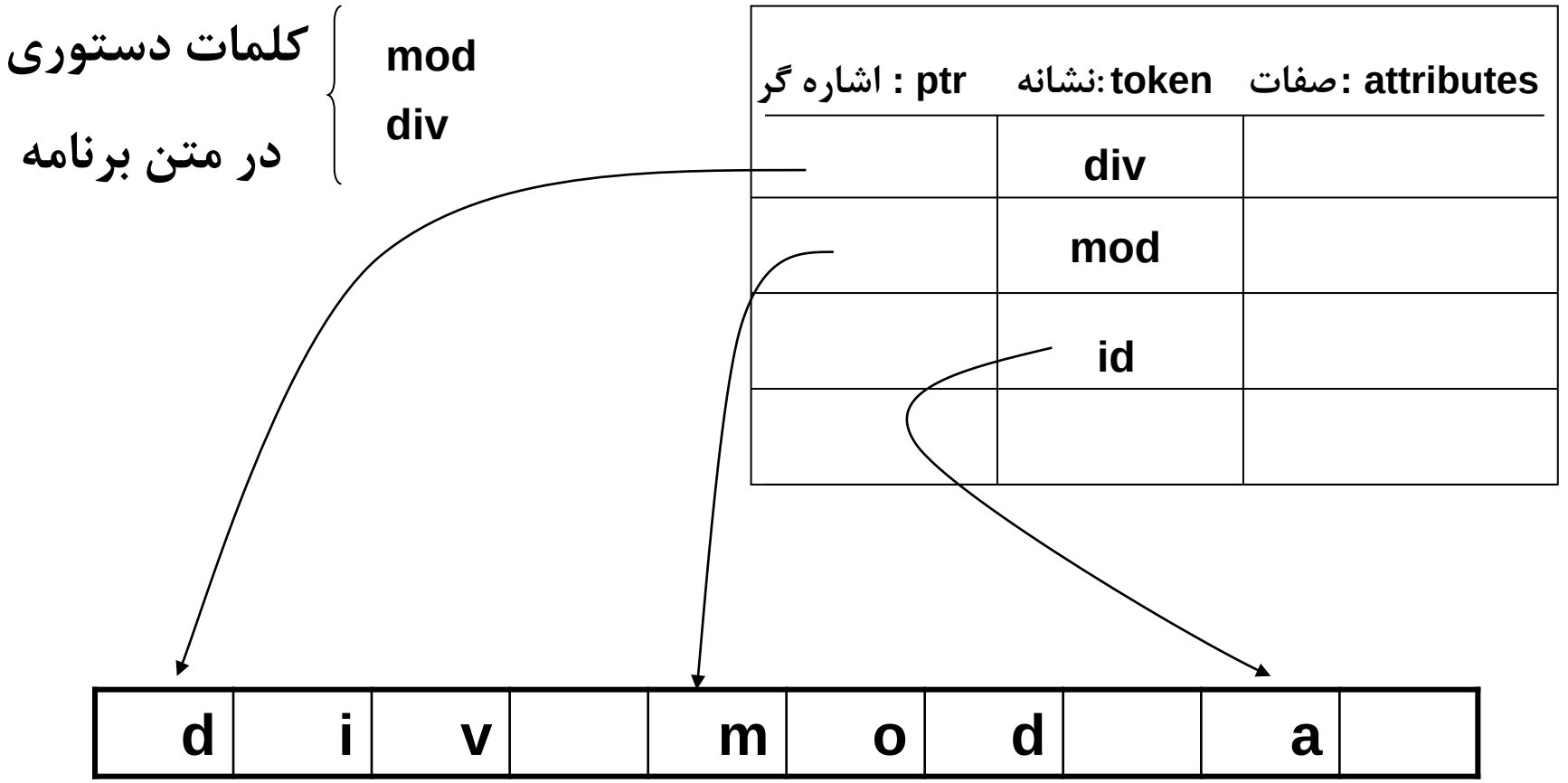


اگر s در جدول است، اندیس آن برمی گردد اگر نه، صفر برمی گردد

عمل **Lookup** تعیین وجود شناسه در جدول نماد

عمل **Insert** در صورت عدم وجود شناسه، درج آن در جدول

جدول نماد- پیاده سازی



ماشین پشته انتزاعی

ماشین پشته انتزاعی: شکل مرسوم نمایش میانی تولید کد

اجزای ماشین

۱- حافظه دستورات

۲- حافظه داده ها

محاسبات ماشین

۱- محاسبات صحیح

۲- دستورات دستکاری پشته

۳- روند کنترل

دستورات محاسباتی

- قابلیت پیاده سازی مستقیم عملیات پایه مانند جمع، تفریق و ... در ماشین انتزاعی
- پیاده سازی عملیات پیچیده تر با دنباله ای از دستورات اولیه ماشین
- استفاده از نمایش پسوندی در کد ماشین برای ارزیابی یک عبارت محسباتی
- استفاده از پشته در حین ارزیابی عبارات

ارزشیابی عبارت در ماشین

انجام عملگر بر روی مقدار پشته

بیرون پراندن عملوندها از پشته

قرار دادن نتیجه بر روی پشته

مثال ارزیابی عبارت محاسباتی با پشته

عبارت محاسباتی 13 + 5*

۱- عدد ۱ را روی پشته قرار ده

۲- عدد ۲ را روی پشته قرار ده

۳- دوتا از بالاترین عناصر پشته را با هم جمع و آن دو را از پشته بیرون ده

۴- نتیجه یعنی عدد ۴ را روی پشته قرار ده

۵- عدد ۵ را روی پشته قرار ده

۶- دوتا بالاترین عناصر پشته را در هم ضرب و آنها را بیرون ده

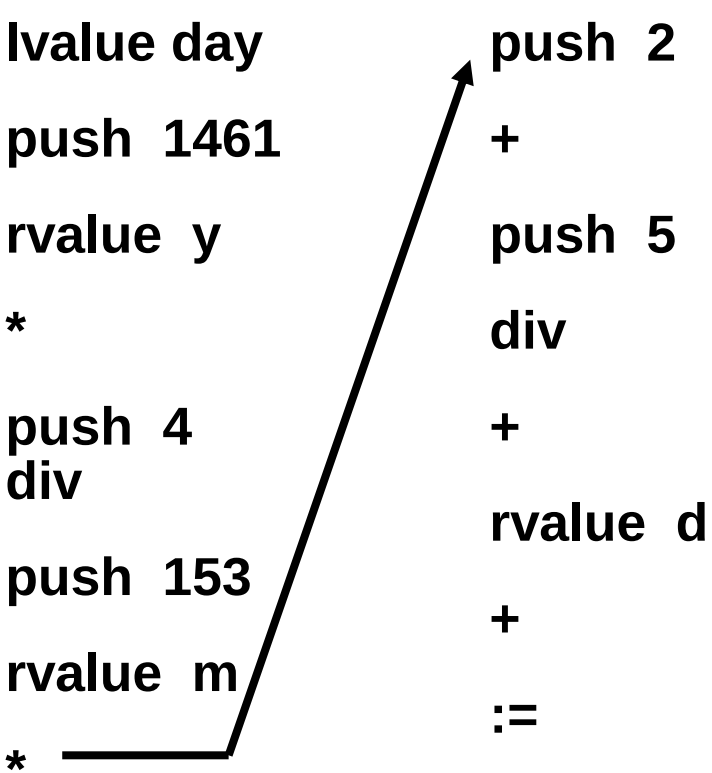
۷- نتیجه یعنی عدد ۲۰ را روی پشته قرار ده

دستکاری پشته

- S را روی پشته قرار ده — Push s
- محتویات مکان L را روی پشته قرار ده — rvalue l
- آدرس L را روی پشته قرار ده — lvalue l
- مقدار در بالای پشته را دور بریز — pop
- r-value بروی پشته در l-value زیر آن گذاشته و هر دو از پشته خارج — :=
- یک نسخه از مقدار بالای پشته را بر روی پشته فشار بده — copy

مثال عملیات در پشته هنگام محاسبه

day := (1461*y) div 4 + (153*m + 2) div 5 + d ترجمه



کنترل جریان در ماشین

تعیین مکان پرش

۱- تعیین محل پرش با عملوند دستور

۲- تعیین فاصله نسبی برای پرش مثبت یا منفی با عملوند دستور

۳- تعیین محل پرش با نماد های دستور

امکان برداشتن عملوند از بالای پشته

کنترل جریان - دستورات

lable l	عدم تاثیر در مقصد پرش ها به L
goto l	اجرای دستور بعدی از حکمی با برچسب L
gofalse l	خارج نمودن مقدار بالای پشته، پرش در صورت صفر بودن
gotrue l	خارج نمودن مقدار بالای پشته ، پرش در صورت صفر نبودن
halt	توقف اجرا

پایان فصل ۲

خواص عمومی زبان ها و انواع گرامرها

(یاد آوری نظریه زبان ها)