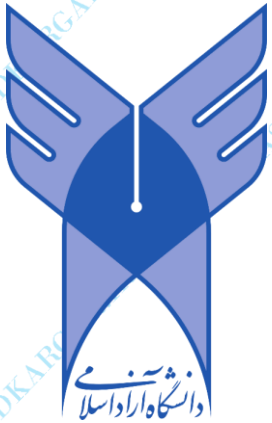


دانشگاه آزاد اسلامی واحد تبریز

ساختمان داده ها

ارایه: دکتر مسعود کارگر

برگرفته از مجموعه جزوات پارسه (پشته)

هر لیستی از داده‌ها که عمل حذف (pop) و درج (push) در آن از یک طرف (بالای پشته) به کمک اشاره گری به نام "top" انجام می‌گیرد.

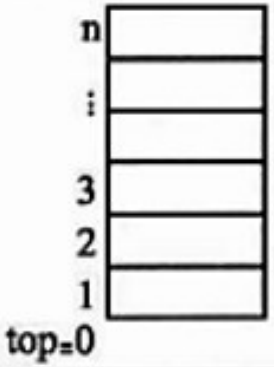
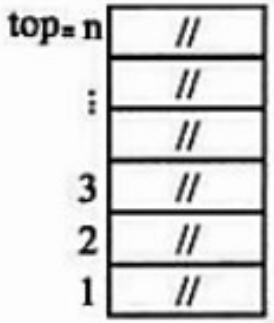
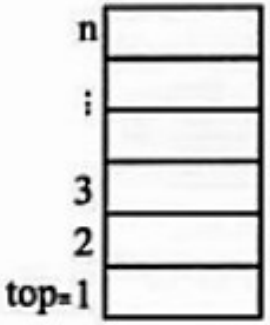
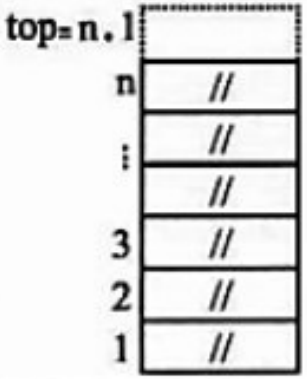
پیاده سازی:

برای ذخیره سازی ساختار یک پشته می توان یکی از ۲ ساختار زیر را در نظر گرفت:

۱ـ آرایه (Array) : که ساختاری ایستا (static) و محدود دارد.

۲ـ لیست پیوندی (Linked List) : که ساختاری پویا (dynamic) و متناسب با داده ها دارد.

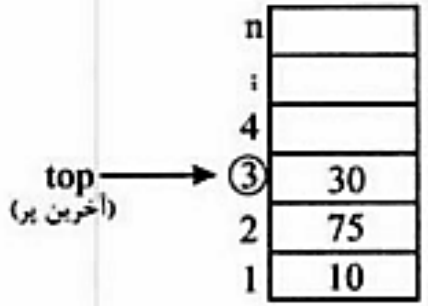
در صورتی‌که از آرایه stack [1 .. n] برای نگهداری عناصر پشته استفاده شود، شرایط مرزی به صورت زیر خواهد بود:

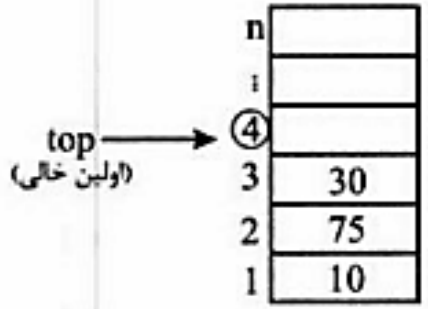
وضعیت top	پشته خالی (empty)	پشته پر (Full)
آخرین خانه پر(پیش فرض)		
اولین خانه خالی		

نکته مهم : همیشه به طور پیش فرض top به آخرین خانه پر اشاره می‌کند.

الگوریتم‌های pop (حذف) و push (درج)

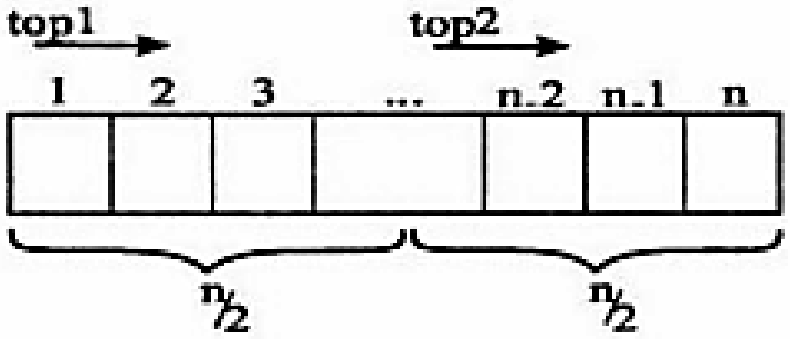
الگوریتم(حذف)	الگوریتم(درج)	وضعیت top

procedure pop (var item : data type);
begin
if top = 0 then
write ('stack is Empty')
else
begin
item : = stack [top];
top : = top - 1;
end;
	procedure push (var item : data type);
	begin
	if top = n then
	write ('stack is Full')
	else
	begin
	top : = top + 1;
	stack [top] : = item;
	end;
		

procedure pop (var item : data type);
begin
if top = 1 then
write ('stack is Empty')
else
begin
top : = top - 1;
item : = stack [top];
end;
	procedure push (var item : data type);
	begin
	if top = n+1 then
	write ('stack is Full')
	else
	begin
	stack [top] : = item;
	top : = top + 1;
	end;
		

بهینه سازی قرار گرفتن دو stackدر یک آرایه:

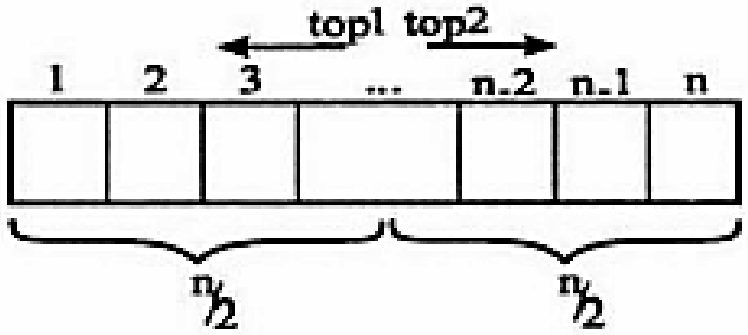
در صورتی‌که بخواهیم دو Stackرا در یک آرایه قرار دهیم، وضعیت‌های مختلفی برای مدیریت پشته ها به شرح زیر وجود دارد :



الف)

top1 : از ابتدا تا وسط حرکت می کند.

top2 : از وسط تا انتها حرکت می کند.



ب)

top1 : از وسط تا ابتدا حرکت می کند.

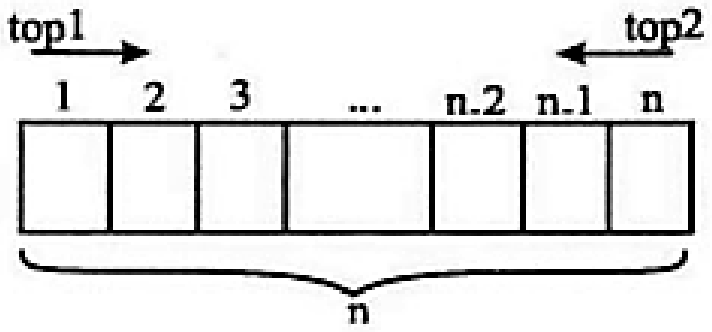
top2 : از وسط تا انتها حرکت می کند.

ج)

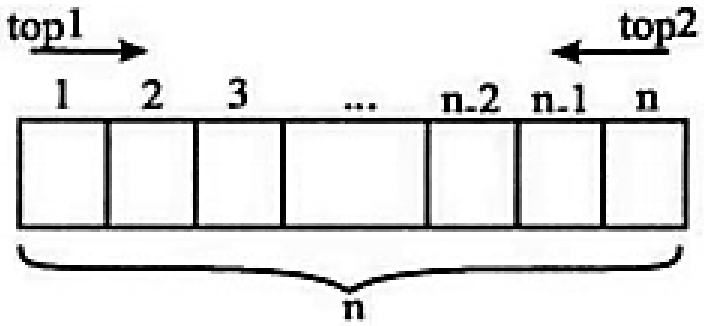
top1: از ابتدا تا **top2** حرکت می کند.

top2: از انتها تا **top1** حرکت می کند.

(بهترین وضعیت)



در بهترین وضعیت **شرایط پرشدن پشته ها** به شرح زیر خواهد بود:



وضعیت top1,top2	شرط پربودن پشته ها
top1,top2 به آخرین خانه پر اشاره کنند	top1 = top2 - 1 یا top2 = top1 + 1
top1,top2 یکی به آخرین خانه پر ویکی به اولین خانه خالی	top1 = top2
top1,top2 به اولین خانه خالی اشاره کنند	top1 = top2 + 1 یا top2 = top1 - 1

خروجی‌های مجاز برای ورودی‌های پشته:

در یک پشته nتایی در صورتی‌که داده‌های $a_1, a_2, \ldots, a_n$ به ترتیب وارد Stack شوند، با رعایت قواعد زیر می‌توانیم خروجی‌های مجاز را تعیین کنیم:

الف) هر داده به محض ورود می تواند از پشته خارج شود.

ب) در هر مرحله هرگاه بخواهیم داده‌ای را در خروجی ببینیم که هنوز وارد Stackنشده است می‌بایست داده‌ها را پشت سرهم وارد پشته کنیم تا به داده مورد نظر برسیم سپس داده را وارد پشته کرده و سپس خارج می‌کنیم.

ج) در هر مرحله هرگاه بخواهیم داده‌ای را در خروجی ببینیم که پایین Stack است و عنصر بالای پشته نیست *این عمل غیرمجاز است.*

مثال : یک پشته خالی با اعداد از ۱ تا ۶ در ورودی داده شده است. اعمال زیر بر روی پشته قابل انجام هستند:

PUSH :: کوچکترین عدد ورودی را برداشته و وارد پشته می کنیم.

POP :: عنصر بالای پشته را در خروجی نوشته و سپس آن را حذف می‌کنیم.

مثال : کدام‌یک از گزینه‌های زیر را **نمی‌توان** با هیچ ترتیبی از ورودی فوق به دست آورد؟ (اعداد را از چپ به راست بخوانید.)

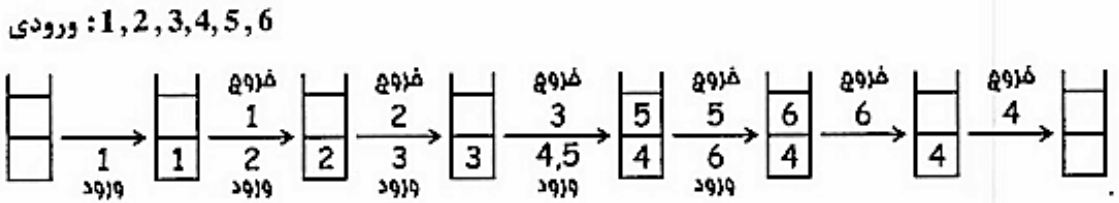
$\overline{1,2,3,4,5,6}$

(کارشناسی ارشد دولتی ۷۴)

۱) 1,2,3,5,6,4 ۲) 3,2,4,6,5,1 ۳) 4,3,2,1,6,5 ۴) 2,1,5,3,6,4

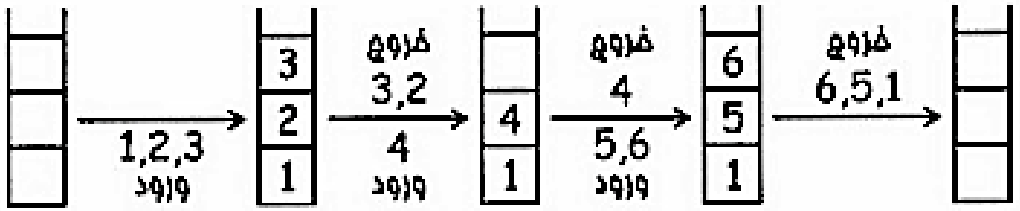
حل: گزینه ۴ درست است.

گزینه ۱ مجاز است زیرا:



خروجی: 1,2,3,5,6,4

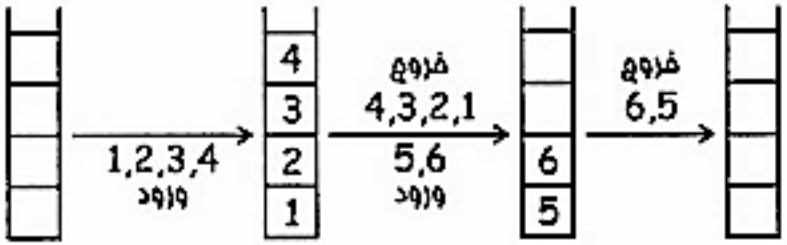
گزینه ۲ مجاز است. زیرا :



خروج: 3,2,4,6,5,1

گزینه ۳ مجاز است. زیرا

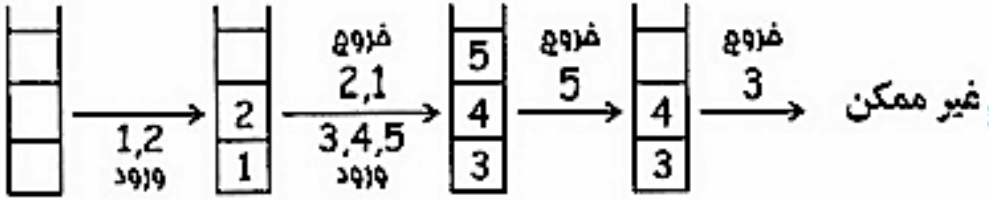
ورودی : 1,2,3,4,5,6



خروج : 4,3,2,1,6,5

گزینه ۴ امکان‌پذیر نیست زیرا :

ورودی : 1,2,3,4,5,6



خروج : 2,1,5,3,6,4

تعداد خروجی مجاز

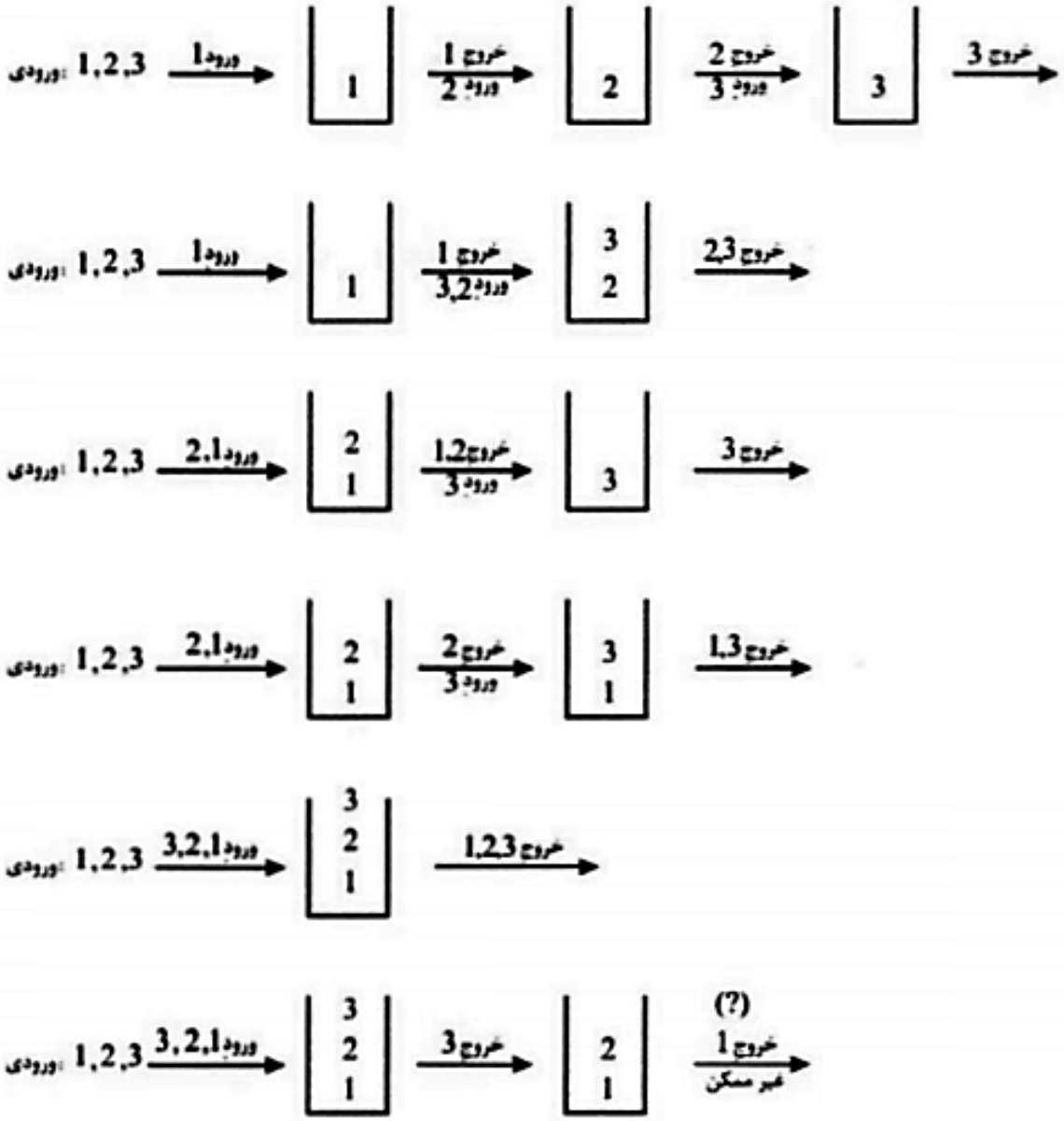
تعداد خروجی‌های مجاز در یک پشتهٔ n تایی همان عدد کاتالان بوده و بصورت زیر بدست می‌آید:

$$\frac{\binom{2n}{n}}{n+1}$$

مثال : در صورتی‌که $\overline{1,2,3}$ به ترتیب وارد یک پشته شوند تعداد حالت‌های مجاز کدام هستند.

$$n = 3 \rightarrow \frac{\binom{6}{3}}{3+1} = 5$$ تعداد حالت‌های مجاز

خروجی‌های مجاز و غیرمجاز به شرح زیرمی باشد:



پشته‌های مجاز

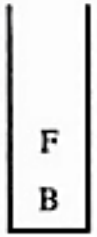
یک پشته از ورودی داده‌ها زمانی مجاز است که داده‌های قبلی روی داده‌های بعدی قرار نگیرند، چرا که داده‌های قبلی یا باید قبلاً وارد پشته شده و از آن خارج شده باشند یا این‌که خارج نشده باشند و پایین داده‌های بعدی باشند.

مثال : ورودی $\overline{A,B,C,D,E,F}$ به یک پشته را در نظر بگیرید، کدام یک از ترتیب های زیر در پشته مجاز (ممکن) است؟

ابتدا A , B وارد شده سپس B خارج شده C , D وارد شده سپس D خارج شده E وارد می‌شود.

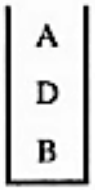
(مجاز)

ابتدا A وارد سپس خارج شده سپس B, C, D , E وارد شده سپس E , D , C خارج شده F وارد می‌شود.



(مجاز)

A یا باید پایین B قرار گیرد یا قبلاً از پشته حذف شده باشد.



(غیرمجاز)

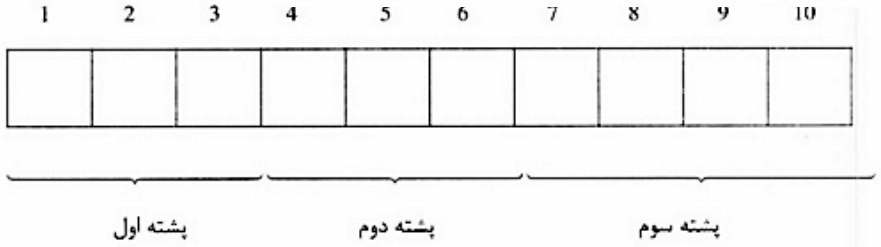
پشته چندگانه (Multiple Stack)

برای نمایش **n پشته** در یک آرایه **m تایی** مانند **stack [1 .. m]**، آرایه مورد نظر را به **n قسمت مساوی** تقسیم کرده و به هر پشته خانه‌های مشخص تخصیص داده می‌شود.

نکته: سعی داریم که فضای m خانه آرایه را به طور مساوی بین n پشته تقسیم کنیم در این حالت اگر n مضربی از m نباشد، فضای اضافی از آرایه m تایی به پشته آخر می‌رسد.

مثال : 3 پشته در آرایه 10تایی:

در این حالت برای هر پشته $\left\lfloor \frac{10}{3} \right\rfloor = 3$ خانه در نظر گرفته می‌شود اما یک خانه اضافی به پشته آخر می‌رسد.



پیاده سازی

برای پشته i ام از دو اشاره گر b[i](اشاره به ابتدای پشته) و t [i] (اشاره به انتهای پشته) استفاده t[i] به اولین خانه خالی اشاره می کند.

شرایط مرزی

پشته i ام پر (Full)	پشته i ام خالی (Empty)
t[i] = b [i + 1]	b [i] = t [i] = (i – 1) [$\frac{m}{n}$] + 1 i = 1,2,…,n

نکته:

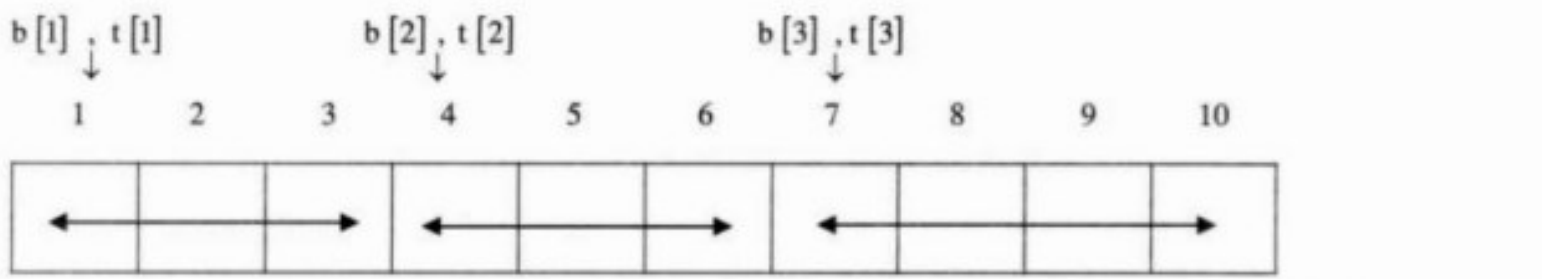
شرایط پر بودن پشته nام : t [i] = m + 1

مثال : می‌خواهیم در یک آرایه 10 تائی 3 پشته را قرار دهیم مطلوبست شرایط اولیه برای هر پشته؟

$$\left.\begin{matrix} m = 10 \\ n = 3 \end{matrix}\right\} \longrightarrow \begin{cases} i = 1 \rightarrow b[1] = t[1] = (1-1)\left\lfloor \frac{10}{3} \right\rfloor + 1 = 1 \\ i = 2 \rightarrow b[2] = t[2] = (2-1)\left\lfloor \frac{10}{3} \right\rfloor + 1 = 4 \\ i = 3 \rightarrow b[3] = t[3] = (3-1)\left\lfloor \frac{10}{3} \right\rfloor + 1 = 7 \end{cases}$$

$$\left\lfloor \frac{m}{n} \right\rfloor = \left\lfloor \frac{10}{3} \right\rfloor = 3$$

تعداد خانه‌های هر پشته



دیده می‌شود که یک خانه اضافی در انتهای آرایه به stack آخر (سوم) تعلق گرفته است.

عملیات در پشته چندگانه

push (درج)

```
procedure   push (i : Stack Number, item : data type):
begin
if   t [i] = b [i + 1]   OR   t[i] = m + 1   then
      write ('Full')
else      begin
            stack [t [i]] : = item;
            t [i]: = t [i] + 1;
          end;
      end;
```

چون در ابتدا t [i] به خانه خالی اشاره می‌کند برای درج (push)، ابتدا عمل درج انجام می‌شود سپس t [i] یک واحد به بالا حرکت می‌کند.

pop (حذف)

```
procedure   pop (i : Stack Number : var  item : data type);
    begin
    if   b [i] = t [i]   then
        write ('Empty')
    else
        begin
            t [i] : = t [i] - 1;
            item : = stack [t [i]];
        end;
```

چون در ابتدا t [i] به خانه خالی اشاره می‌کند برای حذف (pop) ابتدا t [i] یک واحد به پایین حرکت می‌کند تا به خانه پر برسد سپس داده مورد نظر حذف شده و در item قرار می‌گیرد.

بعضی کاربردهای پشته‌ها

۱ـ استفاده در توابع بازگشتی (برج هانوی، تابع Ackerman ، دنباله Fibonachi ، ....)

۲ـ ارزیابی عبارت‌های محاسباتی (prefix – postfix – infix)

۳ـ الگوریتم مسیر حرکت Maze

۴ـ پیمایش عمقی درخت‌ها و گراف‌ها

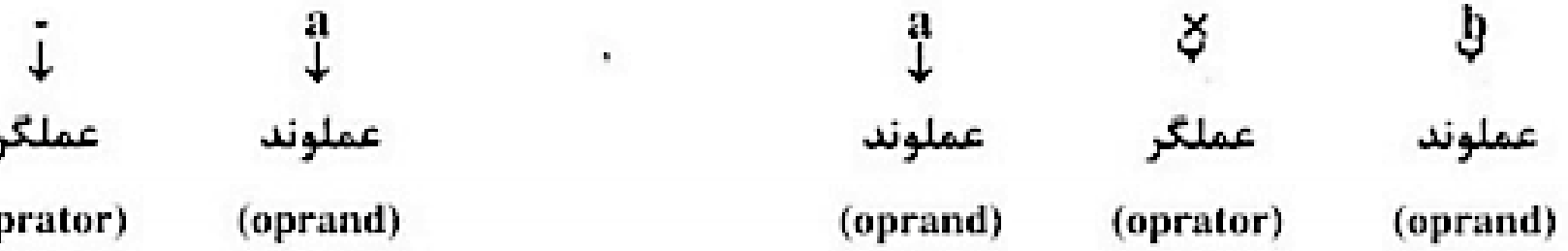
۵ـ استفاده در روش‌های مرتب‌سازی quicksort و mergesort

ارزشیابی عبارت‌های محاسباتی

هر عبارت محاسباتی با توجه به اولویت عملگرهایش قابل ارزیابی و محاسبه است.

یادآوری ۱:

در هر عبارت محاسباتی:



یادآوری ۲:

با توجه به یادآوری ۱ دیده می‌شود که در عبارت‌های محاسباتی دو نوع عملگر وجود دارد:

۱ـ یکانی (unary) **۲ـ باینری (binary)**

عملگرهای یکانی مانند: ـ (منفی) و + (مثبت) فقط به یک عملوند نیاز دارند.

عملگرهای دودویی مانند : × (ضرب)، / (تقسیم)، + (جمع) و − (تفریق) و ↑ (توان) به دو عملوند نیاز دارند.

اولویت عملگرها

به طور کلی صرف‌نظر از هر زبان برنامه‌سازی اولویت عمومی عملگرها را می‌توان به شرح زیر در نظر گرفت:

اولویت	نام عملگر	توضیحات
۱	( )	
۲	- (منفی)، + (مثبت)	
۳	↑ یا ^ (توان)	اولویت توان‌های متوالی از راست به چپ
۴	* (ضرب)، / (تقسیم)	هم اولویت، اولویت از چپ به راست
۵	+ (جمع)، ـ (تفریق)	هم اولویت، اولویت از چپ به راست.

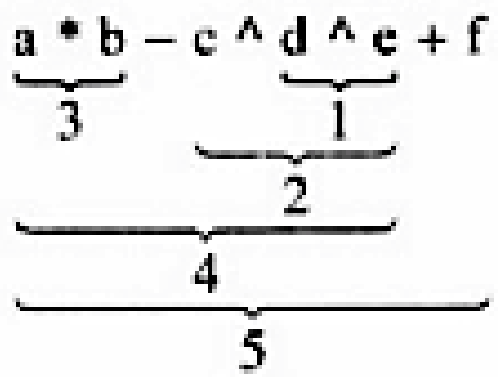
دقت کنید:

۱ـ در هر عبارت محاسباتی اگر **تعداد عملوندها 1 واحد بیشتر از تعداد عملگرها** باشد، در آن عبارت **همه عملگرها دودویی** هستند.

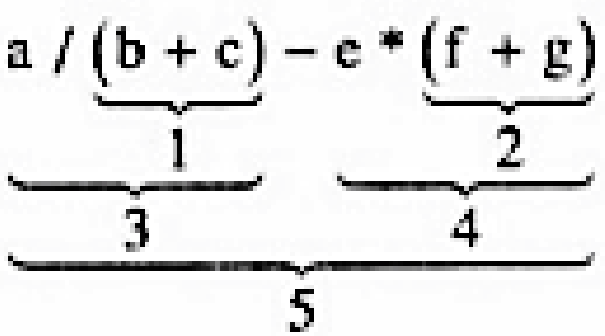
۲ـ در هر عبارت محاسباتی اگر **تعداد عملگرها بیشتر یا مساوی تعداد عملوندها** باشد، در آن عبارت **حتماً عملگر یکانی** وجود دارد.

مثال : اولویت‌بندی عبارت‌های زیر را در نظر بگیرید:

عبارت ۱:



عبارت ۲:



تعداد عملگر یکانی (unary)

در هر عبارت محاسباتی اگر **تعداد عملگرها بیشتر یا مساوی تعداد عملوندها** باشد:

1 + تعداد عملوندها ـ تعداد عملگرها = **تعداد عملگرهای یکانی**

درخت عبارت محاسباتی (درخت پارس)

برای تشکیل درخت هر عبارت محاسباتی به صورت زیر عمل می‌کنیم:

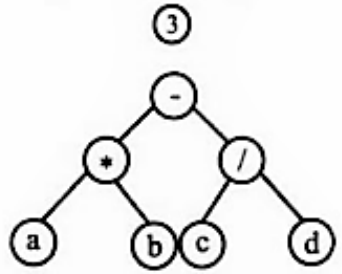
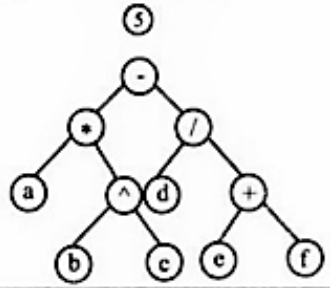
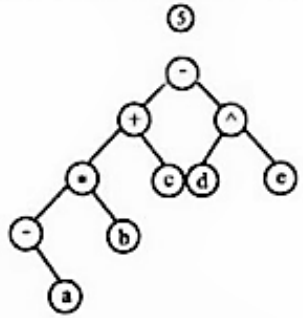
۱ـ ابتدا عبارت را برحسب اولویت عملگرهایش **اولویت‌بندی(شماره گذاری)** می‌کنیم.

۲ـ ریشه درخت، عملگر با **کمترین اولویت (بیشترین شماره)**است.

۳ـ ریشه زیر درختان چپ و راست نیز به همین ترتیب عملگر با **کمترین اولویت (بیشترین شماره)** در چپ و راست ریشه درخت خواهد بود.

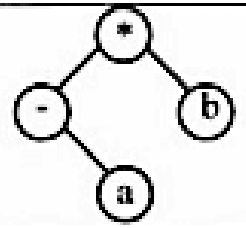
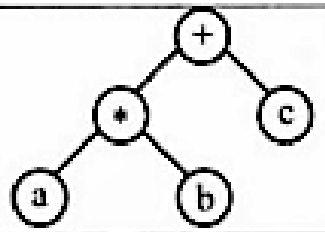
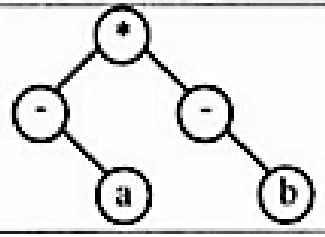
۴ـ **برگ‌های** درخت **عملوند** و سایر گره ها **(افرزندی یا ۲ فرزندی)، عملگر** خواهند بود.

به مثال‌های زیر دقت کنید:

درخت عبارت	اولویت بندی عبارت	عبارت محاسباتی
 کم اولویت‌ترین عملگر ـ (تفریق)بوده که در ریشه قرار گرفته است.	$\underbrace{\underbrace{a*b}_{1}-\underbrace{c/d}_{2}}_{3}$	a * b - c / d
	$\underbrace{\underbrace{a*\underbrace{b\,\hat{}\,c}_{2}}_{3}-\underbrace{d/\underbrace{(e+f)}_{1}}_{4}}_{5}$	a * b ^ c - d / (e + f)
 دیده می‌شود که عملوند مربوط به عملگر یکانی (ـ)، یعنی a سمت راست آن قرار گرفته است.	$\underbrace{\underbrace{\underbrace{-a}_{1}*b}_{3}+c-\underbrace{d\,\hat{}\,e}_{2}}_{4}}_{5}$	- a * b + c - d ^ e

رابطه بین تعداد گره‌ها و عملگرهای یکانی (Unary)

به عبارات و درختان زیر دقت کنید:

توضیحات	تعداد گره	درخت عبارت	عبارت
1 عملگر یکانی داریم	4 گره		- a * b
0 عملگر یکانی داریم(**عملگر یکانی نداریم**)	5 گره		a * b + c
2 عملگر یکانی داریم	5 گره		- a * - b

نتیجه :

تعداد گره‌های درخت عبارت	تعداد عملگر یکانی (Unary)	توضیحات
زوج	1,3,5,... (فرد)	حداقل ۱ عملگر یکانی داریم
فرد	0,2,4,... (زوج)	ممکن است عملگر یکانی نداشته باشیم

مثال : تعداد گره‌های یک درخت دودویی که یک عبارت ریاضی را نمایش می‌دهد 14 می‌باشد. عملگرهای این عبارت، دودویی یا یکانی (unary) می‌باشند. کدام‌یک از گزینه‌های زیر درست است؟

۱) این عبارت حتماً تعداد فردی عملگر دودویی دارد.
۲) این عبارت حتماً تعداد زوجی عملگر یکتایی دارد.
۳) این عبارت نمی‌تواند عملگر دودویی داشته باشد.
۴) حداقل یک عملگر یکانی در این عبارت وجود دارد.

حل: گزینه ۴ درست است.

عبارات infix (میانوندی) ـ postfix (پسوندی) ـ prefix (پیشوندی)

عبارت infix (میانوندی)

در هر عبارت عمومی محاسباتی هر عملگر دودویی بین عملوندهایش قرار می‌گیرد، که به این نحوه قرار گرفتن عملگرها در بین عملوندها روش میانوندی گفته می‌شود.

عملگر × بین عملوندهای a , b قرار دارد. ← a × b

در هر عبارت میانوندی (infix) اولویت عملگرها مطرح بوده و از ( ) برای تغییر اولویت عملگرها استفاده می‌شود.

در هر عبارت میانوندی (infix) اولویت عملگرها باتوجه به جدول اولویت عملگرها مشخص می‌شود.

عبارت postfix (پسوندی)

در این روش هر عملگر بعد از عملوندهایش قرار می‌گیرد.

عملگر × بعد از عملوندهایش قرار گرفته است. ab× ←

در هر عبارت postfix ، اولویت عملگرها مطرح نبوده و ( ) وجود ندارد.

به روش postfix روش لهستانی معکوس یا polish وارونه یا RPN (Reverse polish Notation) نیز می‌گویند.

عبارت prefix (پیشوندی)

در این روش هر عملگر قبل از عملوندهایش قرار می‌گیرد.

عملگر × قبل از عملوندهایش قرار گرفته است. $\times a\,b \rightarrow$

در هر عبارت prefix ، اولویت عملگرها مطرح نبوده و ( ) وجود ندارد.

به روش prefix روش لهستانی یا روش polish نیز می‌گویند چون اولین بار این روش توسط ریاضیدان لهستانی معرفی شد. به این دلیل است که به روش postfix روش لهستانی وارونه گفته می‌شود.

نتیجه:

$a \times b + c \rightarrow$ infix

$a\ b \times c + \rightarrow$ postfix

$+ \times a\ b\ c \rightarrow$ prefix

۱- در تبدیل عبارات infix ، postfix و prefix به یکدیگر ترتیب عملوندها به هیچ وجه عوض نمی‌شود.

۲ـ در هر عبارت postfix همیشه سمت راست عبارت عملگر است.

۳ـ در هر عبارت prefix همیشه سمت چپ عبارت عملگر است

تبدیل عبارت infix به عبارت‌های postfix و prefix

برای این تبدیل سه روش عمومی وجود دارد:

۱ـ تبدیل با پرانتزگذاری ۲ـ تبدیل با پشته (stack) ۳ـ تبدیل با پیمایش درخت عبارت محاسباتی

۱ـ روش پرانتزگذاری

تبدیل infix به postfix

برای این تبدیل به ترتیب زیر عمل می‌کنیم:

الف) عبارت را به طور کامل برحسب اولویت عملگرها پرانتزگذاری می‌کنیم در حین پرانتزگذاری عملگر مربوط به هر پرانتز را روی پرانتز بسته می‌گذاریم.

ب) عبارت را از چپ به راست (شامل عملوندها و عملگرهای روی پرانتزهای بسته) در خروجی می‌نویسیم.

مثال ۱:

$$\text{infix}: a \times b + c \rightarrow \left((a \times b)^{*} + c\right)^{+} \longrightarrow \text{postfix}: a\ ,\ b\ ,\ \times\ ,\ c\ ,\ +$$

الف)

$$\text{infix}: a \times (b + c) - g / d \longrightarrow \left(\left(a \times (b + c)^{*}\right)^{*} - (g / d)^{\prime}\right)^{-} \rightarrow \text{postfix}: a\ ,\ b\ ,\ c\ ,\ +\ ,\ \times\ ,\ g\ ,\ d\ ,\ /\ ,\ -$$

ب)

$$\text{infix}: \sqrt{b^{2} - 4ac} = \left(b \uparrow 2 - 4 \times a \quad .\right) \uparrow (1 / 2) \rightarrow \left(\left((b \uparrow 2)^{*} - \left((4 \times a)^{*} \times c\right)^{*}\right)^{-} \uparrow (1 / 2)^{\prime}\right)^{+}$$

ج)

$$\rightarrow \text{postfix}: b\ ,\ 2\ ,\ \uparrow\ ,\ 4\ ,\ a\ ,\ \times\ ,\ c\ ,\times\ ,\ -\ ,\ 1\ ,\ 2\ ,\ /\ ,\ \uparrow$$

$$\text{infix}: a \times b + c \uparrow d \uparrow e - f \rightarrow \left(\left((a \times b)^{*} + \left(c \uparrow (d \uparrow e)^{\uparrow}\right)^{\uparrow}\right)^{*} - f\right)^{-}$$

د)

$$\rightarrow \text{postfix}: a\ ,\ b\ ,\ \times\ ,\ c\ ,\ d\ ,\ e\ ,\ \uparrow\ ,\ \uparrow\ ,\ +\ ,\ f\ ,\ -$$

$$\text{infix}: a * - b \uparrow c \uparrow d - e / f \rightarrow \left(\left(a * \left((-b)^{-} \uparrow (c \uparrow d)^{\uparrow}\right)^{\uparrow}\right)^{*} - (e / f)^{\prime}\right)^{-}$$

ه)

$$\rightarrow \text{postfix}: a\ ,\ b\ ,\ -\ ,\ c\ ,\ d\ ,\ \uparrow\ ,\ \uparrow\ ,\ *\ ,\ e\ ,\ f\ ,\ /\ ,\ -$$

مثال ۲: عبارت postfix معادل عبارت $(A+B)*D+E/(F+A*D)$ برابر است با ..... (کارشناسی ارشد ـ علوم کامپیوتر ۸۰)

۱) $AB+D*E+F/A+D*$

۲) $AB+D*EFAD*+/+$

۳) $ABDEFAD+*+/+*$

۴) $AB+DE+FAD*+/$

حل: گزینه ۲ درست است.

$$(A+B)*D+E/(F+A*D)$$

$$\xrightarrow{\text{postfix:}} (((A+B)^{+}*D)^{*}+(E/(F+(A*D)^{*})^{+})^{/})^{+}$$

$$\rightarrow AB+D*EFAD*+/+$$

تبدیل infix به prefix

برای این تبدیل به ترتیب زیر عمل می‌کنیم:

الف) عبارت را به طور کامل برحسب اولویت عملگرها پرانتزگذاری می‌کنیم در حین پرانتزگذاری عملگر مربوط به هر پرانتز را روی پرانتز باز می‌گذاریم.

ب) عبارت را از چپ به راست (شامل عملوندها و عملگرهای روی پرانتزهای باز) در خروجی می‌نویسیم.

قبل از قسمت (الف) ابتدا وضعیت پرانتزهای عبارت را مشخص می‌کنیم.

مثال ۱:

$$\text{infix}: a * b + c \rightarrow {}^{*}\big({}^{+}(a * b) + c\big) \rightarrow \text{prefix}: +\,,*\,,a\,,b\,,c$$

الف)

$$\text{infix}: a * (b + c) - g / d \rightarrow {}^{-}\Big({}^{*}\big(a * {}^{+}(b + c) - {}^{/}(g / d)\big)\Big) \rightarrow \text{prefix}: -\,,*\,,a\,,+\,,b\,,c\,,/\,,g\,,d$$

ب)

$$\text{infix}: \sqrt{b^2 - 4ac} = (b \uparrow 2 - 4 * a * c) \uparrow (1 / 2) \rightarrow {}^{\uparrow}\Big({}^{-}\big({}^{\uparrow}(b \uparrow 2) - {}^{*}({}^{*}(4 * a) * c)\big) \uparrow {}^{/}(1 / 2)\Big)$$

$$\rightarrow \text{prefix}: \uparrow\,,-\,,\uparrow\,,b\,,2\,,*\,,*\,,4\,,a\,,c\,,/\,,1\,,2$$

ج)

د)

infix : a * b + c ^ d ^ e – f → $-\left(+\left(*(a * b)+\uparrow\left(c \uparrow \uparrow(d \uparrow e)\right)\right)-f\right)$

→ prefix : – , + , * , a , b , ↑ , c , ↑ , d , e , f

ه)

infix : a * – b ↑ c ↑ d – e / f → $-\left(*\left(a * \uparrow\left(-(-b) \uparrow \uparrow(c \uparrow d)\right)\right)-/(e / f)\right)$

→ prefix : – , * , a , ↑ , – , b , ↑ , c , d , / , e , f

مثال ۲: عبارت prefix معادل عبارت میانوندی زیر کدام است؟

A*B^(C+D^-E*F)/G-H*K

۱) -/*A^B+C*^D-EFG*HK

۲) -*^AB C+^DEF*/G-*HK

۳) -/*^A BC+*D^-EFG*HK

۴) ^^*ABC+-EF*/G-HK*

حل: گزینه ۱ درست است.

A * B ^ (C + D ^ - E * F) / G - H * K

- / * ^ + * ^ - *

(((A*(B^(C+((D^(-E))*F)))))/G)-(H*k))

prefix : → -/*A^B+C*^D-EFG*HK

۲ـ تبدیل با پشته (stack)

تعداد pop و push

در این تبدیلات به **کمک پشته** همواره **تعداد pop و push با هم برابر** بوده و از رابطه زیر بدست می‌آید:

تعداد پرانتزهای باز '(' + تعداد عملگرها = تعداد Push = تعداد pop

حداقل فضا برای پشته (حداقل متغیر میانی)

حداکثر فضایی که در حین تبدیل استفاده می‌شود، همان حداقل فضایی است که برای تبدیل مورد نیاز است.

تبدیل Infix به postfix

برای این تبدیل به ترتیب زیر عمل می‌کنیم:

الف) عبارت را از سمت چپ پیمایش می‌کنیم.

ب) عملوندها را در خروجی می‌نویسیم.

ج) به هر پرانتز '(' که رسیدیم آن‌را به راحتی داخل Stack قرار می‌دهیم.

د) به هر عملگر که رسیدیم به شرطی که اولویت آن عملگر از عملگر بالای stack بیشتر باشد، آن را داخل stack قرار می‌دهیم. در غیر این صورت آن‌قدر از بالای stack عملگر خارج کرده و در خروجی می‌نویسیم تا یا stack خالی شده و یا به عملگری برسیم که بتوانیم عملگر مورد نظر را روی آن در stack قرار دهیم.

یادآوری:

باتوجه به مورد (د)، عملگر + نمی‌تواند روی + یا ـ قرار گیرد، همین‌طور ـ روی + یا ـ و / روی * یا / و * روی * یا / نمی‌توانند قرار گیرند. اما توان (↑) روی توان (↑) می‌تواند قرار گیرد، چون اولویت توان‌های پشت سرهم از راست به چپ بررسی می‌شود.

ه) اگر بالای stack پرانتز '(' باشد هر عملگری به راحتی روی آن قرار می‌گیرد.

و) به هر پرانتز '(' که رسیدیم آن‌قدر از stack عملگر خارج کرده و در خروجی می‌نویسیم تا به '(' برسیم در این وضعیت '(' و ')' با هم خنثی می‌شوند.

ز) زمانی‌که به انتهای عبارت رسیدیم در صورتی‌که، stack خالی نباشد، آن را به طور کامل در خروجی می‌نویسیم.

مثال ۱: حداقل اندازهٔ پشته برای تبدیل عبارت میانوندی $a / (b - c * (d + e))$ به معادل پسوندی (postfix) کدام است؟

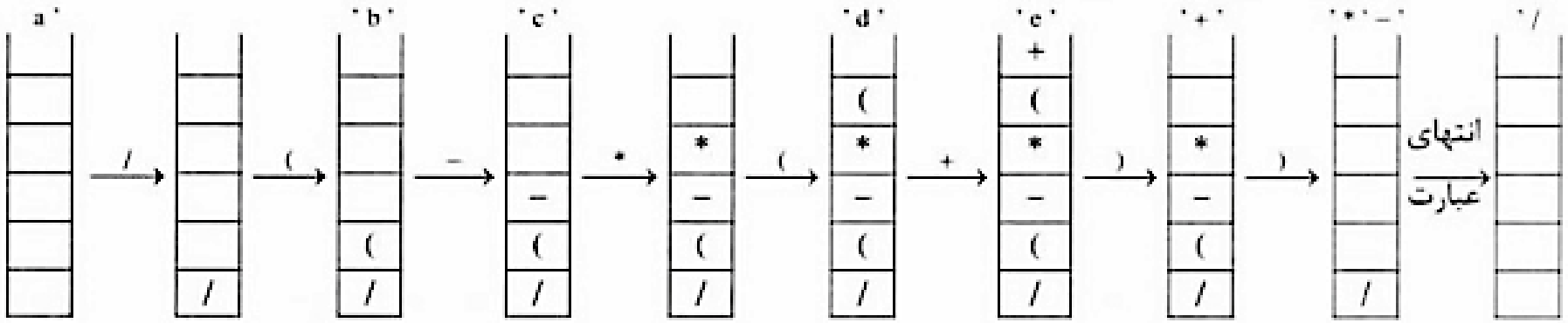
۱) 4 ۲) 5 ۳) 6 ۴) 3

حل: گزینه ۳ درست است.

مثال ۱: حداقل اندازهٔ پشته برای تبدیل عبارت میانوندی $a / (b - c * (d + e))$ به معادل پسوندی (postfix) کدام است؟

۱) 4 ۲) 5 ۳) 6 ۴) 3

از چپ به راست عبارت را پیمایش می‌کنیم.

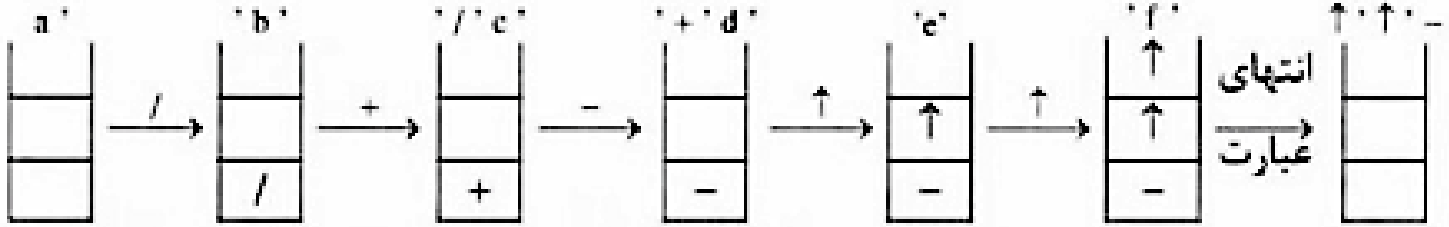


نتیجه را از چپ به راست می‌نویسیم: a , b , c , d , e , + , * , - , /

حداقل فضای مورد نیاز : 6

تعداد pop = تعداد push : تعداد عملگرها + تعداد'(' = 4 + 2 = 6

مثال۲: حداقل اندازه پشته برای تبدیل عبارت میانوندی a / b + c – d ↑ e ↑ f به پسوندی کدام است؟



نتیجه از چپ به راست: – , ↑ , ↑ , f , e , d , + , c , / , b , a خواهد بود.

حداقل فضای مورد نیاز: 3

تعداد pop = تعداد push : تعداد عملگرها + تعداد "()" = 0 + 5 = 5

تبدیل infix به prefix

برای این تبدیل به ترتیب زیر عمل می‌کنیم:

الف) عبارت را از سمت راست پیمایش می‌کنیم.

ب) عملوندها را در خروجی می‌نویسیم.

ج) به هر پرانتز ')' که رسیدیم آن‌را به راحتی داخل stack قرار می‌دهیم.

د) به هر عملگر که رسیدیم به شرطی که اولویت آن عملگر از عملگر بالای stack بیشتر یا مساوی باشد، آن را داخل stack قرار می‌دهیم، در غیر این صورت آن قدر از بالای stack عملگر خارج کرده و در خروجی می‌نویسیم تا یا stack خالی شده و یا به عملگری برسیم که بتوانیم عملگر مورد نظر را روی آن در stack قرار دهیم.

یادآوری:

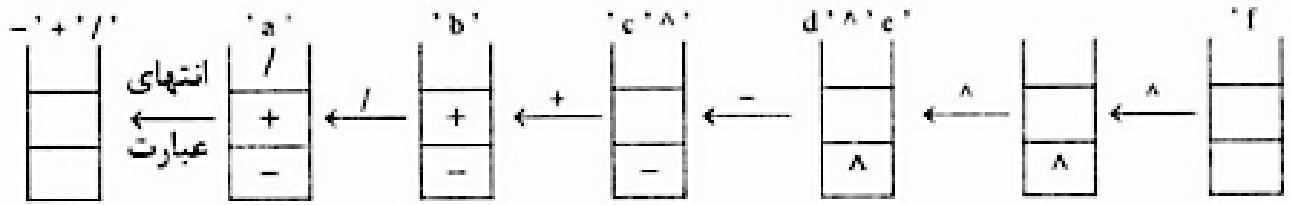
باتوجه به مورد (د)، عملگر ـ و + می‌توانند روی + یا ـ قرار گیرند، همین‌طور * و / می‌توانند روی * یا / قرار گیرند. اما توان ↑ روی توان ↑ نمی‌تواند قرار گیرد. چون اولویت توان‌های پشت سرهم از راست به چپ بررسی می‌شود.

ه) اگر بالای stack پرانتز '(' باشد هر عملگری به راحتی روی آن قرار می‌گیرد.

و) به هر پرانتز ')' که رسیدیم آن‌قدر از stack عملگر خارج کرده و در خروجی می‌نویسیم تا به '(' برسیم در این وضعیت '(' و ')' با هم خنثی می‌شوند.

ز) زمانی‌که به ابتدای عبارت رسیدیم (چون عبارات را از راست به چپ بررسی می‌کنیم) در صورتی‌که stack خالی نباشد، آن را به طور کامل در خروجی می‌نویسیم.

مثال ۱: حداقل اندازهٔ پشته برای تبدیل عبارت میانوندی a / b + c – d ^ e ^ f به عبارت پیشوندی معادل کدام است؟



نتیجه از چپ به راست: f , ^ , e , ^ , d , – , c , + , b , / , a , + , –

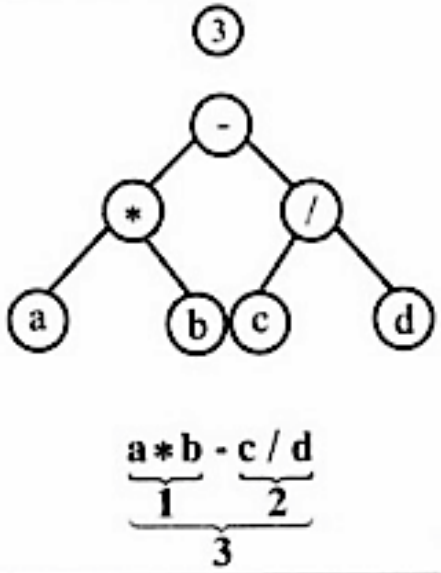
حداقل فضای مورد نیاز: ۳

تعداد pop = تعداد push : تعداد عملگرها + تعداد '(' = 0 + 5 = 5

۳ـ تبدیل با پیمایش درخت عبارت محاسباتی

در صورتی‌که درخت عبارت محاسباتی موجود باشد ، پیمایش‌های (preorder = VLR) ، (postorder = LRV)، (inorder = LVR) از آن به ترتیب نمایش‌های پیشوندی (prefix) ،پسوندی (postfix) و میانوندی (infix) را نشان خواهند داد.

مثال :

پیمایش‌های درخت	درخت عبارت	عبارت محاسباتی
VLR = -*ab/cd = prefix LRV = ab*cd/- = postfix LVR = a*b-c/d = infix		a * b - c / d

تبدیل عبارات prefix و Postfix به عبارات infix

برای این تبدیل دو روش عمومی وجود دارد:

۱ـ روش پرانتزگذاری

۲ـ استفاده از stack

۱ـ روش پرانتزگذاری

الف) ابتدا کنترل می‌کنیم که آیا تعداد عملوندها یک واحد از عملگرها بیشتر است، چون در غیر این صورت در عبارت عملگر یکانی وجود داشته و ارزشیابی آن شرایط خاصی خواهد داشت.

ب) عبارت را از سمتی بررسی می‌کنیم که با عملوند شروع می‌شود، در postfix از چپ و در prefix از راست عبارت را بررسی می‌کنیم.

ج) با رسیدن به هر عملگر دو عملوند در خلاف جهت در نظر می‌گیریم (در postfix سمت چپ عملگر و در prefix سمت راست عملگر)

د) برای دو عملوند، عملگر را بین آن‌ها قرار داده و برای آن‌ها پرانتز در نظر می‌گیریم.

ه) عبارت را از سمت چپ به راست به ترتیب شامل عملوندها و عملگرهای بین عملوندها ارزیابی می‌کنیم.

مثال ۱: عبارت را از سمت عملوندها (چپ) بررسی می‌کنیم.

$$A\,,B\,,C\,,*\,,D\,,/\,,+$$

$$\longrightarrow \left(A+,\left(\left(B^{*},C,\right)',D,/\right),+\right)\longrightarrow A+B*C/D$$

مثال ۲: عبارت را از سمت عملوندها (راست) بررسی می‌کنیم.

$$+\,,-\,,a\,,/\,,b\,,c\,,d$$

$$\longrightarrow \left(+,\left(-,a^{-},\left(/,b',c\right)\right)^{+},d\right)\longrightarrow a-b/c+d$$

مثال ۳: عبارت prefix زیر داده شده است: (کارشناسی ارشد ـ آزاد۷۹)

$$+-*\uparrow ABCD/E/F+GH$$

کدام‌یک از عبارات زیر معادل infix برای این عبارت است؟

۱) $A^{B*(C-D)}+E(F/(G+H))$

۲) $A^{B}*(C-D)+\frac{EF}{G}+H$

۳) $A^{B}*C-D+E/F/G+H$

۴) $A^{B}*C-D+E/(F/(G+H))$

حل: گزینه ۴ درست است.

از سمت عملوندها (راست) عبارت را بررسی می‌کنیم:

$$\Big(+\big(-\big(*(\uparrow A \uparrow B)^{*} C\big)^{-} D \big)^{+} \big(/E^{/}\big(/F^{/}(+G^{+} H)\big)\big)\Big)$$

$$= A \wedge B * C - D + E / (F / (G + H))$$

مثال۴: ارزش عبارت پسوندی روبرو کدام است ؟

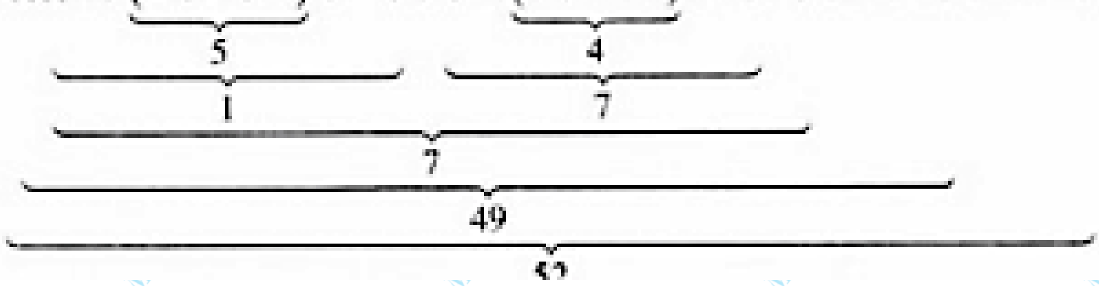
6 , 2 , 3 , + , – , 3 , 8 , 2 , / , + , * , 2 , ↑ , 3 , +

۱) 48 ۲) 52 ۳) 34 ۴) 7

حل: گزینه ۲ درست است.

منظور از ارزش عبارت پسوندی تبدیل آن به میانوندی **(infix)** است.

((((6 , (2 , 3 , +) , –) , (3 ,(8 , 2 , /) , +) *) , 2 , ↑) , 3 , +)



مثال۵: حاصل عبارت پیشوندی روبرو 3 , 2 , + , 5 , / , 10 , / , 1 , 1 , - , 3 , 2 , ^ , * , - , + کدام است؟

۱) 9 −　　۲) 2 -　　۳) 1　　۴) 10

حل: گزینه ۳ درست است.

$$\underbrace{\underbrace{+}_{-9+10=1}\ \underbrace{\underbrace{-}_{-8-1=-9}\ \underbrace{*\ \underbrace{\uparrow\ 2\ 3}_{2\uparrow 3=8}}_{8*(-1)=-8}\ \underbrace{-1}_{-1}}\ \underbrace{\underbrace{/\ \underbrace{10}_{10/1=10}\ \underbrace{/\ 5}_{5/5=1}}\ \underbrace{+\ 2\ \overline{3}}_{2+3=5}}}$$

منظور از ارزش عبارت پیشوندی تبدیل آن به میانوندی **(infix)** است.

۲ـ روش stack (پشته)

در این تبدیلات به کمک پشته همواره تعداد pop و push با هم برابر بوده و از رابطه زیر بدست می‌آید:

تعداد عملگرها 2× = تعداد Push = تعداد pop

حداقل فضا برای پشته (حداقل متغیر میانی)

حداکثر فضایی که در حین تبدیل استفاده می‌شود، همان حداقل فضایی است که برای تبدیل مورد نیاز است.

با فرض آن‌که عملگرها باینری (دودویی) هستند (تعداد عملگرها از عملوندها یک واحد کمتر باشد) به صورت زیر عمل می‌کنیم:

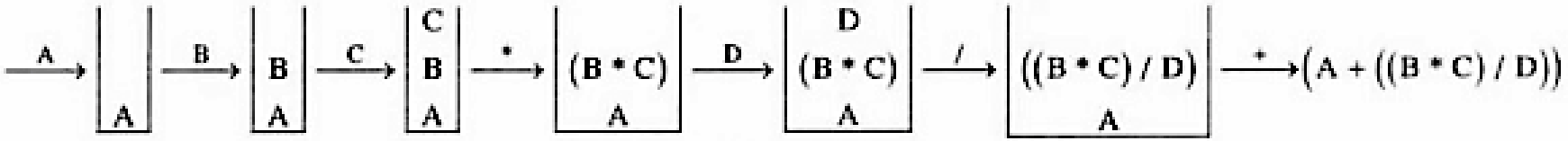
الف) عبارت را از سمتی بررسی می‌کنیم که با عملوند شروع می‌شود، در postfix از چپ و در prefix از راست عبارت را بررسی می‌کنیم.

ب) عملوندها را در stack قرار می‌دهیم و با رسیدن به هر عملگر (به جز عملگر آخر) دو عملوند خارج کرده و بعد از محاسبه دوباره حاصل عبارت را داخل stack قرار می‌دهیم. (دقت کنید که دو عملوندی که از stack خارج می‌کنیم از سمت چپ به راست عمل می‌کنیم)

ج) به عملگر آخر که رسیدیم بعد از خارج کردن دو عملوند آخر و محاسبه آن را دیگر در stack قرار نمی دهیم و حاصل را در خروجی می‌نویسیم.

مثال ۱:

postfix : $\bar{A}$, B , C , * , D , / , +



حداقل فضای مورد نیاز: 3

تعداد push = تعداد pop : تعداد عملگرها × 2 = 3 × 2 = 6

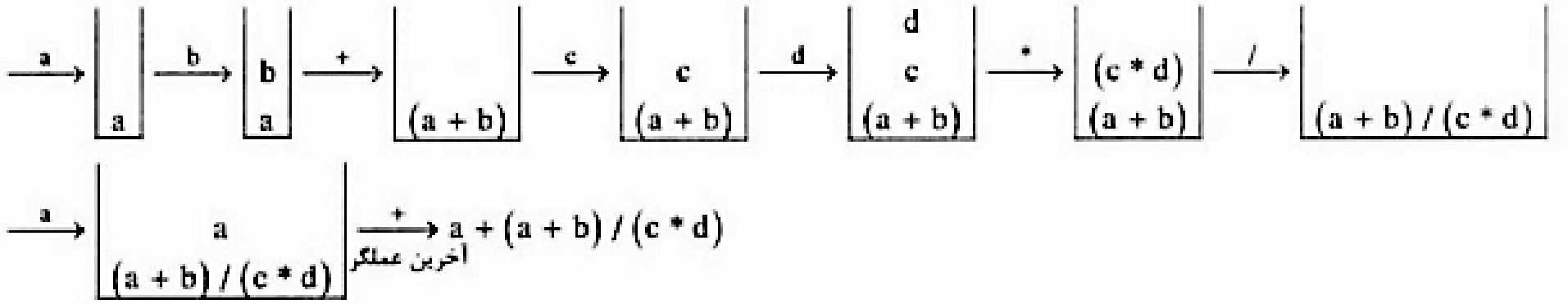
مثال ۲: مینیمم تعداد متغیرهای میانی در محاسبه عبارت جبری ab+cd*/a+ که به صورت postfix است، برابر است با ...

(کارشناسی ارشد ـ علوم کامپیوتر ۷۹)

۱) 1 ۲) 2 ۳) 3 ۴) 4

حل: گزینه ۳ درست است.

از سمت چپ بررسی می‌کنیم:



حداقل فضای مورد نیاز: 3

تعداد push = تعداد pop : تعداد عملگرها × 2 = 4 × 2 = 8

تبدیل prefix به postfix و بالعکس

۱ـ اگر در هنگام تبدیل prefix به infix در روش پرانتزگذاری به جای قرار دادن عملگر بین عملوندها، عملگر را روی پرانتز بسته بگذاریم به فرم postfix می‌رسیم.

۲ـ اگر در هنگام تبدیل postfix به infix در روش پرانتزگذاری به جای قرار دادن عملگر بین عملوندها، عملگر را روی پرانتز باز بگذاریم به فرم prefix می‌رسیم.

مثال ۱: معادل پیشوندی در عبارت پسوندی زیر کدام است؟

postfix : A B C * D / +

حل: از سمتی که با عملوندشروع می شود، عبارت را بررسی می‌کنیم (سمت چپ)

+ (A / (* (B C *) D /) +)

prefix :+ A / * B C D

مثال ۲: معادل پسوندی عبارت پیشوندی زیر کدام است؟

prefix : + a / b * + d e − a c

حل: از سمتی که با عملوندشروع می شود، عبارت را بررسی می‌کنیم (سمت راست)

$$= \Big(+ a \Big(/ b \Big(* \big(+ d e \big)^{+} \big(- a c \big)^{-} \big)^{*} \Big)^{/} \Big)^{+}$$

postfix :a b d e + a c − * / +

مثال۳: عبارت پیشوندی (prefix) زیر داده شده است: (کارشناسی ارشد ـ دولتی ۷۷)

$$++a/b-cd/-ab-+c*d5/a-bc$$

معادل پسوندی (postfix) آن کدام است؟

۱) $abcd-/+ab-cd5*+abc-/-/+$

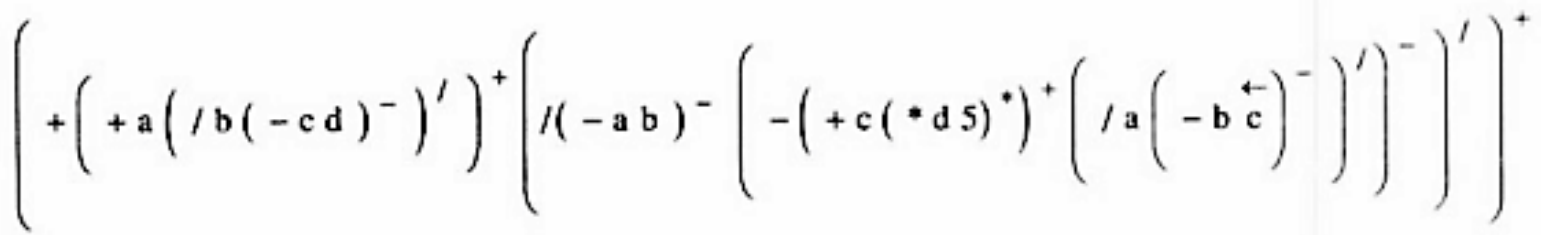
۲) $ab+cd-/ab-cd+/5*+ab/c--+$

۳) $ab+cd-ab-+/cd5*+/abc-/-$

۴) $abcd/-+abc-d5*abc-+/-/+$

حل: گزینه ۱ درست است.

از سمت عملوندها عبارت را بررسی می‌کنیم:



postfix: a b c d - / + a b - c d 5 * + a b c - / - / +