بسم الله الرحمن الرحیم

چگونه الگوریتم بنویسیم؟

شیوه مطالعه کتاب:

پس از مطالعه ی توضیحات در ابتدای فصل ها،مثال ها را فقط مطالعه کنید.در صورتی که ابهامی در مطالعه ی مثال ها برایتان وجود داشت ،توضیح مربوط به مثال را مطالعه کنید.تمریناتی که در طول یک فصل آورده شده را با کمک مثال های داده شده انجام دهید.تمرینات انتهایی هر فصل را بدون کمک از کتاب،خودتان به تنهایی انجام دهید و سپس برای تست درستی آنها،رعایت اصول مربوط به نوشتن الگوریتم را در صفحه ی 7 کتاب،چک کنید.

جدول trace را برای تمرینات رسم کرده و درستی جواب ها در خروجی را به صورت ذهنی با دستی تست کنید.

گرد آورنده:م.فرید

در صورت مشاهده ی اشکالات تقاضامند است به ایمیل masoumeh.farid@yahoo.com اطلاع رسانی نمایید.

الگوریتم[1] چیست؟

مجموعه ای از دستورالعمل ها که مراحل مختلف انجام کاری(حل یک مسئله)را به زبان دقیق،با جزییات کافی و ترتیب مناسب بیان می کند،به طوری که از یک نقطه شروع شده و خاتمه پذیر باشد.

چرا از الگوریتم استفاده می کنیم؟

در دنیای واقعی،زمانی که می خواهیم مجموعه دستورالعمل هایی را انجام دهیم که به هدف خاصی منجر گردد یا یک مسئله ریاضی ای را حل کنیم،آن را به صورت ذهنی تجزیه و تحلیل کرده و حتی در صورت ساده بودن، می توان آن را در مدت زمان کمی حل کرد.البته سرعت و دقت ما، خود مرهون آموزش هایی است که قبلا دیده ایم،در واقع ما داده های مورد نیاز خود را از طریق حواسمان دریافت و پس از درک آنچه از ما خواسته شده با قوای ذهنی خود،روش حل مسئله(گاهی ممکن است چند روش وجود داشته باشد) را با متناسب با شرایط اعم از سرعت،تعدد مراحل و... انتخاب می کنیم تا به هدف مربوطه برسیم.

کامپیوتر بر خلاف انسان،فاقد حواس،ذهن تحلیلگر و قوای تشخیص برای انجام مرتب دستور العملهاست و تنها زمانی به اجرای عملیات یا محاسبات می پردازد،که از قبل روش رسیدن به هدف معین،به طور کامل و با جزییات مربوطه در قالب یک زبان برنامه نویسی نوشته شده باشد.به طوری که وقوع کوچکترین جزییات با کمترین احتمالات در زمان اجرا پیش بینی شود،تا پس از تبدیل الگوریتم به زبان برنامه نویسی،برنامه بدون خطا اجرا گردد. سایر نکات در مورد الگوریتم در توضیحات مربوط به مثال ها شرح داده شده است،ضمن اینکه اصول نوشتن الگوریتم با حل تمرینات متعدد توسط شما موجب تبحر در نوشتن برنامه به زبان های برنامه نویسی مختلف خواهد شد.

مثالی در دنیای واقعی:

از ما خواسته شده شیوه ی تعویض چرخ پنچر شده را با توجه به توضیحات مربوط به خصوصیات الگورینم بنویسیم:

1- با استفاده از جک،اتومبیل را به اندازه ای که چرخ قابل تعویض باشد بالا ببرید.
2- پیچ های چرخ پنچر شده را بازکنید.
3- چرخ پنچرشده را از محل خود خارج کنید.
4- چرخ سالم را جایگزین کنید.
5- پیچهای چرخ را بچرخانید
6- اگر پیچها سفت نشده اند،به مرحله ی 5 برگردید،در غیر اینصورت نیاز به انجام کار دیگری نیست.

[1] به احترام ریاضیدان بزرگ ایرانی،ابو موسی خوارزمی و روشی که او در حل مسائل به شیوه منطقی به کار برده،این روش را الخوارزمی می نامیدند که بعد از معرب شدن این واژه بصورت الگوریتم بکار برده شده است.

در این مثال، ما ساده ترین اصول الگوریتم که شامل تشریح جز به جز و مرتب مراحل کار و نقطه ی آغاز و پایان معین برای دستورالعملهاست را رعایت کرده ایم،بنابراین اگر کسی بنا بر این دستورالعملها اقدام به تعویض چرخ کند،قطعا موفق خواهد شد.

شروع و پایان الگوریتم:

از آنجا که در الگوریتم های دست نویس و زبان های برنامه نویسی، مکان شروع و پایان الگوریتم یا برنامه، باید مشخص باشد،ما الگوریتم را با شماره ی 0 به عنوان شروع آغاز و پس از نوشتن کلیه ی خطوط،آخرین خط را با عنوان پایان می نویسیم.

چگونه الگوریتم بنویسیم؟

اگر مفهوم الگوریتم و مثال قبل را متوجه شده باشید،حال می توانید خودتان الگوریتم های مختلفی را برای کارهای روزمره بنویسید:دم کردن چایی،تماس تلفنی،دستور آشپزی،کار با وسایل الکتریکی و... .همانطور که این موارد را در ذهن خود مرور می کنید،می بینید که نوشتن الگوریتم های مربوطه برای شما بسیار ساده است،چرا که با روش انجام آنها به طور کامل آشنایی قبلی داشته و محدودیت خاصی برای بیان آن با زبان گفتاری ندارید.

هدف ما در ادامه ی این فصل،یادگیری نوشتن و تست الگوریتم های ساده با محاسبات ریاضی خواهد بود

شروع به نوشتن الگوریتم

مثال 1:الگوریتمی بنویسید که دو عدد را از ورودی دریافت و آنها را با هم جمع کند.

- **قدم اول:** باید صورت مسئله را درک کنید،این امر زمانی ممکن است که آنچه از شما خواسته شده،بدون ابهام باشد:**کدام دو عدد؟**دو عدد خاص و ثابت یا اینکه چه اعدادی با هم جمع شوند اهمیتی ندارد؟در این صورت مسئله، عدد خاصی تعیین نشده،بنابراین الگوریتم باید قادر باشد هر دو عددی را با هم جمع کند. اما **این دو عدد چگونه به وجود خواهند آمد؟** با ارجاع به یک صفحه ی خاص در یک کتاب،با دو مقدار فرضی مثل 2و4، یا توسط یک کاربر یا مجری (ورودی) تعیین شود؟با نگاهی به صورت مسئله مشخص می شود که اعداد باید از ورودی (کاربر یا مجری الگوریتم یک مقدار دلخواه را اعلام کند)دریافت شود.**قدم دوم:** بر اساس اصول گفته شده ی قبلی(شروع و پایان مشخص،ترتیب در نوشتن دستورات،دقت درجزییات و...)با انتخاب یک **روش معین** برای رسیدن به هدف صورت مسئله،شروع به نوشتن الگوریتم کنید :

0- شروع
1- عدد اول را دریافت کن
2- عدد دوم را دریافت کن
3- دو عدد را با هم جمع کن.

مجری الگوریتم می تواند در زمان تست الگوریتم،هر دو عددی را که می خواهد به عنوان ورودی در نظر بگیرد.

4- حاصل جمع را بنویس(معمولا هدف ما از نوشتن الگوریتم نمایش یک نتیجه است)

5- پایان

اصلاح الگوریتم قبل:

از آنجا که نوشتن جملات طولانی موجب کاهش سادگی در پیگیری راه حل مسئله می گردد،الگوریتم خود را پس از درک مفهوم **متغیر** در برنامه نویسی، اصلاح می کنیم.

متغیر چیست؟

همان طور که از اسم این کلمه پیداست،متغیر چیزیست که به دلیلی تغییر می کند،برای مثال ظرف نگهدارنده ی آب می تواند متغیری باشد که به شربت،شیر،آبمیوه و... ممکن است تغییر یابد.در برنامه نویسی،متغیر بایک نام مشخص به عنوان محلی در حافظه ی رایانه،می تواند مقداری را در خود نگه دارد،به طوری که بعداً قابل دسترسی و یا تغییر است،برای مثال،به متغیری به نامa(نام متغیر می تواند ترکیبی از حروف و اعداد باشد و بهتر است این نام مرتبط با عملکرد متغیر باشد) مقدار 10 را نسبت داده ایم ،در هر خط از الگوریتم اگر تغییری روی a اعمال کنیم در واقع مقدار نسبت داده شده به آن را تغییر داده ایم،مثلاً دستور a←a+1 a،(حاصل عبارت سمت راست در متغیر سمت چپ ریخته شود)موجب می شود که مقدار aاز 10 به 11 تغییر کند.

حال،ما برای اصلاح الگوریتم مثال قبل به جای عدد اول و عدد دوم از دو متغیر به نام های a,b استفاده می کنیم،ضمن اینکه متغیری را برای نگه داری مجموع این دو عدد به منظور نمایش در خروجی فرضی در نظر گرفته ایم.

0- شروع

1- a را بگیر

2- b را بگیر

3- sum←a+b(از علامت← برای نمایش قرار گرفتن مقدار یا حاصل عبارت سمت راست در متغیر سمت چپ استفاده می شود)

4- sum را بنویس(معمولا هدف ما از نوشتن الگوریتم نمایش یک نتیجه است)

5- پایان

سوال:نام متغیر مربوط به نگهداری حاصل جمع چیست؟

مثال 2:

الگوریتمی بنویسید که طول و عرض مستطیلی را از ورودی گرفته،محیط و مساحت آن را محاسبه کند ودر خروجی نمایش دهد:

در مثال های این فصل،برای آشنایی و تمرکز شما در مورد چگونگی نوشتن یک الگوریتم،قبل از نوشتن هر الگوریتم آن را تجزیه و تحلیل کرده و **هدف،روش نوشتن الگوریتم یا روال کار،ورودی و خروجی** را تعیین می کنیم.

در این مثال داریم:

هدف:محاسبه ی محیط و مساحت یک مستطیل

روش: جایگزینی طول و عرض در فرمول های زیر:

طول*عرض= محیط مستطیل (طول+عرض)*2= مساحت مسطیل

ورودی ها(داده های مورد نیاز برای رسیدن به هدف بر اساس روش انتخابی):طول و عرض

خروجی(که معمولا نمایش هدف یا قسمتی از آن است):محیط و مساحت

قبل از نوشتن الگوریتم یا در طول نوشتن آن،بر اساس ورودی ها و همچنین روش انتخابی ، در صورت نیاز از متغیرهای مشخصی استفاده می گردد، **عموما برای گرفتن هر داده از ورودی،یک متغیر** با کابرد معین،استفاده می شود،در اینجا،ما از حرف m برای طول و حرف n برای عرض(نام متغیر می تواند حرف یا کلمه باشد) استفاده می کنیم:

0- شروع

1- m را دریافت کن

2- n را دریافت کن

آیا متغیر دیگری نیاز داریم؟**در الگوریتم هایی که محاسبات ریاضی صورت می گیرد،معمولاً به متغیری برای نگهداری حاصل عبارات نیاز داریم**،تا اگر بخواهیم حاصل به دست آمده را در خروجی نمایش دهیم و یا در محاسبات بعدی خود از آن استفاده کنیم،به حاصل عبارت خود دسترسی داشته باشیم،این کار با اختصاص نامی به عنوان یک متغیر صورت می گیرد:ما از mohit و masahat برای این منظور استفاده می کنیم،پس:

3- mohit←(m+n)*2

4- masahat←m*n

تا اینجا،ورودی ها تعیین و خطوط مربوطه طبق روش انتخابی نوشته شد،به طوری که ما به هدف مورد نظر خود رسیدیم و حالا نوبت نمایش خروجی است،که می تواند با کلماتی مانند:چاپ کن یا بنویس مشخص شود و اگر ما نام متغیری را به منظور چاپ یا نوشتن ذکر کنیم،در واقع، منظور،نمایش مقداری است که متغیر در خود نگه می دارد نه نام لاتین آن.

5-Mohit,masahat را بنویس.

نهایتاً خطی را به عنوان پایان الگوریتم اختصاص می دهیم- الگوریتم کامل مثال قبل به صورت زیر است :

0- شروع

1- m را دریافت کن

2- n را دریافت کن

3- mohit←(m+n)*2

4- masahat←m*n

5- Mohit,masahat را بنویس.

6- پایان

کار نوشتن الگوریتم پایان یافت،برای اینکه مشخص شود که آیا با اجرای آن به هدف خود خواهیم رسید یا نه و الگوریتم نوشته شده نیاز به اصلاح دارد یا خیر،آن را با داده های فرضی امتحان می کنیم، برای این منظور از جدولی به نام جدول trace یا جدول تست الگوریتم استفاده می شود.در واقع می خواهیم دستورالعمل های خطوط الگوریتم را به ترتیب از شروع تا پایان،خط به خط انجام داده و با جایگذاری مقادیر متغیرها در جول چک کنیم که بر اساس داده های فرضی در ورودی،خروجی چه خواهد شد(آیا به هدف مورد نظر در صورت مسئله خواهیم رسید؟)

چگونگی رسم جدولTrace:

الف)کشیدن یک جدول ساده،با اختصاص یک ستون به هر متغیر و در نظر گرفتن یک ستون به عنوان خروجی.برای مثال قبل داریم:

m	n	mohit	Masahat	خروجی

ب)الگوریتم را از نقطه ی شروع آغاز و هر زمان که طبق دستورات الگوریتم،مقداری از ورودی خوانده می شود،یک مقدار فرضی را زیر ستون متغیر مربوطه در جدول می نویسیم،همین کار را برای متغیرهایی که مقدارشان در محاسبات ریاضی یا عملیات انتساب(مقداردهی یک متغیر با یک مقدار یا متغیری دیگر) در الگوریتم تعیین می شود،انجام می دهیم. یعنی مقدار به دست آمده یا تعیین شده را زیر ستون مربوط به آن متغیر می نویسیم.

برای مثال اگر در خط 1 الگوریتم قبل،مقدار فرضی برای طول 4 باشد،عدد 4 در ستون اول قرار گرفته و اگر در خط 2،عدد 3 برای عرض دریافت شود،3 در ستون مربوط به متغیر n قرار می گیرد،در خط 3 نیز ابتدا مقادیر اختصاص داده شده به m,n با هم جمع شده و سپس در عدد 2 هم ضرب می شوند و حاصل در ستون مربوط به متغیر mohit قرار می گیرد.(شماره ی خط مربوط به مقداردهی هر متغیر بالای ستون آن نوشته شده است)

خط 1 m	خط 2 n	خط 3 mohit	خط 4 masahat	خط 5 خروجی
4	3	14	12	12و14
8	2			
6	4			
2	1			

تمرین1:الگوریتم فوق را با داده های فرضی در ادامه ی جدول تست کنید.

تمرین2:جدول Traceمربوط به مثال یک را رسم کرده و با مقادیر فرضی a=8,b=23 گرفته شده از ورودی تست کنید.

تمرین3:موارد زیرا را برای نوشتن الگوریتمی که محیط و مساحت یک دایره را محاسبه و در خروجی نمایش دهد،مشخص کنید:

الف)هدف:..... ب)روش رسیدن به هدف[2]:.....

ج)ورودی های موردنیاز:..... د)خروجی:.....

نکات زیر را مرور کرده،الگوریتم مربوط به تمرین 3 را بنویسید:(**اصول نوشتن الگوریتم**)

1. نقطه ی شروع الگوریتم مشخص باشد(اختصاص شماره ی 0 به خط شروع)
2. دستورات مبهم نباشد.
3. ترتیب دستورالمعل ها به صورتی باشد که ابتدا خطوط پیش نیاز نوشته شود(برای مثال حتماً قبل از انجام عملیات محاسباتی، ورودی ها ی مربوطه دریافت شده باشد)
4. خطوط یا به صورت امری است و یا شرطی.(aرا دریافت کن،یک دستور امری است،دستورات شرطی بعدا توضیح داده خواهد شد)
5. استفاده از متغیرها را در الگوریتم خود فراموش نکنید.

پس از نوشتن الگوریتم،چک کنید که آیا موارد فوق را رعایت کرده اید یا خیر؟در صورت نیاز آن را اصلاح کنید.

تمرین4: الگوریتم خود را با مقادیر فرضی زیر تست نموده و درستی اجرای آن را با جواب های داده شده چک کنید.(شعاع=2وشعاع=4)

12.56=مساحت دایره	12.56=محیط دایره	2=شعاع
50.24=مساحت دایره	25.12=محیط دایره	4=شعاع

مثال: الگوریتمی بنویسید که میانگین سه عدد دریافتی از ورودی را محاسبه و در خروجی نمایش دهد.

0- شروع
1- a,b,c را دریافت کن
2- sum←a+b+c
3- avg←sum/3
4- avg را بنویس
5- پایان

سوال: متغیرهای sum و avg به ترتیب به چه منظور استفاده شده اند؟

الف) ورودی- نگهدارنده ی مجموع سه عدد **ب)** نگهدارنده ی مجموع سه عدد- نگه دارنده ی میانگین

تمرین: جدول trace مثال را با داده های فرضی a=19,b=17,c=18 رسم کنید.

[2] - شعاع*3.14*2=محیط دایره شعاع*شعاع*3.14=مساحت دایره

اگر موفق به انجام تمرینات قبل شدید و قادر به انجام مثال ها نیز هستید،ادامه ی مباحث را در فصل های بعد دنبال کنید، مگر اینکه مطمئن باشید اشکالات شما با انجام مثال های بیشتر برطرف می گردد. چرا که مفاهیم قبلی پایه و اساس نوشتن الگوریتم و برنامه به زبان های برنامه نویسی است.

فصل دوم:الگوریتم های ساده

قبل از شروع این فصل، تقسیم بندی از انواع الگوریتم انجام می دهیم:

- الگوریتم هایی با دستورات ساده(مشابه مثال های فصل قبل)
- الگوریتم ها با دستورات شرطی
- الگوریتم ها با تکرار تعدادی از دستورات(حلقه)

گرچه عموم الگوریتم ها ترکیبی از موارد فوق است،با این تقسیم بندی در واقع سعی می کنیم،کاربرد انواع دستورات در الگوریتم را با مثال های ساده شروع کنیم تا قادر به تجزیه ی صورت مسائل برای نوشتن الگوریتم ها ی پیچیده باشیم.

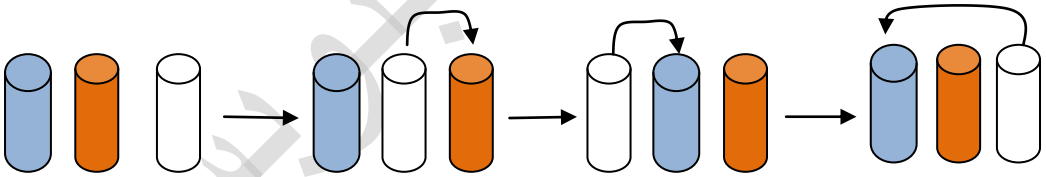
الگوریتم های ساده:

منظور ما در اینجا از الگوریتم های ساده،الگوریتم هایی است که روال کار آن شامل دستورات ساده ی محاسباتی و یا فرمول های ریاضی بوده و تنها با یکبار انجام هر خط،در پایان الگوریتم به هدف خود یا نمایش خروجی می رسیم.مثال های فصل قبل نیز از این نوع بود.در ادامه پس از یک مثال دیگر از این نوع دستورات،مثال هایی از ترکیب این الگوریتم ها با انواع شرطی آورده شده است:

مثال 1-2:

الگوریتمی بنویسید که دو عدد را از ورودی دریافت و سپس مقادیر دو متغیر را جابجا کرده و در خروجی نمایش دهد.

هدف:جابجایی مقادیر دو متغیر **روش:**روش خود را با ذکر مثالی در دنیای واقعی شرح می دهیم،زمانی که می خواهیم،محتویات دو لیوان آب و نوشابه را با هم جابجا کنیم از یک ظرف کمکی استفاده می کنیم.



برای جابجایی دو متغیر نیز از یک متغیر کمکی استفاده می کنیم.به این ترتیب که ابتدا یکی از متغیرهای ورودی را به متغیر کمکی انتساب داده(help←a)،سپس متغیر دیگر را به متغیری که مقدار آن را قبلا در help ذخیره کرده ایم،نسبت می دهیم(a←b) و در نهایت،مقدار قبلی متغیر اول را که اکنون در متغیر کمکی است به متغیر دوم نسبت می دهیم(b←help).

0- شروع

1- a,b را دریافت کن

2- help←a

3- a←b

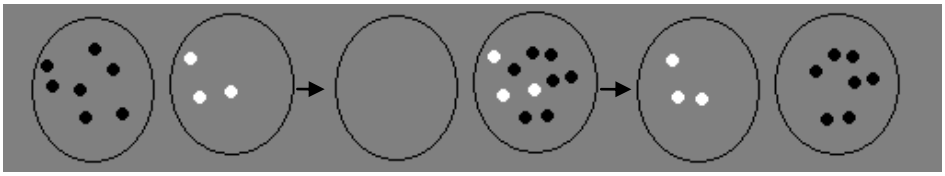
4- b←help
5- a,b را بنویس
6- پایان

تمرین: جدول trace الگوریتم قبل را رسم نمایید و با داده های فرضی a=7 و b=3 آن را تست نمایید.

الگوریتم مثال قبل را بدون استفاده از متغیر کمکی نیز می توان نوشت:

0- شروع
1- a,b را دریافت کن
2- a←a+b
3- b←a-b
4- a←a-b
5- a,b را بنویس
6- پایان

برای درک این روش،دو کیسه را در نظر بگیرید که درون یکی از آنها،چند مهره ی سیاه و در دیگری مهره هایی به رنگ سفید قرار گرفته است،برای جابجایی مهره های درون این کیسه ها،مهره های سفید را درون کیسه با مهره های سیاه ریخته(یا بالعکس) سپس مهره های سیاه را جدا کرده و درون کیسه ی خالی می اندازیم.



قبل از تست این الگوریتم،نحوه ی پر کردن ادامه ی سطرها در جدول trace را تشریح می کنیم:زمانی که تغییرات مربوط به یک یا چند متغیر در بیشتر از یک خط الگوریتم صورت می گیرد،یعنی پس از دریافت متغیر از ورودی یا مقداردهی اولیه آن،در خطوط بعدی الگوریتم یا برنامه،مقدار جدیدی به همان متغیر نسبت داده می شود.مقدار قبلی از بین رفته و مقدار جدید را در ستون مربوط به آن متغیر در جدول می نویسیم.پس از آن نیز،هر زمان که در دستورات از این متغیر استفاده می شود،آخرین مقدار نوشته شده در ستون متغیر مربوطه باید در نظر گرفته شود.

a	b	خروجی
7	3	
10	7	
3		3و7

خروجی جدول فوق را با خروجی جدول خودتان در مثال قبل مقایسه کنید.

مثال2-2:الگوریتمی بنویسید که با اجرای آن،عملیات محاسباتی ساده ی ریاضی(+و*و/و-)را روی دو عدد دریافت شده از ورودی انجام داده و حاصل مربوط به هر محاسبه در خروجی نمایش داده شوند.(قسمت های زیر را خودتان تکمیل کنید)

هدف:.....روش:.....

ورودی ها:.....خروجی ها:.....

0-شروع

؟ 1- a,b را دریافت کن.

دستورات مربوط به روش حل و رسیدن به هدف:
2- sum←a+b
3- mul←a*b
4- sub←a-b
5- div←a/b

؟ 6- sum,mul,sub,div را بنویس

7- پایان

؟) سعی کنید دستورات مربوط به هر قسمت(هدف،ورودی و...) را کنار علامت {،مشخص کنید.برای مثال خطوط 2 تا 5 مربوط به روش الگوریتم و رسید به هدف است.

✓ توجه کنید که متغیرهای sum,mul,sub,div هر کدام برای نگه داری حاصل یکی از عبارات محاسباتی است،برای مثال a,b با هم جمع شده و حاصل جمع آنها در sum ریخته می شود تا بعدا در خروجی نمایش داده شود.

جدول Trace زیر،اجرای خط به خط الگوریتم را نشان می دهد.(الگوریتم را دو بار و با داده های مختلف تست کرده ایم،اجرای سوم را شما با داده های فرضی a=9,b=0 انجام دهید)

a	b	sum (خط 2)	mul (خط 3)	sub (خط 4)	div (خط 5)	خروجي (خط 6)
3	4	7	12	-1	0.75	7,12,-1,0.75
8	2	10	16	6	4	10,16,6,4
9	0				؟	

اگرچه الگوریتم فوق با داده های فرضی بسیاری به طور کاملاً صحیح عمل می کند اما زمان تست برنامه با داده های فرضی 9و0 ،هنگامی که به خط 5 در الگوریتم می رسیم،مقدار عددی برای حاصل این عبارت وجود ندارد،در زبان های برنامه نویسی اجرای برنامه با این داده ها موجب بروز خطا و عدم اجرای صحیح برنامه می گردد.حال می خواهیم با همین فرض الگوریتم خود را اصلاح کنیم،چرا که این الگوریتم ها بعدا به صورت برنامه در خواهند آمد و همانطوری که قبلا گفتیم،برنامه باید قادر به اجرا با هر نوع داده ی فرضی در ورودی و با کمترین احتمال باشد.

در ادامه،پس از مطالعه ی الگوریتم ها با دستورات شرطی،الگوریتم قبل را اصلاح و مجدداً تست می کنیم.

الگوریتم با دستورات شرطی

مثالی در دنیای واقعی:

شما می توانید شرایط خود را برای انجام هر عملی به صورت کلامی بیان کرده یا در ذهن خود با توجه به شرایط موجود تصمیمی را اتخاذ کنید:

- ✓ **اگر** هوا آفتابی بود،**آنگاه** عینک آفتابی ات را همراه خود بیاور درغیر اینصورت **اگر** هوا بارانی بود **آنگاه** چترت را همراه خود بیاور

به طور کلی:

اگر (شرط) **آنگاه** انجام دستور یا دستورات مربوط به شرط

یا

اگر (شرط) **آنگاه** انجام دستور یا دستورات مربوط به شرط
در غیر اینصورت انجام دستورات مربوط به عدم برقراری شرط
انجام سایر دستورات

در نوشتن الگوریتم و زبان های برنامه نویسی نیز،می توان شروط مورد نیاز برای اجرا یا عدم اجرای دستور یا دستورات مورد نظر را مشخص کرد.از این طریق خوانایی الگوریتم و برنامه بالا رفته و دست ما برای نوشتن دستورات متنوع باز است.

اصلاح مثال 2-2:

در صفحه ی قبل به این نتیجه رسیدیم:**در صورتی که در محاسبات،مقسوم علیه برابر با 0 شود،مقدار عددی به عنوان خروجی تولید نخواهد شد**،حال می خواهیم الگوریتم را طوری بنویسیم که اگر عدد دوم 0 بود،دستور مربوط به عمل تقسیم انجام نشود،در ادامه دو نمونه الگوریتم اصلاح شده برای این مثال آورده شده است.

0-شروع
1- a,b را دریافت کن.
2- sum←a+b
3- mul←a*b
4- sub←a-b
5- اگر b=0 آنگاه

اگر b برابر با صفر باشد،از خط 0 تا 4 این الگوریتم مشابه الگوریتم قبل است،در خط 5 به منظور اصلاح الگوریتم قبلی،از یک شرط استفاده کرده ایم که توسط آن عدد دوم را چک می کنیم و **به این صورت درنظر گرفته می شود**:اگر b برابر با صفر است آنگاه ...،و با اجرای آن اگر شرط برقرار باشد یعنی مقدار متغیر bبرابر صفر باشد،ادامه ی دستورات بعد از آنگاه انجام می شود یعنی چاپ پیغام"second number is zero" (دومین عدد صفر است)و مقادیر متغیرهای sum,mul,subو نهایتا اجرا به خط 8یعنی پایان الگوریتم خواهد رفت.(از علامت " برای نشان دادن پیغام استفاده می شود)

بنویس "second number is zero",sum,mul,subو برو به خط 8
6- div←a/b
7- sum,mul,sub,div را بنویس
8- پایان

اگرb برابر با صفر نباشد،پس از خطوط 0 تا 4،دستور شرطی خط 5 انجام می شود و چون شرط برقرار نیست،ادامه ی دستورات پس از آنگاه در ادامه ی خط انجام **نشده** و اجرای ادامه ی الگوریتم به خط بعد(خط 6)خواهد رفت. در جدول های trace رسم شده دو اجرای مختلف الگوریتم را با شماره خطوط مربوطه بالای هر ستون مشاهده می کنید

خط 1	خط 2	خط 3	خط 4	خط 5
a	b	sum	mul	sub	div	خروجي
9	0	9	0	9		9,0,9,second number is zero

خط 1	خط 2	خط 3	خط 4	خط 6	خط 7
a	b	sum	mul	sub	div	خروجي
6	3	9	18	3	2	9,18,3,2

روشی دیگر برای نوشتن مثال 2-2:

0-شروع
2- a,b را دریافت کن.
3- sum←a+b
4- mul←a*b
5- sub←a-b
6- اگر b≠0 آنگاه div←a/b و
sum,mul,sub,div را بنویس
7- اگر b=0 آنگاه (sum,mul,sub,"second number is zero" را بنویس.)
8- پایان

خط 6 به این صورت خوانده می شود،**اگر b مخالف صفر بود** آنگاه...،(علامت ≠ به معنی مخالف یا نا برابر است)به این ترتیب اگر عدد دوم صفر نباشد،ابتدا تقسیم انجام شده،سپس اجرای دستورات بدون انجام خط 7 به خط 8 خواهد رفت.**اگر b برابر صفر باشد**،در خط 6 پس از تست شرط،از آنجا که شرط برقرار نیست،اجرا به خط بعد رفته و شرط بعدی تست می گردد...

تمرین: جدول trace الگوریتم فوق را با داده های فرضی a=9,b=0 و a=6,b=3 رسم کرده و خروجی را با خروجی الگوریتم قبل مقایسه کنید.

اگر تمرین قبل را به درستی انجام داده باشید،خواهید دید که خروجی دو الگوریتم مشابه هم است و فقط نحوه ی نوشتن دستورات متفاوت است.دستورات شرطی می توانند در قالب های مختلفی نوشته شود،ضمن اینکه در زبان های برنامه نویسی دستورات لازم نیز برای این منظور وجود دارد.

مثال 2-3: الگوریتمی بنویسید که عددی صحیح را از ورودی دریافت و مشخص کند،عدد زوج است یا فرد؟(اگر زوج بود even و اگر فرد بود odd در خروجی نشان داده شود)

هدف: تعیین زوج یا فرد بودن عدد **روش:** باقیمانده ی عدد را بر 2 محاسبه می کنیم،اگر 0 باشد،عدد زوج است، در غیر اینصورت عدد فرد است.

ورودی: یک عدد صحیح **خروجی:** نتیجه ی تعیین شده در هدف

در قسمت روش،نحوه ی تشخیص زوج یا فرد بودن را نوشته ایم،حالا نیاز به فرمول یا دستوراتی داریم که کار محاسبه ی باقیمانده را انجام دهد.فرمول ریاضی مربوطه به صورت زیر است:

باقیمانده ی a بر b[3] = a-([a/b]*b)

مثلا اگر a=9 و b=4 باشد،باقیمانده ی تقسیم a/b:

[3] [ ] علامت جزء صحیح در ریاضی.

9-([9/4]*4)=9-(2*4)=9-8=1

ضمن اینکه در زبان های برنامه نویسی برای این منظور معمولا عملگری وجود دارد،مانند mod در زبان پاسکال یا ٪ در زبان C که فرمول فوق را به روی مقادیر خواسته شده اعمال می کند:

10 mod 6=4 25 mod 2=1 7 mod 3=1

در این مثال ،a ورودی و b عدد 2 خواهد بود(محاسبه ی باقیمانده ی عدد دریافتی بر 2) a-([a/2]*2)

0- شروع

1- a را دریافت کن

2- r← a-([a/2]*2) (متغیر rبرای نگهداری باقیمانده است)

3- اگر r=0 آنگاه بنویس "عدد زوج است" و بروبه 5

4- اگر r=1 آنگاه بنویس "عدد فرد است"

5- پایان

a	r	خروجي
8	0	عدد زوج است
25	1	عدد فرد است

خط 1 خط 2 خط 3 خط 4

دو اجرای مختلف برنامه را در جدول trace تست کرده ایم.در اجرای اول از آنجا که باقیمانده صفر شده خط سوم به طور کامل انجام می شود به طوری که پیغام مربوطه نمایش و ادامه ی اجرا بدون انجام خط 4 به خط پایان هدایت می شود،در تست دوم،شرط خط 3 بررسی شده و از آنجا که مقدار r مخالف صفر است بدون انجام دستوراتِ پس از آنگاه در این خط،اجرای الگوریتم به خط 4 رفته و خروجی مشخص می گردد و اجرا به ترتیب دستورات به خط بعد یعنی پایان می رود.

اصلاح الگوریتم:

صفر،زوج است یا فرد؟همانطور که می دانید 0 نه زوج است و نه فرد.

تمرین1:الف)دستوری را در خط 2 الگوریتم زیر بنویسید که با در نظر گرفتن احتمال وارد شدن عدد 0 از ورودی،در خروجی،پیغام مربوطه("صفر نه زوج است و نه فرد")را نشان داده و اجرای دستورات به پایان الگوریتم هدایت شود:

0- شروع

1- a را دریافت کن

2- ؟

3- r← a-([a/2]*2)

4-اگر r=0 آنگاه بنویس "عدد زوج است" و برو به 6

5-اگر r=1 آنگاه بنویس "عدد فرد است"

6- پایان

ب)الگوریتم خود را با داده های 8و25و0 تست کنید،خروجی را با جدولی که قبلاً رسم کرده ایم مقایسه کنید.

مثال4-2:الگوریتمی بنویسید که دو عدد از ورودی دریافت و عدد بزرگتر را مشخص نماید:

هدف:تعیین عدد بزرگتر بین دو عدد **روش:**مقایسه دو عدد با هم در دستورات شرطی و تعیین عدد بزرگتر

ورودی:دو عدد **خروجی:**عدد بزرگتر

0- شروع

1- a,b را دریافت کن

2- اگر a>b آنگاه بنویس a

3- اگرb>a آنگاه بنویس b

4- پایان

تمرین:الگوریتم فوق را طوری اصلاح کنید تا در صورت برابر بودن دو عدد دریافت شده،پیغام "دو عدد با هم برابرند" را در خروجی نشان دهد.جدول trace الگوریتم خود را با داده های فرضی زیر تست کنید:

A=3,b=9 a=11,b=2 a=7,b=7

تمرین:الگوریتمی بنویسید که دو عدد از ورودی دریافت و عدد کوچکتر را در خروجی نمایش دهد،اگر دو عدد با هم برابر بود،پیغام "دو عدد با هم برابرند" را در خروجی نشان دهد.

تمرین:الگوریتمی بنویسید که عددی را از ورودی دریافت و تعیین کند که عدد مثبت است یا منفی؟

هدف:تعیین مثبت یا منفی بودن عدد دریافت شده

روش:مقایسه ی عدد با صفر،اگر عدد از 0 کوچکتر باشد،عدد منفی است و اگر بزرگتر باشد،عدد مثبت است.

ورودی:یک عدد **خروجی:**پیغام مناسب

مثال5-2:الگوریتمی بنویسید که 3 عدد را دریافت و معین کند که آیا می توان با این سه عدد یک مثلث ساخت یا خیر؟(با توجه به اینکه شرایط لازم برای تشکیل یک مثلث با سه مقدارa,b,c به عنوان اضلاع،آن است که a<=b+c,b<=a+c,c<=a+b باشد،آن ها را در الگوریتم جایگذاری می کنیم- قسمت های زیر را خودتان تکمیل کنید)

هدف:..... روش:.....

ورودی:..... خروجی:.....

0- شروع
1- a,b,c را دریافت کن
2- اگر a<=b+c برو به خط 3 در غیر اینصورت برو به خط 6
3- اگر b<=a+c برو به خط 4 در غیر اینصورت برو به خط 6
4- اگر c<=a+b آنگاه برو به خط 5 در غیر اینصورت برو به خط 6
5- بنویس "این مقادیر می توانند یک مثلث را تشکیل دهند"
6- بنویس"این مقادیر نمی توانند،تشکیل یک مثلث بدهند"
7- پایان

تمرین: جدول تست الگوریتم فوق را با داده های فرضی توسط خودتان رسم کنید.

سعی کنید الگوریتم مربوط به مثال های بعد را خودتان بنویسید،سپس آن را با جواب موجود مقایسه کنید.

مثال 6-2: الگوریتمی بنویسید که سن شخص(مثلا 10) را از ورودی دریافت و مشخص کند چند ماه و تقریبا چند روز است؟(هر سال 12 ماه یا 365 روز است).

0- شروع

1- Ageرا دریافت کن

2- month←age*12

3- day←age*365

4- بنویس month,day

5- پایان

خط 1	خط 2	خط 3	خط 4
age	**month**	**day**	**خروجي**
10	120	3650	120,3650
24			

تمرین:سطر دوم جدول فوق را با عدد24 به عنوان ورودی در خط 1 و سایر دستورات در ادامه،کامل نمایید.

تمرین:الگوریتمی بنویسد که مقدارصحیحی را از ورودی به عنوان تعداد ساعات گذشته از یک شبانه روز(مثلا 13) دریافت و مشخص کند شامل چند دقیقه و چند ثانیه است؟

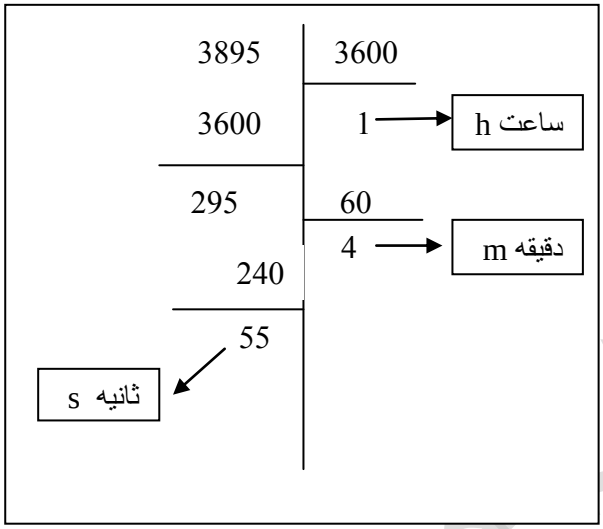
تمرین:الگوریتمی بنویسید که مربع و مکعب عدد دریافت شده از ورودی را در خروجی نشان دهد.(مثلا برای عدد 2 مربع برابر با 4 و مکعب برابر 8 شود)

جدول Trace تمرینات قبل را با داده های فرضی توسط خودتان رسم کنید.

مثال7-2:الگوریتمی بنویسید که مقداری را از ورودی به ثانیه دریافت،آن را به ساعت،دقیقه و ثانیه تبدیل نماید(برای مثال 3895،یک ساعت و 4 دقیقه و 55 ثانیه است)

هدف:الگوریتم مشخص کند،ثانیه ی دریافت شده شامل چند ساعت و دقیقه و ثانیه است(ساعت،دقایقی که به اندازه ی یک ساعت نشود و تعداد ثانیه های باقیمانده ی نهایی که به یک دقیقه نرسد)

روش:هر ساعت 3600 ثانیه و هر دقیقه 60 ثانیه است،پس:عدد دریافت شده را بر 3600 تقسیم صحیح می کنیم تا تعداد ساعت مشخص شود،باقیمانده تعداد ثانیه هایی است که به اندازه یک ساعت نمی رسد ولی ممکن است،برای یک یا چند دقیقه کافی باشد.از آنجا که هر دقیقه 60 ثانیه است،باقیمانده ی تقسیم قبل را بر عدد 60 تقسیم می کنیم و مقدار صحیح تقسیم، تعداد دقایق را مشخص می کند.باقیمانده ی این تقسیم،مسلما مقداری کمتر از 60 است که ثانیه را نشان می دهد.



0- شروع
1- t را دریافت کن
2- h←[t/3600]
3- R←t-([t/3600]*3600) (محاسبه باقیمانده t بر 3600)
4- m←[r/60]
5- s←r-([r/60]*60)
6- h,m,s را بنویس
7- پایان

تمرین:سطر دوم جدول مقابل را با t =7809 کامل کنید.

t	h	r	m	s	خروجی
3895	1	295	4	55	1,4,55
7809					

سوال:درخط پنجم الگوریتم فوق به جای [r/60] از نام کدام متغیر می توان استفاده کرد؟(متغیری که مقدار[r/60] در آن نگهداری می شود،کدام است؟)

تمرین:الگوریتمی بنویسید که مقداری را به عنوان تعداد روز دریافت کرده،مشخص کند این تعداد روز، قابل تجزیه به چند سال و ماه و چند روز است؟

تمرین:جدول Trace را برای الگوریتم زیر رسم نموده و آن را سه بار و با داده های زیر تست کنید.

```
0- شروع
1- A,b,c را دریافت کن
2- max←a
3- اگر b>max
      آنگاه max←b
4- اگر c>max      آنگاه
      max←c
5- Max را بنویس
6- پایان
```

```
A=8   b=3   c=8
A=11  b=15  c=2
A=20  b=10  c=4
```

سوال:بر اساس خروجی جدول خود تعیین کنید که هدف الگوریتم فوق چیست؟

الف)تعیین کوچکترین عدد بین سه عدد ب)تعیین بزرگترین عدد بین سه عدد

ج)تعویض مقادیر درون سه متغیر د)هیچکدام

اگر خطوط 2 و3 را در الگوریتم فوق به صورت زیر بنویسیم،آیا خروجی الگوریتم تغییر خواهد کرد؟

2- max←b

3- اگر a >max آنگاه max←a

اگر الگوریتم را با دستورات جدید به جای خطوط 2و3 قبلی و با همان داده ها تست کنید،خواهید دید خروجی تغییر نخواهد کرد.در واقع این الگوریتم بزرگترین مقدار عددی را بین سه عدد دریافت شده از ورودی پیدا کرده و نمایش می دهد،برای این کار از متغیر کمکی به نام Max استفاده شده که وظیفه ی نگهداری بزرگترین عدد را دارد.در واقع اگر مقداری از max بزرگتر باشد ما مقدار max را به عدد بزرگتر تغییر می دهیم،مثلا در خط 3:اگر b از max بزرگتر باشد،مقدار b را در max بریز،خط 4 نیز به همین صورت است.

چرا در خط دوم مقدارa را درون max ریخته ایم؟به دو دلیل:1- همانطور که قبلاً گفتیم متغیرهایی هستند که ما بر اساس نیاز خود در الگوریتم استفاده می کنیم،مقدار این متغیرها از ورودی دریافت نمی گردد،بلکه معمولا برای نگهداری حاصل محاسبات یا به منظور نگهداری مقادیر خاص استفاده می شود،که به آنها متغیرهای کمکی گفته می شود.**اگر در خطی از الگوریتم از این متغیرها استفاده شود به طوری که محاسبه یا مقایسه ای روی مقدار درون آن انجام شود،باید حتماً از قبل مقداری به آن اختصاص داده باشیم**،در غیر اینصورت،روند الگوریتم با مشکل مواجه خواهد شد،برای مثال اگر ما در خط2، max را برابر باa قرار ندهیم،در خط سوم،b با چه مقداری مقایسه گردد؟(مقدار b از ورودی دریافت شده،در حالی که max را از ورودی دریافت نکرده ایم)

2- فهمیدیم که max حتما باید یک مقدار اولیه داشته باشد،این عدد چگونه تعیین می گردد؟مقدار اولیه ی متغیرهای کمکی باید طوری تعیین شود که موجب اخلال در روند الگوریتم نگردد،در واقع این شما هستید که با توجه به روشی که برای خود در نظر گرفته اید،این مقدار را تعیین می کنید.گاهی تغییر آن به مقادیر مختلف نیز امکانپذیر است،برای مثال دیدید که با تغییر خطوط 2و3 و مقداردهی اولیه max به متغیر b نیز الگوریتم به درستی عمل می کند.**به طور کلی می توان گفت الگوریتم قبل،روش زیر را برای پیدا کردن بزرگترین عدد بین سه عدد دریافتی انتخاب کرده است:**

متغیری را به نام max به منظور نگهداری بزرگترین عدد استفاده کرده و اولین عدد را بزرگترن عدد فرض کرده است(خط 2)،آن را با عدد دوم مقایسه می کند و اگر عدد دوم(b)از Max(با مقدار اولیه ی a)بزرگتر باشد،پس b تا اینجا بزرگترین عدد است،Max را برابر با b قرار بده(خط 3)،خط 4 نیز به طور مشابه انجام می گردد.می بینید که در خط سوم مقایسه ی max با b معادل مقایسه a با b است.

تمرین:الگوریتمی بنویسید که کوچکترین مقدار را بین سه عدد دریافت شده از ورودی پیدا کرده و در خروجی نمایش دهد.

تمرین:برای صورت مسئله های زیر،روش حل مناسب را تشریح کنید(نیازی به نوشتن الگوریتم نیست).

الف)الگوریتمی که اعداد زوج بین 1 تا1000 را تولید و در خروجی نمایش دهد:

2,4,6,8,10,12,14,…,1000

ب)الگوریتمی بنویسید که فاکتوریل یک عدد صحیح را محاسبه کند:

5!=5*4*3*2*1

ج)الگوریتمی که حاصل جمع سری زیر را محاسبه کند(سری فیبوناچی)

1+1+2+3+5+8+13+…

ایجاد حلقه در یک الگوریتم

در تمرینات انتهایی فصل قبل،از شما خواسته شد که روشی برای حل صورت مسئله ها بیابید.در واقع نوشتن الگوریتم تمرینات فصل قبل نیاز به آشنایی با کاربرد الگوریتم ها با ایجاد حلقه و انجام دستورات تکرار دارد،که در این فصل با نوشتن چنین الگوریتم هایی آشنا خواهید شد.ضمناً توصیه می کنیم،حین مطالعه ی مثال ها،**کاربرد انواع متغیرهای کمکی مانند شمارنده،نگهدارنده مجموع محاسبات و... را به خاطر بسپارید.**

مثالی در دنیای واقعی:

اگر مجبور باشید،صد رونوشت از متنی تهیه کنید،آیا شروع به نوشتن دستی متن مربوطه به تعداد صد بار می کنید؟مسلماً نه،بلکه آن را در یک دستگاه کپی قرار داده،تعداد مورد نظر خود را روی آن تنظیم می کنید و دکمه ی مربوط به شروع کار دستگاه را فشار داده تا پس از چند دقیقه انتظار،تعداد مورد نیاز از رونوشت مربوطه برایتان حاضر شود.با این کار در وقت و انرژی شما به میزان قابل توجهی صرفه جویی می گردد.

انجام این کار،دقیقاً مشابه کاربرد حلقه به منظور انجام دستورات تکراری در الگوریتم ها وا زبان های برنامه نویسی است.

چرا در یک الگوریتم یا برنامه از حلقه استفاده می کنیم؟

این کار،اولاً از دوباره نویسی دستورات جلوگیری می کند،ثانیاً باعث صرفه جویی در زمان می شود.ثالثاً در زبان های برنامه نویسی، می تواند از اشغال بیهوده ی حافظه جلوگیری کند.

در واقع ما زمانی حلقه ایجاد می کنیم که می خواهیم دستور یا دستورات خاصی در برنامه به تعداد معلوم یا نامعلوم،عیناً تکرار شوند.

مثال1-3:الگوریتمی بنویسید که 10 عدد از ورودی گرفته و مجموع آنها را محاسبه کند:

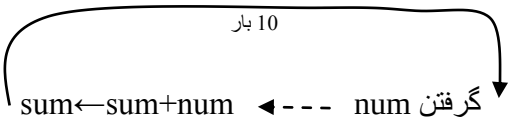
سوال:به نظر شما آیا منطقی است که ما 10 خط دستور جداگانه برای گرفتن ورودی(هر بار یک عدد جدید) بنویسیم و10 متغیر مختلف تعریف کنیم تا در پایان آن متغیرها را باهم جمع کنیم؟اگر به جای ده عدد در صورت مسئله جمع 500 یا هزار عدد یا بیشتر خواسته می شد،آیا راه حل فوق عملی بود؟آیا راه حل دیگری وجود دارد؟

در اینجا می خواهیم الگوریتم مربوطه را با استفاده از ایجاد حلقه بنویسیم:

هدف:محاسبه ی مجموع ده عدد دلخواه

روش حل مسئله:دستور گرفتن عدد،باید 10 مرتبه تکرار گردد.پس از گرفتن هر عدد و جمع آن با مجموع اعداد قبل،دیگر نیازی به نگهداری عدد دریافت شده در متغیر مربوطه نیست،بنابراین ما می توانیم از یک متغیر استفاده کنیم که هر بار داده ی جدیدی که از ورودی گرفته شده در آن قرار می گیرد.برای نگهداری مجموع اعداد قبل با اعداد جدید نیز به متغیری احتیاج داریم که هر بار با مقدار قبلی خودش و ورودی جدید جمع گردد.به طوری که اگر متغیر ورودی num و متغیر نگهدارنده ی مجموع sum باشد ،روال اجرای الگوریتم باید به صورت زیر باشد:

گرفتن num ◄– – (sum←sum+num) ◄– – گرفتن num ◄– – (sum←sum+num). ◄– – .....

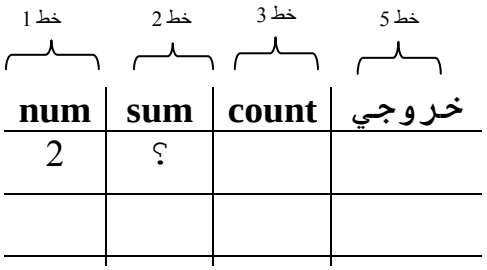


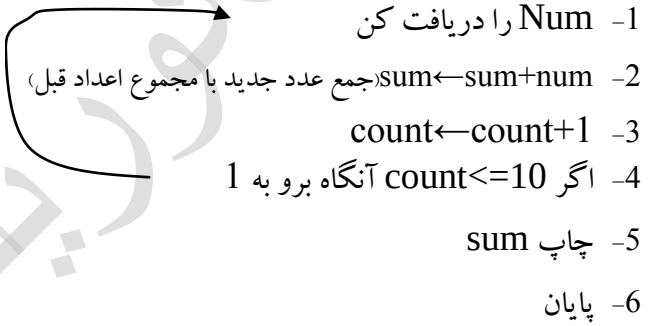
همان طور که مشاهده می کنید،این دو دستور باید به تعداد ده بار در الگوریتم تکرار شوند،حال که روش حل مسئله را فهمیدیم،باید نحوه ی اضافه کردن سایر دستورات را نیز برای کامل کردن الگوریتم مشخص کنیم.

سوال:چگونه در الگوریتم،تعداد تکرار را مشخص کنیم؟در این مثال دستورات تکراری باید ده بار انجام شود،نه کمتر و نه بیشتر.

مهم:به طور کلی وقتی تعداد تکرار دستورات مشخص است،حلقه ای با **تعداد تکرار معین** ایجاد می کنیم و باید از متغیری به عنوان **شمارنده** استفاده کنیم،که پس از هر بار اجرای دستوراتِ مورد نیاز برای تکرار،یکی به مقدار آن اضافه شود.(افزایش مقدار شمارنده جزءِ دستورات درون حلقه خواهد بود).پس از هر بار اجرای دستورات تکراری،مقدار شمارنده چک می شود که آیا به تعداد مورد نظر(در اینجا 10 بار)رسیده ایم یا خیر؟اگر شرط برقرار نبود پس تکرار باید صورت گیرد و اجرای الگوریتم به خط مربوط به شروع حلقه ارجاع داده می شود و اگر شرط برقرار باشد،یعنی نیازی به تکرار دستورات مربوطه نیست،روند اجرای الگوریتم از حالت حلقه خارج شده و به خط بعد از دستورات تکرار می رود.

قبل از نوشتن الگوریتم اصلی،الگوریتم زیر را که برای مثال مجموع اعداد نوشتیم تست می کنیم:





اگرچه روند کلی الگوریتم در نگاه اول،درست به نظر می رسد،اما همانطور که در جدول می بینیم،در خط دوم الگوریتم،دستور مربوطه مبهم است چرا که در عبارت سمت راست،خواسته شده که مقدار درون متغیر num با مقدار درون متغیر sum جمع شود،در حالی که هیچ مقدار قبلی در sum وجود ندارد و از ورودی نیز دریافت نشده است.و این ابهام بوجود می آید که برای مثال،اگر ورودی فرضی در خط 1،عدد 2 باشد،در خط دوم،2 با چه عددی جمع شود!خط سوم نیز به همین صورت است چرا که مقدار قبلی درون count وجود ندارد تا با عدد 1 جمع شود.در چنین مواردی ما باید قبل از استفاده از متغیرها،حتما آنها را مقداردهی اولیه کنیم،اما چگونه؟

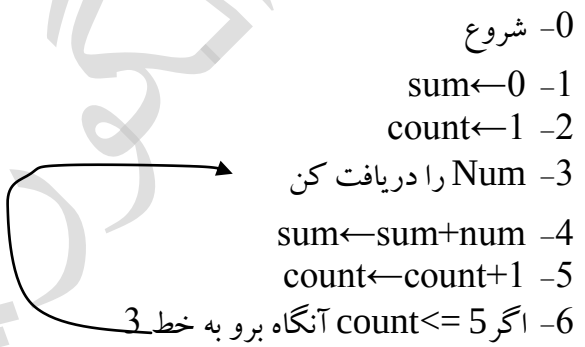
مقداردهی اولیه متغیرها:

قانون ثابتی برای مقداردهی اولیه یک متغیر وجود ندارد،بلکه مقداردهی اولیه ی یک متغیر،با توجه به کاربرد آن در اجرای الگوریتم یا برنامه مشخص می شود.اگرچه ما در مثال های مربوط به فصول قبل نیز این کار را انجام داده ایم،اما استفاده از این مفهوم در اینجا صورت دیگری از کاربرد مقدار دهی اولیه در مواردی است که عملیات محاسباتی روی خود متغیر نیز انجام شده و حاصل به دست آمده نیز در همان متغیر ریخته می شود.مثلا مقدار Count باید با 1 جمع شده و حاصل مجدداً در count قرار گیرد،به طوری که پس از اجرای این دستور مقدار درون count از بین رفته و مقدار جدید

برابر خواهد بود با مقدار قبلی باضافه ی یک(مثلا اگر Count آخرین بار برابر با 2 بوده باشد،در خط سوم 2 با 1 جمع شده،مقدار 2 از بین رفته و از این به بعد count برابر با 3 خواهد بود)

مقداردهی اولیه متغیر sum: گفتیم که از این متغیر برای نگهداری مجموع اعداد دریافت شده از ورودی استفاده می شود.دراینجا باید sum را طوری مقدار دهی اولیه کنیم که تاثیری بر حاصل مجموع کل اعداد و همچنین جمع با اولین عدد دریافتی از ورودی نداشته باشد.یعنی عددی که در محاسبه ی مجموع اعداد خنثی باشد،همانطور که می دانید این عدد 0 است(مشابه 1 برای عمل ضرب).

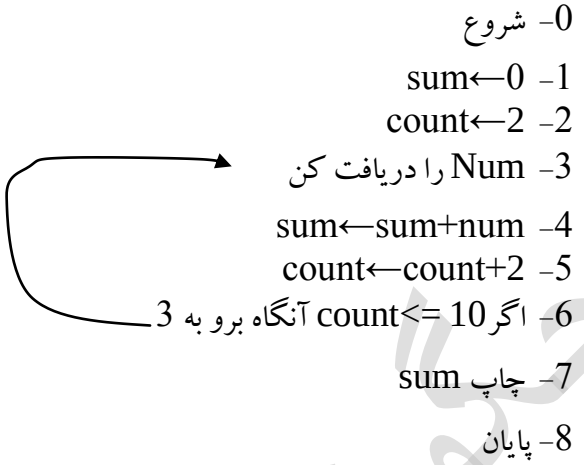
مقداردهی اولیه count: از count برای شمارش تعداد تکرار دستورات حلقه استفاده کرده ایم یعنی count در الگوریتم نقش شمارنده را بازی می کند،بر خلاف Sum مقداردهی اولیه countاهمیت چندانی ندارد و با توجه به طرز نوشتن دستور مربوط به افزایش شمارنده حلقه و همچنین شرط پایان حلقه می تواند هر مقداری باشد،در مثال های بعد انواع مختلفی را برای مقداردهی count نشان می دهیم.در اینجا الگوریتم را برای count با مقدار اولیه 1 نوشته و تست می کنیم.ضمنا برای کاهش زمان تست،نیز الگوریتم را به جای محاسبه ی مجموع ده عدد برای محاسبه ی مجموع 5 عدد می نویسیم.(داده های فرضی:3,6,9,20,4)



	خط 3 num	خط 4 sum	خط 5 count	خط7 خروجي
خطوط 1و2		0	1	
تکرار 1	3	3	2	
تکرار2	6	9	3	
تکرار3	9	18	4	
تکرار 4	20	38	5	
تکرار5	4	42	6	42

همانطور که می بینید پس از پنجمین تکرار حلقه(جدول) و برابر شدن مقدار count با عدد 6،در خط ششم الگوریتم (اگر countکوچکتر یا برابر با 5 بود آنگاه برو به خط 3)،شرط مربوطه برقرار نیست و Count که اکنون برابر با عدد 6 است از 5 بزرگتر بوده،بنابراین روند اجرا به خط 7 و پس از آن به پایان می رود.

تمرین1-3:جدول تست الگوریتم زیر را با داده های فرضی مثال قبل کامل کنید.



num	sum	count	خروجي

خط 3 خط 4 خط 5

سوال:آیا خروجی به دست آمده با آنچه که در مثال قبل به دست آمد، برابر است؟چرا تغییر مقدار اولیه ی count به عدد 2 تغییری بر روند اجرای الگوریتم به منظور محاسبه ی مجموع اعداد نداده است؟

تمرین2-3:شرط پایان حلقه را در الگوریتم تمرین 1-3 را طوری تغییر دهید که با تغییر مقدار اولیه ی count به عدد 4 و افزایش دوتا دوتای شمارنده،خروجی با همان داده های فرضی برابر با 42 شود.الگوریتم خود را حتما تست کنید.

اگر مثال قبل و مفاهیم گفته شده را کاملاً درک کرده باشید،در فهم مثال های بعد و انجام تمرینات مربوطه مشکلی نخواهید داشت.کافیست ذهن خود را برای تعیین روش حل مسائل تقویت کنید که این امر نیاز دارد تا پس از مطالعه ی مثال ها،با تمرین و ممارست،خودتان نیز قادر به نوشتن آنها باشید.پس یکبار دیگر از درک مفهوم **حلقه،شمارنده و مقداردهی اولیه** درمورد مثال قبل مطمئن شوید.

(!)همانطور که تا کنون متوجه شدید،برای نوشتن یک الگوریتم منحصراً یک روش یا یک ترتیب دستوری وجود ندارد و ممکن است روش های مختلف یا ترتیب های متفاوتی برای خطوط الگوریتم مربوط به یک صورت مسئله وجود داشته باشد که از لحاظ عملکرد و دقت کاملاً مشابه باشند که درکتب مختلف نیز وجود دارند ولی این امر نباید موجب شود به درستی الگوریتم تست شده ی خود شک کنید،فقط سعی کنید از درستی الگوریتم در بدترین شرایط احتمالی با داده های فرضی مربوطه مطمئن شوید.

مثال 2-3:الگوریتمی بنویسید که اعداد زوج بین 1 تا 20 را در خروجی نمایش دهد.

2،4،6،8،10،...،18،20

روش حل مسئله:اعداد زوج 1 تا 20 شامل سری فوق است،با مقداردهی اولیه ی یک متغیر با عدد 0 (m←0)و پس از آن افزایش دوتایی آن تا رسیدن به 20،می توان سری را تولید کرد.(m←m+2)

m	M+2
0	2
2	4
4	6
6	...

توجه:برای نوشتن این الگوریتم نیاز به دریافت هیچ داده ای از ورودی نداریم و نوشتن الگوریتم خواسته شده نیاز به اعمال محاسبات روی هیچ داده ی ورودی ندارد و فقط باید با استفاده از روش مربوطه به منظور تولید سری به هدف رسید.

دستورات حلقه چه تعداد باید تکرار شود؟دستور تکراری مربوط به روش حل مسئله m←m+2 است،گفتیم که در حلقه ها با تعداد تکرار معین نیاز به شمارنده ای داریم تا بتوانیم با چک کردن مقدار آن پس از هر بار انجام دستورات تکراری،تصمیم به اجرای مجدد حلقه یا اتمام آن بگیریم.از آنجا که ما نصف اعداد بین 1 تا 20 (اعداد زوج)را در سری تولید خواهیم کرد،پس تعداد تکرار نیز نصف عدد پایانی(20) یعنی تا 10 بار خواهد بود:

0- شروع
1- m←0
2- count←0
3- m←m+2
4- count←count+1
5- بنویس m
6- اگر count<10 برو به خط 3
7- پایان

m	count	خروجي
0	0	
2	1	2
4	2	4
6	3	6
000		
16	8	16
18	9	18
20	10	20

دستورات خط 3 تا 6

تکرار دستورات خط 3 تا 6

با رسیدن شمارنده به عدد 9 در خط 6،روند اجرا به خط 3 ارجاع داده می شود و این بار در خط 4 حلقه،شمارنده برابر با 10 می شود، بنابراین پس از چک شدن مجدد مقدار count در خط 6،از آنجا که اکنون count برابر با 10 بوده و کوچکتر نیست،اجرا به خط 7 می رود.

اصلاح الگوریتم قبل:همانطور که در جدول trace می بینید الگوریتم نوشته شده به درستی عمل کرده و در هر بار تکرارِدستورات در حلقه ی ایجاد شده،عددی جدید از سری اعداد مربوطه تولید و چاپ می شود.می خواهیم با اعمال تغییراتی،دستورات الگوریتم را با حذف متغیر شمارنده count و استفاده از m در شرط پایان حلقه کاهش دهیم.به طور کلی با کمی خلاقیت و دقت و پس از مدتی تمرین،قادر خواهید بود در برخی الگوریتم ها متغیرهایی را به صورت چند منظوره در الگوریتم استفاده کنید،بدون آنکه خللی در روند اجرای الگوریتم ایجاد شود.

0- شروع
1- m←0
2- m←m+2
3- بنویس m
4- اگر m<20 برو به خط 2
5- پایان

تمرین 3-3: جدول trace مربوط به الگوریتم اصلاح شده ی مثال 2-3 را رسم کنید.

تمرین 4-3: الگوریتمی بنویسید که اعداد بخشپذیر بر 5 را بین 0 تا 50 نمایش دهد.
5،10،15،20،...،40،45
سپس جدول trace آن را رسم نمایید.

مثال 3-3: الگوریتمی بنویسید که میانگین 15 نمره ی دریافتی از ورودی را محاسبه و نمایش دهد.

روش: استفاده از حلقه ای که دستورات مربوط به گرفتن نمره جدید و جمع آن با مجموع نمرات قبلی در آن محاسبه شود.دستور افزایش مقدار شمارنده نیز باید جز حلقه باشد.نهایتاً،مجموع اعداد بر تعداد آنها یعنی 15 تقسیم گردد.

گرفتن نمره جدید ← مجموع نمرات+نمره جدید ← مجموع نمرات ← افزایش تعداد شمارنده حلقه

در اینجا،برای جلوگیری از طولانی شدن جدول ،الگوریتم خود را برای محاسبه میانگین 5 نمره ی فرضی 17و19و20و15و10 نوشته ایم،با تغییر شرط در خط ششم و دستور خط 7 می توان آن را به

0- شروع
1- Sum ←0
2- Count←0
3- Score را دریافت کن
4- Sum←Sum+Score
5- Count←Count+1
6- اگر Count<5 آنگاه برو به خط 3
7- Avg←Sum/5
8- چاپ کن Avg
9- پایان

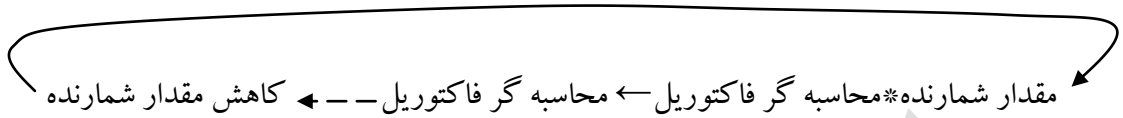
Score	Sum	Count	Avg	خروجی
	0	0		
10	10	1		
15	25	2		
20	45	3		
19	64	4		
17	81	5	16.2	16.2

تکرار دستورات خط 3 تا 6

تمرین 5-3: الگوریتم فوق را با داده های فرضی 19/5و18و17/25و16/25و20 تست کنید.

مثال 4-3: الگوریتمی بنویسید که فاکتوریل عدد دریافتی از ورودی را محاسبه کرده و نمایش دهد.(5!=5*4*3*2*1)

روش حل:نوشتن الگوریتم محاسبه ی فاکتوریل روش های مختلفی دارد،روش انتخابی ما بدین صورت است که فاکتوریل با ضرب عدد در یک مقدار کمتر از خودش تا رسیدن به یک محاسبه شود.برای این منظور از متغیر شمارنده ی حلقه ی خود به صورت کاهشی استفاده خواهیم کرد،به طوری که **مقدار اولیه** ی آن برابر با عدد دریافتی خواهد بود و تعداد تکرار دستورات مربوطه تا رسیدن مقدار شمارنده به صفر است. m*(m-1)*(m-2)*(m-3)*…*1



0- شروع
1- m را دریافت کن(عدد دریافتی که الگوریتم،فاکتوریل آن را محاسبه خواهد کرد)
2- fact←1(متغیر محاسبه ی فاکتوریل با مقدار اولیه خنثی در ضرب،یعنی 1)
3- count←m(شمارنده نزولی با مقدار اولیه ی متغیر دریافتی)
4- fact←fact*count
5- count←count-1
6- اگر count>1آنگاه برو به خط 4
7- Fact را بنویس
8- پایان

m	fact	count	خروجی
5	1	5	
	5	4	
	20	3	
	60	2	
	120	1	120

- فلش مربوط به دستوراتی که در الگوریتم ایجاد حلقه می کند را خودتان رسم کنید.
- سطرهای مربوط به انجام دستورات تکراری است را در جدول trace مشخص کنید.

تمرین 6-3:الگوریتم فوق را با داده فرضی n=6 در جدول تست کنید.

تمرین7-3:الگوریتم محاسبه ی فاکتوریل را به روشی بنویسید که در آن از شمارنده ای به صورت افزایشی استفاده شود.(راهنمایی:5*4*3*2*1=!5)- الگوریتم خود را با مقدار 5 به عنوان ورودی تست کرده و خروجی خود را با جدول trace مثال 4-3 مقایسه کنید.

تمرین8-3:الگوریتمی بنویسید که مربع اعداد 1 تا 10 را نمایش دهد.جدول trace مربوطه را رسم کنید.

مثال5-3:الگوریتمی بنویسید که عدد طبیعی n را از ورودی دریافت و سری زیر را تولید و در خروجی نمایش دهد.

1/2+1/4+…1/n

0- شروع
1- n را دریافت کن
2- S←0
3- count←2
4- s←s+1/count
5- count←count+2
6- اگر count<=n بروبه خط 4
7- S را بنویس

n	s	count	خروجی
6	0	2	
	0.5	4	
	0.75	6	
	0.94	8	0.94

8- پایان

ایجاد حلقه با تعداد تکرار نامعین

در مثال های قبل،تعداد تکرار دستورات مربوطه برای ایجاد یک حلقه،با توجه به صورت مسئله یا انتخاب روش نوشتن الگوریتم،قابل تشخیص بود.اما گاهی اوقات تعداد تکرار مورد نیاز،در صورت مسئله مشخص نیست،در روش انتخاب شده برای نوشتن الگوریتم نیز،مقدار عددی خاصی قابل تعیین نیست.

مثالی در دنیای واقعی:

می خواهیم مهره هایی را با اندازه های مختلف درون یک کیسه قرار دهیم(اینکه از هر سایز چه تعداد در کیسه قرار بگیرد مهم نیست)،اما کیسه ظرفیت تمام این مهره ها را ندارد و شما تصمیم می گیرید هر تعداد مهره را با هر اندازه ای داخل کیسه بیندازید،تا کیسه پر شود.آیا در این صورت مشخص خواهد بود که چه تعداد مهره در چه سایزهایی ظرفیت کیسه را تکمیل خواهد کرد؟خیر.شما برای درخواست این کار از فرد دیگر به او خواهید گفت:تا جایی که کیسه جا دارد،درون آن مهره قرار بده.**تا جایی که کیسه جا دارد**،در واقع شرطی را بیان می کند که در صورت برقراری آن(جا داشتن کیسه) فرد باید مهره ها را در کیسه قرار دهد(تکرار عملیات گذاشتن مهره در کیسه)

پس،در حلقه های با تعداد تکرار نامشخص،شرطی وجود دارد که برقراری آن شرط موجب تکرار دستورات مربوطه می گردد و اگر شرط برقرار نباشد،تکرار دستورات خاتمه می پذیرد.

(زمان نوشتن الگوریتم با تعداد تکرار معین نیز شرط پایان وجود دارد که عملکرد آن مشابه شرط حلقه با تعداد تکرار نامعین است،این تقسیم بندی از انواع حلقه،بیشتر به خاطر وجود دستورات متفاوت ایجاد حلقه در زبان های برنامه نویسی و فراگیری راحت تر کاربرد حلقه هاست)

مثال6-3:الگوریتمی بنویسید که تعداد نامشخصی عدد را از ورودی دریافت و حاصلجمع اعداد دریافت شده از ورودی را محاسبه کند.اگر عدد دریافت شده 0 بود،عملیات محاسبه حاصل جمع پایان پذیرد.(الگوریتم خود را با داده های فرضی به ترتیب 0و2و5و3و1 تست کنید)

روش نوشتن الگوریتم:استفاده از حلقه ای که شرط آن مقدار ورودی را چک کند،اگر مقدار دریافتی مخالف صفر بود ،آن را با مجموع اعداد قبلی جمع کند(اولین بار با صفر جمع می شود)و اگر 0 باشد،عملیات خاتمه پذیرد.

0- شروع
1- sum←0
2- num←0 (مقداردهی اولیه Num با صفر برای رفع ابهام خط 3)
3- sum←sum+num
4- num را دریافت کن
5- اگر num≠0 آنگاه برو به خط 3
6- Sum را بنویس
7- پایان

خروجی	num	sum
	0	0
	1	0
	3	1
	5	4
	2	9
11	0	11

- اگر ترتیب دیگری برای نوشتن الگوریتم فوق به ذهنتان رسید،آن را با داده های فرضی ذکر شده در مثال تست کنید،در صورتی که خروجی تولید شده،همان بود،الگوریتم شما درست است.

مثال3-7:الگوریتم مثال قبل را طوری تغییر دهید که علاوه بر مجموع اعداد دریافتی،تعداد آنها نیز نمایش داده شود.

روش:کافیست شمارنده ی افزایشی ای به دستورات مربوط به حلقه اضافه کنیم،که پس از وارد شدن هر عدد از ورودی،یکی به مقدار آن اضافه شود.

0- شروع
1- sum←0
2- num←0
3- count←0(شمارنده ی تعداد اعداد دریافتی از ورودی)
4- sum←sum+num
5- num را دریافت کن.
6- count←count+1
7- اگر Num≠0 آنگاه برو به خط 4
8- چاپ کن sum,count
9- پایان

تمرین1:جدول تست مربوط به این الگوریتم را با داده ها ی فرضی قبل رسم کنید.

تمرین2:دستوری به الگوریتم اضافه کنید که صفر را جز اعداد ورودی حساب نکند.(مقدار count زمان نمایش در خروجی یکی کاهش یابد)

مثال3-8:الگوریتمی بنویسید که اعداد زوج دو رقمی را نمایش دهد.

روش نوشتن الگوریتم:در فصل قبل، الگوریتم مربوطه را پس از محاسبه ی ذهنی تعداد تکرار حلقه(با تعداد تکرار معین)نوشتیم،حال می خواهیم الگوریتم را تنها با استفاده از یک متغیر(تولید کننده ی اعداد زوج)تا زمان رسیدن به عدد100 بنویسم.این اعداد شامل 2,4,6,…,98 می باشد.متغیر خود را با عدد 2 مقداردهی اولیه کرده،در حلقه ی مربوطه آن را چاپ و دوتا دوتا اضافه می کنیم.

0- شروع
1- num←2
2- چاپ num
3- num←num+2
4- اگر num<100 آنگاه برو به خط 2
5- پایان

خروجی	num
2	2
4	4
6	6
...	...
96	96
98	98
	100

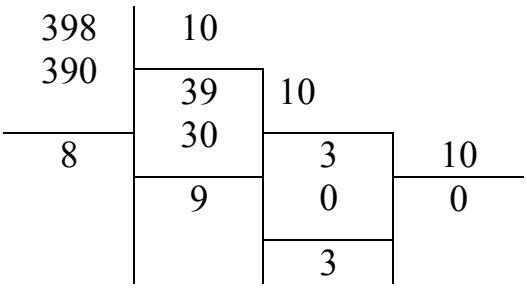
تمرین:اگر در نوشتن الگوریتم مثال قبل، num را با عدد 0(خط 1)،مقداردهی اولیه کنیم،ترتیب دستورات و شرط پایان حلقه را به چه صورت تغییر دهیم تا عملکرد الگوریتم برای رسیدن به هدف مورد نظرهمچنان صحیح باقی بماند؟جدول تست الگوریتم خود را رسم نمایید.

تمرین:الگوریتمی بنویسید که تعداد نامحدودی عدد طبیعی را از ورودی دریافت کرده و میانگین آنها را محاسبه نماید.با وارد شدن عدد صفر،عملیات دریافت ورودی پایان پذیرد.(راهنمایی:مراجعه به مثال های 3-3 و 7-3 و اینکه،نیاز به شمارنده ای داریم که تعداد اعداد دریافتی از ورودی را محاسبه نماید تا پس ازاتمام دریافت ورودی ها با وارد شدن عدد صفر،بتوان میانگین را از طریق تقسیم مجموع اعداد دریافتی بر تعداد آنها محاسبه نمود)
جدول تست الگوریتم خود را با داده های فرضی 5,4,2,3,6,0 رسم نمایید.(خروجی باید برابر با عدد 4 شود)،عدد صفر در محاسبه ی میانگین در نظر گرفته نمی شود و فقط برای اتمام حلقه از ورودی گرفته می شود.

تمرین:الگوریتمی بنویسید که حاصلضرب سری مقابل را محاسبه نماید. 1*3*5*7*...*15

تمرین:الگوریتمی بنویسید که تعدادی عدد طبیعی یک یا دو رقمی را از ورودی دریافت و مربع آنها را محاسبه کند.(اگر عدد از 99 بزرگتر باشد،حلقه پایان یابد)

مثال 9-3:الگوریتمی بنویسید که عدد صحیحی را دریافت و مقلوب آن را در خروجی نمایش دهد.
روش نوشتن الگوریتم:روش خود را با ذکر یک مثال تشریح می کنیم.مقلوب 398 برابر با 893 است،اما چگونه می توان شیوه ای را برای تبدیل 398 یا هر عدد دیگر به مقلوب خودش تعیین نمود؟به شکل زیر توجه کنید:



می بینید که با تقسیمات صحیح متوالی عدد بر 10،ارقام عدد به صورت تجزیه شده در باقیمانده به دست آمد.قبلا شیوه ی محاسبه ی باقیمانده تقسیم اعداد صحیح را بر یکدیگر آموختیم(مثال 3-2).نکته ای که باید به آن توجه شود،این است که برای مثال اگر عدد دریافتی در متغیر N است،طبق تقسیمات فوق،n باید پس از هر بار تقسیم بر 10 و محاسبه ی باقیمانده به خارج قسمت تقسیم تبدیل گردد تا سایر ارقام نیز به دست آیند.یعنی از 398 به 39 و از 39 به3 و از 3 به0.

0- شروع
1- N را دریافت کن
2- r←n-10*[n/10]
3- r را بنویس
4- n←[n/10]
5- اگر n>0 انگاه برو به خط 2
6- پایان

تمرین: فلش مربوط به اتصال خطوط تکرار را رسم کرده و درستی الگوریتم را با داده ی فرضی 763 به عنوان ورودی در جدول Trace چک کنید.

مثال10-3:الگوریتمی بنویسید که 12 جمله از سری زیر را تولید کند و آنها را در خروجی نمایش دهد.

$$1,1,2,3,5,8,\ldots,a_n,\ldots$$

روش نوشتن الگوریتم:معمولاً در تولید یک سری،رابطه ی ریاضی ای وجود دارد که الگوریتم باید بر اساس آن نوشته شود.اعداد تولید شده پس از 1 ها در این سری،از مجموع دوعدد قبلی تولید شده به دست آمده اند.

1+1= 2 1+2= 3 2+3=5 3+5=8

ما نیاز به متغیری برای تولید عدد جدید و متغیرهای به منظور نگهداری دو عدد قبلی برای استفاده ی بعدی داریم.

0- شروع
1- n←3
2- m1←1
3- m2←1
4- m1,m2 را بنویس
5- fibo←m1+m2
6- fibo را بنویس
7- m1←m2
8- m2←fibo
9- n←n+1
10- اگر n<=12 آنگاه برو به خط 5
11- پایان

خروجی	fibo	m2	m1	n
1,1,2	2	1	1	12
3	3	2	1	
5	5	3	2	
8	8	5	3	

تمرین:جدول Trace فوق را تا رسیدن n به عدد 12 کامل کنید.

حلقه های تودرتو

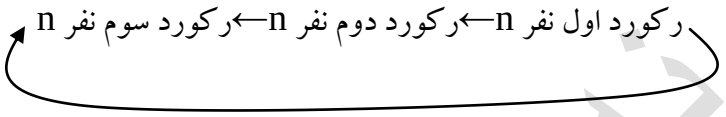
در این مبحث ، به نحوه نوشتن الگوریتم هایی می پردازیم که تعدادی دستور تکراری باید تکرار شوند.یعنی نحوه ایجاد حلقه های تو درتو.

مثالی در دنیای واقعی:

نام	رکورد اول	رکورد دوم	رکورد سوم
علي عباسي	15s	14s	12s
احمد موسوي	15s	15s	13s
رضا اسدي	20s	18s	16s
محمد قنبري	25s	20s	17s
سعيد عسگري	26s	24s	20s

لیست روبرو مربوط به ثبت رکورد توسط 5 ورزشکار به تعداد 3 بار برای هر کدام در مسابقه ی دو می باشد.

به طور کلی عمل ثبت رکورد 15 بار،شامل 3 بار برای هر یک از 5 نفر،می باشد(سطری).



مثال11-3: الگوریتمی بنویسید که جدول ضرب 10*10 را در خروجی نمایش دهد.

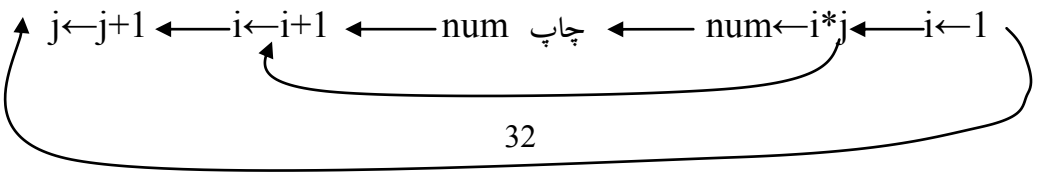
روش نوشتن الگوریتم: همانطور که می دانید،جدول ضرب اعداد 10*10 شامل ضرب یک یک اعداد،درون سطری از 1 تا 10 در ستونی از 1 تا 10 است.

1	2	3	4	…	9	10
2	4	6	8	…	18	20
3	6	9	12	…	27	30
.						
.						
.						
10	20	30	40	…	90	100

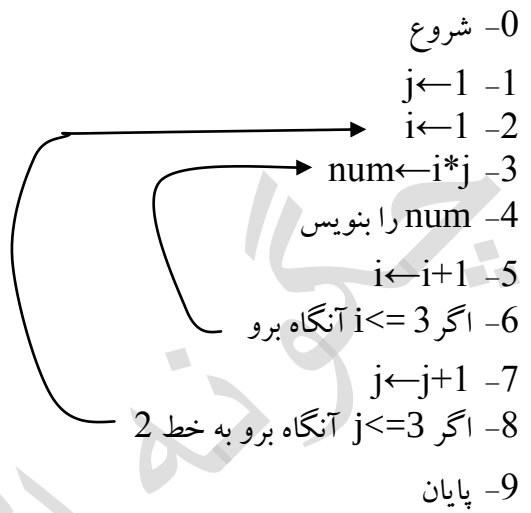
ما نیز باید چنین روندی را در الگوریتم خود ایجاد کنیم یعنی(روش سطری): اولا اعداد 1 تا 10 را به طریقی تولید کنیم ثانیا هر بار که عددی تولید می شود،عدد تولید شده را در اعداد 1 تا 10 ضرب کنیم.و حاصلضربها را در خروجی نمایش دهیم.

چگونه اعداد 1تا 10 را تولید کنیم؟ برای تولید این سری،از شمارنده ای استفاده می کنیم که هر بار یکی به مقدار آن اضافه می شود.در واقع اعداد را در یک حلقه تولید می کنیم.(j←j+1)

چگونه عدد تولید شده را در اعداد 1 تا 10 ضرب کنیم؟عملیات ضرب در اعداد یک تا ده برای هر عدد تولید شده نیز با استفاده از حلقه ی دیگری که اعداد 1 تا 10 را تولید کند،صورت می گیرد(باید مجددا اعداد 1تا 10 را به صورت جداگانه تولید کنیم).به طوری که پس از تولید هر عدد در حلقه ی قبلی،همان عدد در اعداد 1 تا 10(i)تولید شده در یک حلقه ی داخلی ضرب شده و نتیجه در خروجی نشان داده می شود.الگوریتم مجددا به حلقه ی بیرونی ارجاع داده می شود،شمارنده ی حلقه ی داخلی برابر با 1 شده وعدد بعد،نیز در اعداد 1 تا 10 تولید شده در حلقه ی داخلی ضرب می شود.



برای طولانی نشدن جدول trace الگوریتم،در اینجا الگوریتم خود را برای جدول ضرب 3*3 می نویسیم.کافیست در قسمت شرط پایان حلقه ها،عدد 3 را به 10 تغییر دهیم



j	i	num	خروجی
1	1	1	1,2,3
	2	2	
	3	3	
2	1	2	2,4,6
	2	4	
	3	6	
3	1	3	3,6,9
	2	6	
	3	9	
4	پایان حلقه ی بیرونی		

تمرین:الگوریتم جدول ضرب را به صورت n*m بنویسید.(تعداد سطرها n و تعداد ستون ها m از ورودی دریافت شود)

تمرین:الگوریتم خود را با فرض اینکه n=3 و m=4 از ورودی دریافت شده باشد،تست کنید.

مثال12-3:الگوریتمی بنویسید که 5 نمره ی 10 دانش آموز را از ورودی دریافت و میانگین هرکدام را محاسبه و در خروجی نمایش دهد.(توضیحات مربوط به محاسبه ی میانگین قبلاً در مثال های.....شرح داده شده است)

0- شروع
1- sum←0
2- j←1(شمارنده ی تعداد دانشجویان)
3- i←1(شمارنده ی تعداد نمرات دریافت شده برای هر دانشجو که باید برای هر دانشجو از 1 تا 5 بشمرد)
4- score را دریافت کن
5- sum←sum+score
6- i←i+1
7- اگر i<=5 آنگاه برو به خط4
8- avg←sum/5
9- avg را بنویس
10- j←j+1
11-اگر j<=10 آنگاه برو به خط 3
12- پایان

تمرین:فلش مربوط به حلقه ی داخلی و بیرونی مثال 12-3 را رسم کرده،الگوریتم را با داده های فرضی برای 3 دانشجو در جدول trace تست کنید.

مثال:الگوریتمی بنویسید که با ارقام 1و2و3و4و5 خروجی زیر را تولید کند.

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

روش نوشتن الگوریتم:با دقت به ساختار فوق و توجه به رابطه ی شماره ی سطرها و ستون ها در می یابیم که اعداد تولید شده در هر سطر به تعداد شماره ی سطر بوده و با عدد 1 آغاز می شود.برای مثال سطر چهارم شامل اعداد 1 تا 4 است.در واقع شمارنده ی حلقه ی داخلی باید به تعداد مقدار فعلی شمارنده ی بیرونی تکرار و از عدد 1 تا این مقدار،یکی یکی افزایش یابد.

خروجی	i	j
1	1	1
1 2	1 2	2
1 2 3	1 2 3	3

0- شروع
1- j←1
2- i←1
3- i را بنویس
4- i←i+1
5- اگر i=>j آنگاه برو به خط 3
6- j←j+1
7- اگر j=>5 آنگاه برو به خط 2
8- پایان

تمرین:جدول فوق را با جایگذاری مقادیر متغیرها در ادامه ی تست الگوریتم کامل کنید.

نکته:در زبان های برنامه نویسی دستوراتی وجود دارد که امکان رفتن به سطر بعد را برای نمایش خروجی ها فراهم می کند.

تمرین:الگوریتمی بنویسید که فاکتوریل اعداد 1 تا n را در خروجی نمایش دهد.(n از ورودی دریافت شود)