



دانشگاه  
 شهر

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِيْمِ

اصول برنامه نویسی کامپیوتر

# اصول برنامه نویسی به زبان C++

محمد فرشی

بخش علوم کامپیوترا-دانشکده علوم ریاضی-دانشگاه یزد

۱۴۰۳-۱

# زبان برنامه‌نویسی

## زبان برنامه‌نویسی

**(ووش هل مسئله با استفاده از اعمال اساسی را بیان گردیم.**

هدف ما این است که از کامپیوتر به عنوان مجری الگوریتم‌ها استفاده کنیم.

لازم تشریح (ووش برای مجری آن است که یک زبان مشترک بین ما و مجری وجود داشته باشد.

بیان (ووش به زبان کامپیوتر؛ ممکن است اما سفت است.

آموزش زبان انسان به کامپیوتر و استفاده از زبان انسان؛ یکی از شاخصهای هوش مصنوعی اما زبان انسان دقیق نیست و مثلاً دارای سوء تفاهم است.

یک راه حل، استفاده از یک زبان مصنوعی واسط (معروف به زبان برنامه‌نویسی) و استفاده از یک مترجم (معروف با کامپایلر یا مفسر) برای ترجمه از زبان برنامه‌نویسی به زبان ماشین.



دانشگاه  
بهشتی  
دانشگاه علم پژوهی

# زبان برنامه‌نویسی

## زبان برنامه‌نویسی

(و)ش حل مسئله با استفاده از اعمال اساسی را بیان گردیدم.

هدف ما این است که از کامپیوتر به عنوان مجری الگوریتم‌ها استفاده کنیم.

لازم تشریح (و)ش برای مجری آن است که یک زبان مشترک بین ما و مجری وجود داشته باشد.

بیان (و)ش به زبان کامپیوتر؛ ممکن است اما سفت است.

آموزش زبان انسان به کامپیوتر و استفاده از زبان انسان؛ یکی از شاخصهای هوش مصنوعی اما زبان انسان دقیق نیست و مثلاً دارای سوء تفاهم است.

یک راه حل، استفاده از یک زبان مصنوعی واسط (معروف به زبان برنامه‌نویسی) و استفاده از یک مترجم (معروف با کامپایلر یا مفسر) برای ترجمه از زبان برنامه‌نویسی به زبان ماشین.



دانشگاه  
بهشتی  
دانشگاه علم پژوهی

### زبان برنامه‌نویسی

روش حل مسئله با استفاده از اعمال اساسی را بیان کردیم.

هدف ما این است که از کامپیوتر به عنوان مجری الگوریتم‌ها استفاده کنیم.

لازم‌آمد تشریح روشن برای مجری آن است که یک زبان مشترک بین ما و مجری وجود داشته باشد.

بیان روشن به زبان کامپیوتر؛ ممکن است اما سفت است.

آموزش زبان انسان به کامپیوتر و استفاده از زبان انسان؛ یکی از شاخصه‌های هوش مصنوعی اما زبان انسانی دقیق نیست و مثلاً دارای سوء‌تفاهم است.

یک راه حل، استفاده از یک زبان مصنوعی واسط (معروف به زبان برنامه‌نویسی) و استفاده از یک مترجم (معروف با کامپایلر یا مفسر) برای ترجمه از زبان برنامه‌نویسی به زبان ماشین.



دانشکده علم پزشکی  
دانشگاه شهید بهشتی

# زبان برنامه‌نویسی

## زبان برنامه‌نویسی

(و)ش حل مسئله با استفاده از اعمال اساسی را بیان کردیم.

هدف ما این است که از کامپیوتر به عنوان مجری الگوریتم‌ها استفاده کنیم.

لازم تشریح (و)ش برای مجری آن است که یک زبان مشترک بین ما و مجری وجود داشته باشد.

بیان (و)ش به زبان کامپیوتر؛ ممکن است اما سفت است.

آموزش زبان انسان به کامپیوتر و استفاده از زبان انسان: یکی از شاخصهای هوش مصنوعی اما زبان انسان دقیق نیست و مثلاً دارای سوء تفاهم است.

یک راه حل، استفاده از یک زبان مصنوعی واسط (معروف به زبان برنامه‌نویسی) و استفاده از یک مترجم (معروف با کامپایلر یا مفسر) برای ترجمه از زبان برنامه‌نویسی به زبان ماشین.



### زبان برنامه‌نویسی

(و)ش حل مسئله با استفاده از اعمال اساسی را بیان کردیم.

هدف ما این است که از کامپیوتر به عنوان مجری الگوریتم‌ها استفاده کنیم.

لازم است تشریح (و)ش برای مجری آن است که یک زبان مشترک بین ما و مجری وجود داشته باشد.

بیان (و)ش به زبان کامپیوتر؛ ممکن است اما سفت است.

آموزش زبان انسان به کامپیوتر و استفاده از زبان انسان: یکی از شاخه‌های هوش مصنوعی اما زبان انسان دقیق نیست و مثلاً دارای سوء تفاهem است.

یک راه حل، استفاده از یک زبان مصنوعی واسط (معروف به زبان برنامه‌نویسی) و استفاده از یک مترجم (معروف با کامپایلر یا مفسر) برای ترجمه از زبان برنامه‌نویسی به زبان ماشین.



### زبان برنامه‌نویسی

(ووش هل مسئله) با استفاده از اعمال اساسی را بیان گردیم.

هدف ما این است که از کامپیوتر به عنوان مجری الگوریتم‌ها استفاده کنیم.

لازم است تشریح (ووش برای مجری آن است که یک زبان مشترک بین ما و مجری وجود داشته باشد.

بیان (ووش به زبان کامپیوتر؛ ممکن است اما سفت است.

آموزش زبان انسان به کامپیوتر و استفاده از زبان انسان: یکی از شاخه‌های هوش مصنوعی اما زبان انسان دقیق نیست و مثلاً دارای سوء تفاهem است.

یک راه حل، استفاده از یک زبان مصنوعی واسط (معروف به زبان برنامه‌نویسی) و استفاده از یک مترجم (معروف با کامپایلر یا مفسر) برای ترجمه از زبان برنامه‌نویسی به زبان ماشین.



# زبان‌های برنامه‌نویسی

زبان‌های برنامه‌نویسی:

ساختاری مشابه زبان‌های طبیعی دارند.

زبان‌ها دارای مجموعه کاراکترها، مجموعه کلمات و یک مجموعه دستور زبان یا گرامر هستند.

با استفاده از آنها می‌توان جملات مختلف را ساخت و منظور خود را به طرف مقابل (ساند).

در زبان‌های برنامه‌نویسی، تعداد کلمات (که کلمات کلیدی زبان برنامه‌نویسی نامیده می‌شوند) و دستور (زبان یا گرامر بسیار محدود می‌باشد و لذا یادگیری آن به مرتب ساده‌تر از زبان‌های طبیعی نظیر انگلیسی یا آلمانی است.

برفی (زبان‌ها دارای خواص و امکاناتی است که آنها را برای استفاده در امور خاصی راه‌تر می‌کند.

زبان‌هایی نظیر C++ و پاسکال تقریباً همه منظوره هستند و در اکثر موارد قابل استفاده‌اند.



# زبان‌های برنامه‌نویسی

زبان‌های برنامه‌نویسی:

ساختاری مشابه زبان‌های طبیعی دارند.

زبان‌ها دارای مجموعه کاراکترها، مجموعه کلمات و یک مجموعه دستور زبان یا گرامر هستند.

با استفاده از آنها می‌توان جملات مختلف را ساخت و منظور خود را به طرف مقابل (ساند).

در زبان‌های برنامه‌نویسی، تعداد کلمات (که کلمات کلیدی زبان برنامه‌نویسی نامیده می‌شوند) و دستور (زبان یا گرامر بسیار محدود می‌باشد و لذا یادگیری آن به مرتب ساده‌تر از زبان‌های طبیعی نظیر انگلیسی یا آلمانی است.

برfü (زبان‌ها دارای خواص و امکاناتی است که آنها را برای استفاده در امور خاصی راه‌تر می‌کند.

زبان‌هایی نظیر C++ و پاسکال تقریباً همه منظوره هستند و در اکثر موارد قابل استفاده‌اند.



دانشکده علم پزشکی  
دانشگاه شهید بهشتی

# زبان‌های برنامه‌نویسی

زبان‌های برنامه‌نویسی:

ساختاری مشابه زبان‌های طبیعی دارند.

زبان‌ها دارای مجموعه کاراکترها، مجموعه کلمات و یک مجموعه دستور زبان یا گرامر هستند.  
با استفاده از آنها می‌توان جملات مختلف را ساخت و منظور خود را به طرف مقابل (ساند).

در زبان‌های برنامه‌نویسی، تعداد کلمات (که کلمات کلیدی زبان برنامه‌نویسی نامیده می‌شوند) و دستور (زبان یا گرامر بسیار محدود می‌باشد و لذا یادگیری آن به مرتب ساده‌تر از زبان‌های طبیعی نظیر انگلیسی یا آلمانی است.

برفی (زبان‌ها دارای خواص و امکاناتی است که آنها را برای استفاده در امور خاصی راه‌تر می‌کند.  
زبان‌هایی نظیر C++ و پاسکال تقریباً همه منظوره هستند و در اکثر موارد قابل استفاده‌اند.



دانشکده علم رسانی  
دانشگاه شهید بهشتی

# زبان‌های برنامه‌نویسی

زبان‌های برنامه‌نویسی:

ساختاری مشابه زبان‌های طبیعی دارند.

زبان‌ها دارای مجموعه کاراکترها، مجموعه کلمات و یک مجموعه دستور زبان یا گرامر هستند.

با استفاده از آنها می‌توان جملات مختلف را ساخت و منظور خود را به طرف مقابل (ساند).

در زبان‌های برنامه‌نویسی، تعداد کلمات (که کلمات کلیدی زبان برنامه‌نویسی نامیده می‌شوند) و دستور (زبان یا گرامر بسیار محدود می‌باشد و لذا یادگیری آن به مرتب ساده‌تر از زبان‌های طبیعی نظیر انگلیسی یا آلمانی است.

برfü (زبان‌ها دارای خواص و امکاناتی است که آنها را برای استفاده در امور خاصی راه‌تر می‌کند.

زبان‌هایی نظیر C++ و پاسکال تقریباً همه منظوره هستند و در اکثر موارد قابل استفاده‌اند.



دانشکده علم پزشکی  
دانشگاه شهید بهشتی

# زبان‌های برنامه‌نویسی

زبان‌های برنامه‌نویسی:

ساختاری مشابه زبان‌های طبیعی دارند.

زبان‌ها دارای مجموعه کاراکترها، مجموعه کلمات و یک مجموعه دستور زبان یا گرامر هستند.

با استفاده از آنها می‌توان جملات مختلف را ساخت و منظور خود را به طرف مقابل (ساند).

در زبان‌های برنامه‌نویسی، تعداد کلمات (که کلمات کلیدی زبان برنامه‌نویسی نامیده می‌شوند) و دستور (زبان یا گرامر بسیار محدود می‌باشد و لذا یادگیری آن به مرتب ساده‌تر از زبان‌های طبیعی نظیر انگلیسی یا آلمانی است.

برخلاف زبان‌ها دارای خواص و امکاناتی است که آنها را برای استفاده در امور خاصی راه‌تر می‌کند.

زبان‌هایی نظیر C++ و پاسکال تقریباً همه منظوره هستند و در اکثر موارد قابل استفاده‌اند.



دانشکده علم پزشکی  
دانشگاه شهید بهشتی

# زبان‌های برنامه‌نویسی

زبان‌های برنامه‌نویسی:

ساختاری مشابه زبان‌های طبیعی دارند.

زبان‌ها دارای مجموعه کاراکترها، مجموعه کلمات و یک مجموعه دستور زبان یا گرامر هستند.

با استفاده از آنها می‌توان جملات مختلف را ساخت و منظور خود را به طرف مقابل (ساند).

در زبان‌های برنامه‌نویسی، تعداد کلمات (که کلمات کلیدی زبان برنامه‌نویسی نامیده می‌شوند) و دستور (زبان یا گرامر بسیار محدود می‌باشد و لذا یادگیری آن به مرتب ساده‌تر از زبان‌های طبیعی نظیر انگلیسی یا آلمانی است.

برخلاف زبان‌ها دارای خواص و امکاناتی است که آنها را برای استفاده در امور خاصی راحت‌تر می‌کند.

زبان‌هایی نظیر C++ و پاسکال تقریباً همه منظوره هستند و در اکثر موارد قابل استفاده‌اند.



# فصوصیات زبان C++

## فصوصیات زبان C++:

زبان C++ در اوایل دهه ۱۹۸۰ میلادی با توسعه زبان C ایجاد شد و به دلیل قابلیت‌های آن به طور وسیع در کاربردهای مختلف مورد استفاده قرار گرفت.

قدرت این زبان به حدی است که به عنوان یک زبان برنامه‌نویسی سیستم استفاده شده و می‌شود، به این معنی که بسیاری از برنامه‌های کاربردی پایه‌ای نظیر سیستم‌عامل‌ها با این زبان نوشته شده‌اند. ارتباط نزدیکی بین C++ و زبان اسembly (به عنوان یک زبان سطح پایین) وجود دارد. به عبارتی، تمام امکانات زبان اسembly را می‌توان در C++ استفاده کرد.

امکان توسعه امکانات جدید در زبان C++ بسیار بالاست و همین قابلیت باعث توسعه سریع ابزارها و قابلیت‌های جدید آن از طریق ارائه کتابخانه‌های متعدد است.

این زبان به حروف بزرگ و کوچک مساس است. مثلاً در تابع‌گذاری متغیرها، دو نام Sum و SUM با هم متفاوت هستند و یا کلمات کلیدی زبان باید دقیقاً به همان شکل بیان شده (از نظر حروف بزرگ و کوچک) استفاده شود.



# فصوصیات زبان C++

## فصوصیات زبان C++:

زبان C++ در اوایل دهه ۱۹۸۰ میلادی با توسعه زبان C ایجاد شد و به دلیل قابلیت‌های آن به طور وسیع در کاربردهای مختلف مورد استفاده قرار گرفت.

قدرت این زبان به حدی است که زبان برنامه‌نویسی سیستم استفاده شده و می‌شود، به این معنی که بسیاری از برنامه‌های کاربردی پایه‌ای نظیر سیستم‌عامل‌ها با این زبان نوشته شده‌اند.

از تابع نزدیکی بین C++ و زبان اسمنبلی (به عنوان یک زبان سطح پایین) وجود دارد. به عبارتی، تمام امکانات زبان اسمنبلی را می‌توان در C++ استفاده کرد.

امکان توسعه امکانات جدید در زبان C++ بسیار بالاست و همین قابلیت باعث توسعه سریع ابزارها و قابلیت‌های جدید آن از طریق ارائه کتابخانه‌های متعدد است.

این زبان به حروف بزرگ و کوچک مساس است. مثلاً در تابع‌گذاری متغیرها، دو نام Sum و SUM با هم متفاوت هستند و یا کلمات کلیدی زبان باید دقیقاً به همان شکل بیان شده (از نظر حروف بزرگ و کوچک) استفاده شود.



# خصوصیات زبان C++

## خصوصیات زبان C++:

زبان C++ در اوایل دهه ۱۹۸۰ میلادی با توسعه زبان C ایجاد شد و به دلیل قابلیت‌های آن به طور وسیع در کاربردهای مختلف مورد استفاده قرار گرفت.

قدرت این زبان به حدی است که زبان برنامه‌نویسی سیستم استفاده شده و می‌شود، به این معنی که بسیاری از برنامه‌های کاربردی پایه‌ای نظیر سیستم‌عامل‌ها با این زبان نوشته شده‌اند. ارتباط نزدیکی بین C++ و زبان اسembly (به عنوان یک زبان سطح پایین) وجود دارد. به عبارتی، تماه امکانات زبان اسembly را می‌توان در C++ استفاده کرد.

امکان توسعه امکانات جدید در زبان C++ بسیار بالاست و همین قابلیت باعث توسعه سریع ابزارها و قابلیت‌های جدید آن از طریق ارائه کتابخانه‌های متعدد است.

این زبان به حروف بزرگ و کوچک مساس است. مثلاً در تابع‌گذاری متغیرها، دو نام Sum و SUM با هم متفاوت هستند و یا کلمات کلیدی زبان باید دقیقاً به همان شکل بیان شده (از نظر حروف بزرگ و کوچک) استفاده شود.



# خصوصیات زبان C++

## خصوصیات زبان C++:

زبان C++ در اوایل دهه ۱۹۸۰ میلادی با توسعه زبان C ایجاد شد و به دلیل قابلیت‌های آن به طور وسیع در کاربردهای مختلف مورد استفاده قرار گرفت.

قدرت این زبان به حدی است که زبان برنامه‌نویسی سیستم استفاده شده و می‌شود، به این معنی که بسیاری از برنامه‌های کاربردی پایه‌ای نظیر سیستم‌عامل‌ها با این زبان نوشته شده‌اند. ارتباط نزدیکی بین C++ و زبان اسembly (به عنوان یک زبان سطح پایین) وجود دارد. به عبارتی، تمام امکانات زبان اسembly را می‌توان در C++ استفاده کرد.

امکان توسعه امکانات جدید در زبان C++ بسیار بالاست و همین قابلیت باعث توسعه سریع ابزارها و قابلیت‌های جدید آن از طریق ارائه کتابخانه‌های متعدد است.

این زبان به حروف بزرگ و کوچک مساس است. مثلاً در تابع‌گذاری متغیرها، دو نام Sum و SUM با هم متفاوت هستند و یا کلمات کلیدی زبان باید دقیقاً به همان شکل بیان شده (از نظر حروف بزرگ و کوچک) استفاده شود.



# خصوصیات زبان C++

## خصوصیات زبان C++:

زبان C++ در اوایل دهه ۱۹۸۰ میلادی با توسعه زبان C ایجاد شد و به دلیل قابلیت‌های آن به طور وسیع در کاربردهای مختلف مورد استفاده قرار گرفت.

قدرت این زبان به حدی است که زبان برنامه‌نویسی سیستم استفاده شده و می‌شود، به این معنی که بسیاری از برنامه‌های کاربردی پایه‌ای نظیر سیستم‌عامل‌ها با این زبان نوشته شده‌اند. ارتباط نزدیکی بین C++ و زبان اسembly (به عنوان یک زبان سطح پایین) وجود دارد. به عبارتی، تمام امکانات زبان اسembly را می‌توان در C++ استفاده کرد.

امکان توسعه امکانات جدید در زبان C++ بسیار بالاست و همین قابلیت باعث توسعه سریع ابزارها و قابلیت‌های جدید آن از طریق ارائه کتابخانه‌های متعدد است.

این زبان به حروف بزرگ و کوچک مساس است. مثلاً در نام‌گذاری متغیرها، دو نام SUM و SUM با هم متفاوت هستند و یا کلمات کلیدی زبان باید دقیقاً به همان شکل بیان شده (از نظر حروف بزرگ و کوچک) استفاده شود.





# ساختار یک برنامه ساده

فضوهای زبان C++:

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است.

در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است.

هر دستور با ; (سمی کالن) ختم می‌شود.

نگذاشتن سمی کالن در پیان هر دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک خطای گرامری (syntax error) می‌نامند.

قراردادن سمی کالن اضافی نیز (در برفی جاهای) خطای گرامری است و در برفی جاهای باعث بروز خطای منطقی می‌شود.

زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.

امکان قرار دادن توضیمات در برنامه وجود دارد (قراردادن // در ابتدای خط، یا محدود کردن با /\* و \*/).

**ساختار یک برنامه ساده**

فضوهای زبان C++:

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است.

در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است.

هر دستور با ; (سمی کالن) ختم می‌شود.

نگذاشتن سمی کالن در پیان هر دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک فطاوی گرامی (syntax error) می‌نامند.

قرادادن سمی کالن اضافی نیز (در برفی جاهای) فطاوی گرامی است و در برفی جاهای باعث بروز فطاوی منطقی می‌شود.

زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.

امکان قرار دادن توضیمات در برنامه وجود دارد (قردادن // در ابتدای فط، یا محدود کردن با /\* و \*/).



# ساختار یک برنامه ساده

فضوهای زبان C++:

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است.

در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است.

هر دستور با ; (سمی کالن) ختم می‌شود.

نگذاشتن سمی کالن در پیان هر دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک فطاوی گرامی (syntax error) می‌نامند.

قرادادن سمی کالن اضافی نیز (در برفی جاهای) فطاوی گرامی است و در برفی جاهای باعث بروز فطاوی منطقی می‌شود.

زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.

امکان قرار دادن توضیمات در برنامه وجود دارد (قردادن // در ابتدای فط، یا محدود کردن با /\* و \*/).



# ساختار یک برنامه ساده

فضوهای زبان C++:

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است.

در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است.

هر دستور با ; (سمی کالن) فتم می‌شود.

نگذاشتن سمی کالن در پیان هر دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک خطای گرامری (syntax error) می‌نامند.

قراردادن سمی کالن اضافی نیز (در برفی جاهای) خطای گرامری است و در برفی جاهای باعث بروز خطای منطقی می‌شود.

زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.

امکان قرار دادن توضیمات در برنامه وجود دارد (قراردادن // در ابتدای خط، یا محدود کردن با /\* و \*/).





زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.  
امکان قرار دادن توضیمات در برنامه وجود دارد (قراردادن // در ابتدای خط، یا محدود کردن با /\* و \*/).

## ساختار یک برنامه ساده

**فضوهای زبان C++:**

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است.

در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است.

هر دستور با ; (سمی کالن) فتم می‌شود.

نگذاشتن سمی کالن در پیش از دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک **خطای گرامری (syntax error)** می‌نامند.

قراءادن سمی کالن اضافی نیز (در برفی جاهای) خطای گرامری است و در برفی جاهای باعث بروز خطای منطقی می‌شود.

# ساختار یک برنامه ساده

فضوهای زبان C++:

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است.

در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است.

هر دستور با ; (سمی کالن) ختم می‌شود.

نگذاشتن سمی کالن در پیش از دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک **خطای گرامری (syntax error)** می‌نامند.

قراردادن سمی کالن اضافی نیز (در برفی جاهای) خطای گرامری است و در برفی جاهای باعث بروز **خطای منطقی** می‌شود.

زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.

امکان قرار دادن توضیمات در برنامه وجود دارد (قراردادن // در ابتدای فط، یا محدود کردن با /\* و \*/).



زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.

امکان قرار دادن توضیمات در برنامه وجود دارد (قراردادن // در ابتدای خط، یا محدود کردن با /\* و \*/).

## ساختار یک برنامه ساده

### فضوهای زبان C++:

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است. در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است. هر دستور با ; (سمی کالن) فتم می‌شود.

نگذاشتن سمی کالن در پیان هر دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک **خطای گرامری (syntax error)** می‌نامند.

قراردادن سمی کالن اضافی نیز (در برفی جاهای) خطای گرامری است و در برفی جاهای باعث بروز خطای منطقی می‌شود.

زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.



# ساختار یک برنامه ساده

فضوهای زبان C++:

نوشتن یک برنامه به یک زبان برنامه‌نویسی، مشابه نوشتن یک متن در یک زبان طبیعی است.

در برنامه‌نویسی، به هر جمله یک دستور می‌گویند.

هر دستور، ترجمهٔ یکی از مرامل الگویته به زبان برنامه‌نویسی است.

هر دستور با ; (سمی کالن) فتم می‌شود.

نگذاشتن سمی کالن در پیان هر دستور برنامه، باعث نادرستی دستور در زبان شده که آن را یک **خطای گرامری (syntax error)** می‌نامند.

قراردادن سمی کالن اضافی نیز (در برفی جاهای) خطای گرامری است و در برفی جاهای باعث بروز خطای منطقی می‌شود.

زبان C++ به فضای خالی بین کلمات مساس نیست. حتی در بین کلمات یک دستور هم می‌توان یک فاصله یا بیشتر قرار داد یا هر کلمه را در یک سطر نوشت.

امکان قرار دادن توضیمات در برنامه وجود دارد (قراردادن // در ابتدای فط، یا محدود کردن با /\* و \*/).



# أنواع دادهها و متغيرها

## أنواع دادهها و متغيرها:

هذا في زبان برنامه نویسی، يك سرى از أنواع دادهها مىگند (ذخیره، محاسبات و مقایسه).

أنواع دادههایی که زبان C++ پشتیبانی میگند:

نوع	كلمه كلیدي	باذه قابل قبول/دقت	فضاهي محفظه (بايت)
صحيح	int , long , long int	[-٢٣١, ٢٣١ - ١]	٤
صحيح بزرگ	long long	[-٢٦٣, ٢٦٣ - ١]	٨
اعشاري	float	٧ (قلم اعشار)	٤
اعشاري با دقت مضاعف	double	١٥ (قلم اعشار)	٨
اعشاري با دقت بیشتر	long double	١٥ (قلم اعشار)	٨
كاراكتر	char	[-١٢٧, ١٢٧]	١
دودويي	bool	false یا true	١

برای دادههای صريح و کاراكتری میتوان از کلمات کلیدی signed یا unsigned نیز جلو آنها استفاده کرد. مقدار پیشفرض، signed است.



# أنواع دادهها و متغيرها

## أنواع دادهها و متغيرها:

هذا في زبان برنامه‌نویسی، يك سرى از أنواع دادهها مى‌گند (ذخیره، محاسبات و مقایسه).

أنواع دادههایی که زبان C++ پشتیبانی می‌گند:

نوع	كلمه كلیدي	باذه قابل قبول/دقت	فضاه محفظه (بايت)
صحيح	int , long , long int	[ -٢٣١ , ٢٣١ - ١ ]	٤
صحيح بزرگ	long long	[ -٢٦٣ , ٢٦٣ - ١ ]	٨
اعشاري	float	٧ ( رقم اعشار )	٤
اعشاري با دقت مضاعف	double	١٥ ( رقم اعشار )	٨
اعشاري با دقت بيشر	long double	١٥ ( رقم اعشار )	٨
كاراكتر	char	[ -١٢٧ , ١٢٧ ]	١
دودوبي	bool	false یا true	١

برای دادههای صحيح و کاراكتری می‌توان از کلمات کلیدی unsigned یا signed نیز جلو آنها استفاده کرد. مقدار پیشفرض، signed است.

# انواع داده ها و متغیرها

انواع داده ها و متغیرها:

هر زبان برنامه نویسی، یک سری از انواع داده ها را پشتیبانی می کند (ذخیره، محاسبات و مقایسه).  
انواع داده هایی که زبان C++ پشتیبانی می کند:

فضای محفظه (بایت)	范畴 قابل قبول / دقت	کلمه کلیدی	نوع
۴	[ -۲۳۱, ۲۳۱ - ۱ ]	int , long , long int	صحیع
۸	[ -۲۶۳, ۲۶۳ - ۱ ]	long long	صحیع بزرگ
۴	۷ رقم اعشار	float	اعشاری
۸	۱۵ رقم اعشار	double	اعشاری با دقت مفهومی
۸	۱۵ رقم اعشار	long double	اعشاری با دقت بیشتر
۱	[ -۱۲۷, ۱۲۷ ]	char	کاراکتر
۱	false یا true	bool	دودویی

برای داده های صحیع و کاراکتری می توان از کلمات کلیدی unsigned یا signed نیز جلو آنها استفاده کرد. مقدار پیش فرض، signed است.



# انواع داده ها و متغیرها

## متغیرها:

برای ذخیره سازی داده ها در حافظه، از متغیرها استفاده می کنیم.

در زبان C++ لازم است قبل از استفاده از هر متغیر، نوع داده ای که قرار است در متغیر ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعریف متغیر** می نامند.  
کرامر دستور تعریف متغیر:

نام متغیرها که با کاما از هم جدا شده اند نوع متغیر

مثال: int A,B;

امکان تعریف متغیرها در هر مکانی قبل از اولین استفاده از متغیر ممکن است.  
تعریف متغیر در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی کند.

## قوانین نامگذاری متغیرها:

نام متغیرها باید از حروف انگلیسی باشد (حروف کوچک و بزرگ هم مجاز هستند).  
نامگذاری کریجی از حروف و اعداد و گزینش باشد (حکم کنید که فقط باید با underline تعریق نیست).

نامگذاری کریجی از حروف و اعداد و گزینش باشد (حکم کنید که فقط باید با underline تعریق نیست).  
نامگذاری کریجی از حروف و اعداد و گزینش باشد (حکم کنید که فقط باید با underline تعریق نیست).

# أنواع دادهها و متغيرها

## متغيرها:

برای ذخیره سازی دادهها در حافظه، از متغيرها استفاده می‌کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغير، نوع دادهای که قرار است در متغير ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعريف متغير** می‌نامند.

## گرامر دستور تعريف متغير:

نام متغيرها که با کاما از هم جدا شده‌اند نوع متغير

مثال: int A,B;

امکان تعريف متغيرها در هر مکانی قبل از اولین استفاده از متغير ممکن است.  
تعريف متغير در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی‌کند.

## قوانين نامگذاری متغيرها:

نام متغيرها باید از حروف انگلیسی تشکیل شوند. نام متغير  
گاید از حروف و اعداد و گزینش\_باشد ( وقتی که که باید با با است و علامت  
تتریق نیست).

نام متغيرها باید از حروف انگلیسی تشکیل شوند. نام متغير  
گاید از حروف و اعداد و گزینش\_با است. C++ از پایه

# أنواع دادهها و متغيرها

## متغيرها:

برای ذخیره سازی دادهها در حافظه، از متغيرها استفاده می‌کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغير، نوع دادهای که قرار است در متغير ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعريف متغير** می‌نامند.  
**گرامر دستور تعريف متغير:**

**;نام متغيرها که با کاما از هم جدا شده‌اند نوع متغير**

**مثال:** int A,B;

امکان تعريف متغيرها در هر مکانی قبل از اولین استفاده از متغير ممکن است.  
تعريف متغير در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی‌کند.

## قوانين نامگذاری متغيرها:

نام متغيرها باید از حروف انگلیسی (A-Z) و اعداد (0-9) و زیرآکشن\_پاشه ( تحت خط ) باشند ( وقتی که همچنان با علامت تریق نیست).

نام متغيرها باید از حروف انگلیسی (A-Z) و اعداد (0-9) و زیرآکشن\_پاشه ( تحت خط ) باشند ( وقتی که همچنان با علامت تریق نیست).

# أنواع دادهها و متغيرها

## متغيرها:

برای ذخیره سازی دادهها در حافظه، از متغيرها استفاده می‌کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغير، نوع دادهای که قرار است در متغير ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعريف متغير** می‌نامند.  
**گرامر دستور تعريف متغير:**

نام متغيرها که با کاما از هم جدا شده‌اند نوع متغير

مثال: int A,B;

امکان تعريف متغيرها در هر مکانی قبل از اولین استفاده از متغير ممکن است.  
تعريف متغير در هر قسمت از برنامه، تأثیری در (وند اجرای برنامه ایجاد نمی‌کند.

## قوانين نامگذاری متغيرها:

نام متغيرها باید از حروف انگلیسی (A-Z) و اعداد (0-9) و خط着重下划线 (underline) باشد ( وقت که نام پایین با ترتیق نیست).

نام متغيرها باید از حروف انگلیسی (A-Z) و اعداد (0-9) و خط着重下划线 (underline) باشد ( وقت که نام پایین با ترتیق نیست).

# انواع داده ها و متغیرها

## متغیرها:

برای ذخیره سازی داده ها در حافظه، از متغیرها استفاده می کنیم.  
 در زبان C++ لازم است قبل از استفاده از هر متغیر، نوع داده ای که قرار است در متغیر ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعریف متغیر** می نامند.  
**گرامر دستور تعریف متغیر:**

نام متغیرها که با کاما از هم جدا شده اند نوع متغیر

int A,B;

امکان تعریف متغیرها در هر مکانی قبل از اولین استفاده از متغیر ممکن است.  
 تعریف متغیر در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی کند.

## قوانین نامگذاری متغیرها:

نامگذاری متغیرها باید از حروف و اعداد و گزینه های پایین باشد ( وقتیکه همچنان که underline تأثیر نداشته باشد).  
 تأثیر نداشتن underline بر نامگذاری متغیرها معمول نیست.

# انواع داده ها و متغیرها

## متغیرها:

برای ذخیره سازی داده ها در حافظه، از متغیرها استفاده می کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغیر، نوع داده ای که قرار است در متغیر ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعریف متغیر** می نامند.  
**گرامر دستور تعریف متغیر:**

نام متغیرها که با کاما از هم جدا شده اند نوع متغیر

int A,B;

امکان تعریف متغیرها در هر مکانی قبل از اولین استفاده از متغیر ممکن است.  
تعریف متغیر در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی کند.

## قوانین نامگذاری متغیرها:

نامگذاری متغیرها باید مطابق با قوانین زیر باشد ( فقط تکید نه می باشد با علامت ترکیق نیست).

# أنواع دادهها و متغيرها

## متغيرها:

برای ذخیره‌سازی داده‌ها در حافظه، از متغيرها استفاده می‌کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغير، نوع داده‌ای که قرار است در متغير ذخیره شود، مشخص شود. اين کار را اصطلاحاً **تعريف متغير** می‌نامند.  
**گرامر دستور تعريف متغير:**

نام متغيرها که با کاما از هم جدا شده‌اند نوع متغير

مثال: int A,B;

امکان تعريف متغيرها در هر مکانی قبل از اولین استفاده از متغير ممکن است.  
تعريف متغير در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی‌کند.

## قوانين نامگذاري متغيرها:

نام متغير، جزء کلمات کلیدی زبان نباشد.  
ثانیاً ترکیبی از هروف و اعداد و کاراکتر \_ باشد (دقت کنید که خط پایین یا underline است و علامت تفريقي نیست).

با يك هرف یا \_ شروع شود یا به عبارت ديگر با عدد شروع نشود.  
توجه: زبان C++ به هروف بزرگ و کوچک مساس است.



# انواع داده‌ها و متغیرها

## متغیرها:

برای ذخیره‌سازی داده‌ها در حافظه، از متغیرها استفاده می‌کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغیر، نوع داده‌ای که قرار است در متغیر ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعریف متغیر** می‌نامند.  
**گرامر دستور تعریف متغیر:**

نام متغیرها که با کاما از هم جدا شده‌اند نوع متغیر

مثال: int A,B;

امکان تعریف متغیرها در هر مکانی قبل از اولین استفاده از متغیر ممکن است.  
تعریف متغیر در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی‌کند.

## قواین نامگذاری متغیرها:

نام متغیر، جزء کلمات کلیدی زبان نباشد.  
ثانیاً ترکیبی از حروف و اعداد و کاراکتر \_ باشد (دقت کنید که فطا پایین یا underline است و علامت تلفیق نیست).

با یک حرف یا \_ شروع شود یا به عبارت دیگر با عدد شروع نشود.  
توجه: زبان C++ به حروف بزرگ و کوچک مساس است.



# انواع داده‌ها و متغیرها

## متغیرها:

برای ذخیره‌سازی داده‌ها در حافظه، از متغیرها استفاده می‌کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغیر، نوع داده‌ای که قرار است در متغیر ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعریف متغیر** می‌نامند.  
**گرامر دستور تعریف متغیر:**

نام متغیرها که با کاما از هم جدا شده‌اند نوع متغیر

مثال: int A,B;

امکان تعریف متغیرها در هر مکانی قبل از اولین استفاده از متغیر ممکن است.  
تعریف متغیر در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی‌کند.

## قواین نامگذاری متغیرها:

نام متغیر، جزء کلمات کلیدی زبان نباشد.  
ثانیاً ترکیبی از حروف و اعداد و کاراکتر \_ باشد (دقت کنید که فقط پایین یا underline است و علامت تلفیق نیست).

با یک حرف یا \_ شروع شود یا به عبارت دیگر با عدد شروع نشود.  
توجه: زبان C++ به حروف بزرگ و کوچک مساس است.



# انواع داده‌ها و متغیرها

## متغیرها:

برای ذخیره‌سازی داده‌ها در حافظه، از متغیرها استفاده می‌کنیم.  
در زبان C++ لازم است قبل از استفاده از هر متغیر، نوع داده‌ای که قرار است در متغیر ذخیره شود، مشخص شود. این کار را اصطلاحاً **تعریف متغیر** می‌نامند.  
**گرامر دستور تعریف متغیر:**

نام متغیرها که با کاما از هم جدا شده‌اند نوع متغیر

مثال: int A,B;

امکان تعریف متغیرها در هر مکانی قبل از اولین استفاده از متغیر ممکن است.  
تعریف متغیر در هر قسمت از برنامه، تاثیری در روند اجرای برنامه ایجاد نمی‌کند.

## قواین نامگذاری متغیرها:

نام متغیر، جزء کلمات کلیدی زبان نباشد.  
ثانیاً ترکیبی از حروف و اعداد و کاراکتر \_ باشد (دقت کنید که فقط پایین یا underline است و علامت تلفیق نیست).

با یک حرف یا \_ شروع شود یا به عبارت دیگر با عدد شروع نشود.  
توجه: زبان C++ به حروف بزرگ و کوچک مساس است.



# عملگرهای زبان C++

## عملگرهای زبان C++

عملگر انتساب: برای دادن یک مقدار به یک متغیر. عملگر =  
`double i, j=0,k;`

مثال	نماد	عملگر
$x+y$	+	جمع
$-x$ یا $x-y$	-	تفریق و تفریق یکانی
$x*y$	*	ضرب
$x/y$	/	تقسیم
$x \% y$	%	باقیماندهٔ تقسیم
$++x$ یا $x++$	++	یک واحد افزایش
$--x$ یا $x--$	--	یک واحد کاهش

عملگرهای محساسباتی:

در زبان C++ عملگری برای محساسبهٔ توان و جمود ندارد.



# عملگرهای زبان C++

## عملگرهای زبان C++

عملگر انتساب: برای دادن یک مقدار به یک متغیر. عملگر =  
`double i, j=0,k;`

مثال	نماد	عملگر
$x+y$	+	جمع
$-x$ یا $x-y$	-	تفریق و تفریق یکانی
$x*y$	*	ضرب
$x/y$	/	تقسیم
$x \% y$	%	باقیمانده تقسیم
$++x$ یا $x++$	++	یک واحد افزایش
$--x$ یا $x--$	--	یک واحد کاهش

عملگرهای محسوباتی:

در زبان C++ عملگری برای محسوبه توان وجود ندارد.



# عملگرهای زبان C++

## عملگرهای زبان C++

عملگر انتساب: برای دادن یک مقدار به یک متغیر. عملگر =  
`double i, j=0,k;`

مثال	نماد	عملگر
$x+y$	+	جمع
$-x$ یا $x-y$	-	تفریق و تفریق یکانی
$x*y$	*	ضرب
$x/y$	/	تقسیم
$x \% y$	%	باقیمانده تقسیم
$++x$ یا $x++$	++	یک واحد افزایش
$--x$ یا $x--$	--	یک واحد کاهش

عملگرهای مهاسباتی:

در زبان C++ عملگری برای مهاسبه توان وجود ندارد.



# عملگرهای زبان C++

اولویت انجام عملیات مماسباتی:

فرض کنید متغیرهای  $x$  و  $y$  به ترتیب مقدار ۱ و ۲ است. حاصل عبارت زیر چیست؟

$$x+y*2-4$$

جدول اولویت انجام عملگرها.

عملگرها	اولویت
$++$ و $--$	اول
- منهای یکانی	دوم
$\%$ و $*$ و $/$	سه
$+$ و $-$	چهارم

می‌توان ترتیب امرا عملگرها را با استفاده از پرانتز عوض کرد. برای این منظور، عبارتی که داخل پرانتز ابتدا محاسبه می‌شود و سپس سایر عملگرها محاسبه می‌شوند.

$$((x+y)*2)-4$$



# عملگرهای زبان C++

اولویت انجام عملیات مماسباتی:

فرض کنید متغیرهای  $x$  و  $y$  به ترتیب مقدار ۱ و ۲ دارند. حاصل عبارت زیر چیست؟

$$x+y*2-4$$

جدول: جدول اولویت انجام عملگرها.

عملگرها	اولویت
$++$ و $--$	اول
- منهای یکانی	دوم
$\%$ و $/$ *	سوم
$+$ و $-$	چهارم

می‌توان ترتیب امرا عملگرها را با استفاده از پرانتز عوض کرد. برای این منظور، عبارتی که داخل پرانتز ابتدا محاسبه می‌شود و سپس سایر عملگرها محاسبه می‌شوند.

$$((x+y)*2)-4$$



# عملگرهای زبان C++

اولویت انجام عملیات مماسباتی:

فرض کنید متغیرهای  $x$  و  $y$  به ترتیب مقدار ۱ و ۲ دارند. حاصل عبارت زیر چیست؟

$$x+y*2-4$$

جدول: جدول اولویت انجام عملگرها.

عملگرها	اولویت
$++$ و $--$	اول
- منهای یکانی	دوم
$\%$ و $*$ و $/$	سوم
$+$ و $-$	چهارم

می‌توان ترتیب اجرا عملگرها را با استفاده از پرانتز عوض کرد. برای این منظور، عبارتی که داخل پرانتز ابتدا محاسبه می‌شود و سپس سایر عملگرها محاسبه می‌شوند.

$$((x+y)*2)-4$$



# عملگرهای زبان C++

## مود انجام محاسبات:

همچنان که داده های مختلف، در فضاهای مختلف از حافظه و با فرمتهای مختلف نگهداری می شوند، انجام عملیات محاسباتی (وی آنها نیز متفاوت است. مثلاً جمع دو عدد صحیح با جمع دو عدد اعشاری متفاوت انجام می شود.

اگر عملوندهای یک عمل جمع از نوع صحیح باشند، آنگاه عملیات اصطلاحاً در مود صحیح انجام شده و حاصل نیز یک عدد صحیح خواهد بود.

پس از اجرای دستور  $x=1/2;$  چه مقداری در متغیر  $x$  قرار می گیرد؟ همانجا متوجه عدم دققت در این (مینه) می تواند باشد بجز فضاهای منطقی در برمیگیرد.

$x=1.0/2;$

$x=(float)1/2;$

$x=float(1)/2;$



# عملگرهای زبان C++

## مود انجام محاسبات:

همچنان که داده های مختلف، در فضاهای مختلف از حافظه و با فرمتهای مختلف نگهداری می شوند، انجام عملیات محاسباتی (وی آنها نیز متفاوت است. مثلاً جمع دو عدد صحیح با جمع دو عدد اعشاری متفاوت انجام می شود.

اگر عملوندهای یک عمل جمع از نوع صحیح باشند، آنگاه عملیات اصطلاحاً در مود صحیح انجام شده و حاصل نیز یک عدد صحیح خواهد بود.

پس از اجرای دستور  $x=1/2;$  چه مقداری در متغیر  $x$  قرار می گیرد؟ همانجا مفرا

عدم دقت در این (مینه) می تواند باعث بروز خطاهای منطقی در برنامه شود.

$x=1.0/2;$

$x=(\text{float})1/2;$

$x=\text{float}(1)/2;$



# عملگرهای زبان C++

## مود انجام محاسبات:

همچنان که داده‌های مختلف، در فضاهای مختلف از حافظه و با فرمتهای مختلف نگهداری می‌شوند، انجام عملیات محاسباتی (وی آنها نیز متفاوت است. مثلاً جمع دو عدد صحیح با جمع دو عدد اعشاری متفاوت انجام می‌شود.

اگر عملوندهای یک عمل جمع از نوع صحیح باشند، آنگاه عملیات اصطلاحاً در مود صحیح انجام شده و حاصل نیز یک عدد صحیح خواهد بود.

پس از اجرای دستور `x=1/2;` په مقداری در متغیر `x` قرار می‌گیرد؟ جواب: صفر!

بعد دقت در این متن می‌تواند باعث نیز مطاهی منطقی در برنامه شود.

`x=1.0/2;`

`x=(float)1/2;`

`x=float(1)/2;`



# عملگرهای زبان C++

## مود انجام محاسبات:

همچنان که داده های مختلف، در فضاهای مختلف از حافظه و با فرمتهای مختلف نگهداری می شوند، انجام عملیات محاسباتی (وی آنها نیز متفاوت است. مثلاً جمع دو عدد صمیع با جمع دو عدد اعشاری متفاوت انجام می شود.

اگر عملوندهای یک عمل جمع از نوع صمیع باشند، آنگاه عملیات اصطلاحاً در مود صمیع انجام شده و حاصل نیز یک عدد صمیع خواهد بود.

پس از اجرای دستور  $x=1/2$ ; په مقداری در متغیر  $x$  قرار می گیرد؟ جواب: صفر!

بعد دقت در این متن می تواند باعث نیز مطاهی منطقی در برنامه شود.

$x=1.0/2;$

$x=(\text{float})1/2;$

$x=\text{float}(1)/2;$



# عملگرهای زبان C++

## مود انجام محاسبات:

همچنان که داده های مختلف، در فضاهای مختلف از حافظه و با فرمتهای مختلف نگهداری می شوند، انجام عملیات محاسباتی (وی آنها نیز متفاوت است. مثلاً جمع دو عدد صمیع با جمع دو عدد اعشاری متفاوت انجام می شود.

اگر عملوندهای یک عمل جمع از نوع صمیع باشند، آنگاه عملیات اصطلاحاً در مود صمیع انجام شده و حاصل نیز یک عدد صمیع خواهد بود.

پس از اجرای دستور `x=1/2;` په مقداری در متغیر `x` قرار می گیرد؟ جواب: صفر! عدم دقت در این زمینه می تواند باعث بروز خطاهای منطقی در برنامه شود.

راه حل: تغییر نوع متغیر یا ثابت.

```
x=1.0/2;  
x=( float )1/2;  
x=float(1)/2;
```



# عملگرهای زبان C++

## مود انجام محاسبات:

همچنان که داده های مختلف، در فضاهای مختلف از حافظه و با فرمتهای مختلف نگهداری می شوند، انجام عملیات محاسباتی (وی آنها نیز متفاوت است. مثلاً جمع دو عدد صحیح با جمع دو عدد اعشاری متفاوت انجام می شود.

اگر عملوندهای یک عمل جمع از نوع صحیح باشند، آنگاه عملیات اصطلاحاً در مود صحیح انجام شده و حاصل نیز یک عدد صحیح خواهد بود.

پس از اجرای دستور  $x=1/2$ ; په مقداری در متغیر  $x$  قرار می گیرد؟ جواب: صفر! عدم دقت در این زمینه می تواند باعث بروز خطاهای منطقی در برنامه شود.

اگه حل: تغییر نوع متغیر یا ثابت.

$x=1.0/2;$

$x=(\text{float})1/2;$

$x=\text{float}(1)/2;$



# عملگرهای زبان C++

## عملگرهای مقایسه ای:

مثال	نماد	عملگر
$x > y$	$>$	بزرگتر
$x >= y$	$>=$	بزرگتر یا مساوی
$x < y$	$<$	کوچکتر
$x <= y$	$<=$	کوچکتر یا مساوی
$x == y$	$==$	تساوی
$x != y$	$!=$	نامساوی

در عملگرهای که از ترکیب دو نماد تشکیل شده است؛ مثل  $=<$ ، بین دو نماد نباید فاصله ای قرار بگیرد.

عملگر تساوی به صورت  $==$  است. یکی از اشتباهات رایج در استفاده از عملگر تساوی، استفاده از یک نماد مساوی است که عملاً به جای بزرگی دو مقدار، به عمل انتساب تبدیل می شود.



# عملگرهای زبان C++

عملگرهای مقایسه ای:

مثال	نماد	عملگر
$x > y$	$>$	بزرگتر
$x >= y$	$>=$	بزرگتر یا مساوی
$x < y$	$<$	کوچکتر
$x <= y$	$<=$	کوچکتر یا مساوی
$x == y$	$==$	تساوی
$x != y$	$!=$	نامساوی

در عملگرهای که از ترکیب دو نماد تشکیل شده است؛ مثل  $=<$ ، بین دو نماد نباید فاصله ای قرار بگیرد.

عملگر تساوی به صورت  $==$  است. یکی از اشتباهات رایج در استفاده از عملگر تساوی، استفاده از یک نماد مساوی است که عملاً به جای بزرگی دو مقدار، به عمل انتساب تبدیل می شود.



# عملگرهای زبان C++

## عملگرهای مقایسه ای:

مثال	نماد	عملگر
$x > y$	$>$	بزرگتر
$x >= y$	$>=$	بزرگتر یا مساوی
$x < y$	$<$	کوچکتر
$x <= y$	$<=$	کوچکتر یا مساوی
$x == y$	$==$	تساوی
$x != y$	$!=$	نامساوی

در عملگرهای که از ترکیب دو نماد تشکیل شده است؛ مثل  $=<$ ، بین دو نماد نباید فاصله ای قرار بگیرد.

عملگر تساوی به صورت  $==$  است. یکی از اشتباهات رایج در استفاده از عملگر تساوی، استفاده از یک نماد مساوی است که عملاً به جای بزرگی دو مقدار، به عمل انتساب تبدیل می شود.



# عملگرهای زبان C++

عملگرهای منطقی:

عملگرهای منطقی:

مثال	نماد	عملگر
$(x==y)!$	!	نقیض
$x<0 \&\& y>1$	$\&\&$	و
$x<0 \parallel y>1$	$\parallel$	یا

اولویت عملگرهای منطقی و (ابطالهای:



# C++ زبان

عملگرهای منطقی:

عملگرهای منطقی:

مثال	نماد	عملگر
$(x==y)!$	!	نقیض
$x<0 \&\& y>1$	$\&\&$	و
$x<0 \parallel y>1$	$\parallel$	یا

عملگرهای منطقی	اولویت
!	اول
$<, >, <=, >=$	پنجم
$==, !=$	ششم
$\&\&$	چهارم
$\parallel$	پنجم

اولویت عملگرهای منطقی و ابتداءی:



# عملگرهای زبان C++

عملگرهای ترکیبی:

عملگر	نماد	مثال	معادل با
انتساب جمع	$+=$	$x+=y$	$x=x+y$
انتساب تفریق	$-=$	$x-=y$	$x=x-y$
انتساب ضرب	$*=$	$x*=y$	$x=x*y$
انتساب تقسیم	$/=$	$x/=y$	$x=x/y$
انتساب باقیمانده تقسیم	$\%=$	$x\%=y$	$x=x\%y$

اولویت عملگرها:



# عملگرهای زبان C++

عملگرهای ترکیبی:

معادل با	مثال	نماد	عملگر
$x=x+y$	$x+=y$	$+=$	انتساب جمع
$x=x-y$	$x-=y$	$-=$	انتساب تفریق
$x=x*y$	$x*=y$	$*=$	انتساب ضرب
$x=x/y$	$x/=y$	$/=$	انتساب تقسیم
$x=x\%y$	$x\% =y$	$\% =$	انتساب باقیمانده تقسیم

اولویت عملگرها:

عملگرها	اولویت
$-, !, ++, --$ (منهای یکانی)	اول
$*, /, \%$	دوم
$+, -$	سه
$<, <=, >, >=$	چهارم
$==, !=$	پنجم
$\&&,   $	ششم
$=, + =, - =, * =, / =, \% =$	هفتم



## ۹۰دی و فروجی

۹۰دی و فروجی:

در زبان C++ دستوری جهت ورودی و فروجی وجود ندارد ولی از یک کتابخانه به نام `iostream` مجهت این منظور استفاده می‌کنیم.

گرفتن یک مقدار و قراردادن آن در متغیر X:

```
cin>>x;  
cin>>x>>y>>z;
```

جهت فروجی:

```
cout<<x;  
cout<<x<<y<<z;
```

جهت فروجی یک عبارت، یعنی نوشتن یک عبارت متنی در فروجی،  
`cout<<"The Number is Prime.";`



## ۹۰ دی و فروجی

۹۰ دی و فروجی:

در زبان C++ دستوری جهت ورودی و خروجی وجود ندارد ولی از یک کتابخانه به نام `iostream` مجهت این منظور استفاده می‌کنیم.

گرفتن یک مقدار و قراردادن آن در متغیر x:

```
cin >> x;  
cin >> x >> y >> z;
```

جهت خروجی:

```
cout << x;  
cout << x << y << z;
```

جهت خروجی یک عبارت، یعنی نوشتن یک عبارت متنی در خروجی،  
`cout << "The Number is Prime.";`



## ۹۰ دی و فروجی

۹۰ دی و فروجی:

در زبان C++ دستوری جهت ورودی و خروجی وجود ندارد ولی از یک کتابخانه به نام `iostream` مجهت این منظور استفاده می‌کنیم.

گرفتن یک مقدار و قراردادن آن در متغیر x:

```
cin >> x;  
cin >> x >> y >> z;
```

جهت خروجی:

```
cout << x;  
cout << x << y << z;
```

جهت خروجی یک عبارت، یعنی نوشتن یک عبارت متنی در خروجی،  
`cout << "The Number is Prime.";`



## ۹۰ دی و فروجی

۹۰ دی و فروجی:

در زبان C++ دستوری جهت ورودی و خروجی وجود ندارد ولی از یک کتابخانه به نام `iostream` مجهت این منظور استفاده می‌کنیم.

گرفتن یک مقدار و قراردادن آن در متغیر x:

```
cin >> x;  
cin >> x >> y >> z;
```

جهت خروجی:

```
cout << x;  
cout << x << y << z;
```

جهت خروجی یک عبارت، یعنی نوشتن یک عبارت متنی در خروجی،  
`cout << "The Number is Prime.";`



# ساختار یک برنامه

ساختار یک برنامه:

**توضیح:** شماره فطوط (قرمز) جزء برنامه نیستند و صرفاً برای ارجاع به فطوط برنامه اضافه شده‌اند.

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5
6
7
8
9     return 0;
}
```

شکل: ساختار کلی یک برنامه به زبان C++.



دانشگاه  
علوم پایه‌ی  
شاهرود

برنامه:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     cout<< 2*(3+4);
6     return 0;
7 }
```

مثال ۱

برنامه ای بنویسید که محیط یک مستطیل به طول ۳ و عرض ۴ را محاسبه و چاپ کند.

الگوریتم:

- ۱ شروع
- ۲  $2 \times (3 + 4)$  را چاپ کن
- ۳ پایان

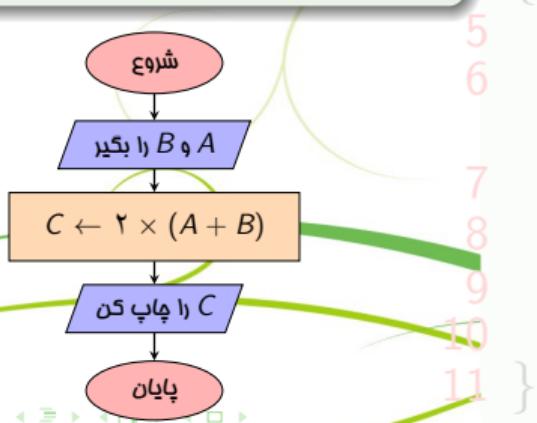


## مثال ۲

برنامه‌ای بنویسید که طول و عرض یک مستطیل را دریافت کرده و محیط مستطیل را محاسبه و چاپ کند.

## بنامه:

- الگوریتم:**
- ۱ شروع
  - ۲ طول و عرض مستطیل را بگیر و در متغیرهای  $A$  و  $B$  قرار بده.
  - ۳ قرار بده  $C \leftarrow 2 \times (A + B)$
  - ۴  $C$  را چاپ کن
  - ۵ پایان



```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double A,B,C;
6     cout<<"Enter Height and Width of
7         rectangle:";
8     cin>>A>>B;
9     C=2*(A+B);
10    cout<<C;
11 }
  
```

## مثال ۲

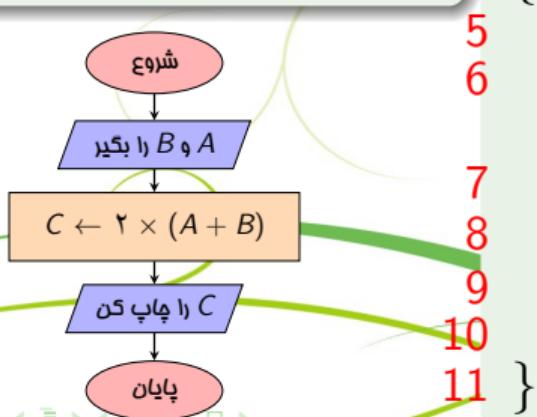
برنامه‌ای بنویسید که طول و عرض یک مستطیل را دریافت کرده و محیط مستطیل را محاسبه و چاپ کند.

## برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double A,B,C;
6     cout<<"Enter Height and Width of
7         rectangle:";
8     cin>>A>>B;
9     C=2*(A+B);
10    cout<<C;
11 }
```

- الگوریتم:**
- ۱ شروع
  - ۲ طول و عرض مستطیل را بگیر و در متغیرهای  $A$  و  $B$  قرار بده.
  - ۳  $C \leftarrow 2 \times (A + B)$  قرار بده
  - ۴  $C$  را چاپ کن
  - ۵ پایان



### مثال ۳

برنامه‌ای بنویسید که سه عدد را دریافت کند و مجموع و میانگین آنها را محاسبه و چاپ کند.

الگوریتم:

۱ شروع

۲ سه عدد را از ورودی را بگیر و آنها را در متغیرهای  $A$  و  $B$  و  $C$  قرار بده.

۳ قرار بده

$AVE \leftarrow S/3$

۴ قرار بده  $AVE$  و  $S$  را چاپ کن

۵ پایان



دانشکده  
علوم پیامبری

## برنامه:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     float A,B,C ;
6     double AVE,S;
7     cout <<"Enter three numbers:" ;
8     cin>>A>>B>>C;
9     S=A+B+C;
10    AVE=S/3;
11    cout<<"Sum is:"<<S;
12    cout<<" Average is: "<<AVE;
13    return 0;
14 }
```

## مثال ۳

برنامه‌ای بنویسید که سه عدد را دریافت کند و مجموع و میانگین آنها را محاسبه و چاپ کند.

## الگوریتم:

۱ شروع

۲ سه عدد را از ورودی را بگیر و آنها را در متغیرهای  $A$  و  $B$  و  $C$  قرار بده.

۳ قرار بده  $S \leftarrow A + B + C$

۴ قرار بده  $AVE \leftarrow S / 3$

۵ چاپ  $AVE$  و  $S$  را چاپ کن

۶ پایان



دانشکده علم رسانی

**مثال ۴**

برنامه‌ای بنویسید که دو عدد را دریافت کرده و در متغیرهای  $A$  و  $B$  ذخیره کند و سپس محتوای دو متغیر را جابجا کرده و نتیجه را چاپ کند.

**الگوریتم:**

- ۱ شروع
- ۲ دو عدد را از ورودی را بگیر و آن را در متغیرهای  $A$  و  $B$  قرار بده.

 $C \leftarrow A$  ۳ $A \leftarrow B$  ۴ $B \leftarrow C$  ۵

- ۶  $B$  و  $A$  را چاپ کن

**۷ پایان**

دانشکده علم رسانی  
دانشگاه شهید بهشتی

**مثال ۴**

برنامه‌ای بنویسید که دو عدد را دریافت کرده و در متغیرهای  $A$  و  $B$  ذخیره کند و سپس محتوای دو متغیر را جابجا کرده و نتیجه را چاپ کند.

**الگوریتم:**

۱ شروع

۲ دو عدد را از ورودی را بگیر و آن را در متغیرهای  $A$  و  $B$  قرار بده.

$C \leftarrow A$  ۳

$A \leftarrow B$  ۴

$B \leftarrow C$  ۵

۶  $A$  و  $B$  را چاپ کن

۷ پایان

**برنامه:**

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double A,B,C;
6     cout<<"Enter two numbers:";
7     cin>>A>>B;
8     C=A;
9     A=B;
10    B=C;
11    cout<<"A="<<A<<
12        ", B="<<B;
13    return 0;
14 }
```



دانشگاه  
یزد

## مثال ۴

برنامه‌ای بنویسید که دو عدد را دریافت کرده و در متغیرهای  $A$  و  $B$  ذخیره کند و سپس محتوای دو متغیر را جابجا کرده و نتیجه را چاپ کند.

الگوریتم:

۱ شروع

۲ دو عدد را از ورودی را بگیر و آن را در متغیرهای  $A$  و  $B$  قرار بده.

$$A \leftarrow A + B \quad ۳$$

$$B \leftarrow A - B \quad ۴$$

$$A \leftarrow A - B \quad ۵$$

۶  $B$  و  $A$  را چاپ کن

۷ پایان



دانشکده علم پزشکی  
دانشگاه شهید بهشتی

**مثال ۴**

برنامه‌ای بنویسید که دو عدد را دریافت کرده و در متغیرهای  $A$  و  $B$  ذخیره کند و سپس محتوای دو متغیر را جابجا کرده و نتیجه را چاپ کند.

**الگوریتم:**

۱ شروع

۲ دو عدد را از ورودی را بگیر و آن را در متغیرهای  $A$  و  $B$  قرار بده.

$$A \leftarrow A + B \quad ۳$$

$$B \leftarrow A - B \quad ۴$$

$$A \leftarrow A - B \quad ۵$$

۶  $A$  و  $B$  را چاپ کن

۷ پایان

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double A, B ;
6     cout<<" Enter two numbers:" ;
7     cin>>A>>B;
8     A=A+B;
9     B=A-B;
10    A=A-B;
11    cout<<"A="<<A
12        <<"," B="<<B;
13    return 0;
14 }
```



دانشگاه  
یزد



# یک دستور یا یک بلوک از دستورات

یک دستور یا یک بلوک از دستورات:

منظور از یک دستور در زبان C++, قسمتی از برنامه است که به یک سمتی کالن ختم می شود.

منظور از یک بلوک از دستورات، یک یا چند دستور است که در بین یک آکولاد باز و یک آکولاد بسته محصور شده است.

در زبان C++, یک بلوک از دستورات دقیقاً کارگردی مشابه یک دستور دارد و هر کجا یک دستور قرار می گیرد را می توان با یک بلوک از دستورات نیز جایگزین کرد.

نکته مهم در فضوهای زبان C++ این است که بلوک از دستورات، وجود آکولاد باز و آکولاد بسته اول و آخر بلوک است.



دانشکده  
علوم پیامبری



# یک دستور یا یک بلوک از دستورات

یک دستور یا یک بلوک از دستورات:

منظور از یک دستور در زبان C++, قسمتی از برنامه است که به یک سمتی کالن ختم می شود.

منظور از یک بلوک از دستورات، یک یا چند دستور است که در بین یک آکولاد باز و یک آکولاد بسته مخصوص شده است.

در زبان C++, یک بلوک از دستورات دقیقاً کارگردی مشابه یک دستور دارد و هر کجا یک دستور قرار می گیرد را می توان با یک بلوک از دستورات نیز جایگزین کرد.

نکته مهم در فضوهای یک بلوک از دستورات، وجود آکولاد باز و آکولاد بسته اول و آخر بلوک است.



دانشکده  
علوم پیامبری



# یک دستور یا یک بلوک از دستورات

یک دستور یا یک بلوک از دستورات:

منظور از یک دستور در زبان C++, قسمتی از برنامه است که به یک سمتی کالن ختم می شود.

منظور از یک بلوک از دستورات، یک یا چند دستور است که در بین یک آکولاد باز و یک آکولاد بسته مخصوص شده است.

در زبان C++, یک بلوک از دستورات دقیقاً کارگردی مشابه یک دستور دارد و هر کجا یک دستور قرار می گیرد را می توان با یک بلوک از دستورات نیز جایگزین کرد.

نکته مهم در فضوهای یک بلوک از دستورات، وجود آکولاد باز و آکولاد بسته اول و آخر بلوک است.



دانشکده  
علوم پیامبری



# یک دستور یا یک بلوک از دستورات

یک دستور یا یک بلوک از دستورات:

منظور از یک دستور در زبان C++, قسمتی از برنامه است که به یک سمتی کالن ختم می‌شود.

منظور از یک بلوک از دستورات، یک یا چند دستور است که در بین یک آکولاد باز و یک آکولاد بسته مخصوص شده است.

در زبان C++, یک بلوک از دستورات دقیقاً کارگردی مشابه یک دستور دارد و هر کجا یک دستور قرار می‌گیرد را می‌توان با یک بلوک از دستورات نیز جایگزین کرد.

نکته مهم در فضوهای زبان C++ این است که آکولاد باز و آکولاد بسته اول و آخر بلوک است.



دانشکده  
علوم پیامبری



# دستور شرطی

## دستور شرطی:

در برخی از الگوریتم‌ها نیاز به بررسی یک شرط داریم و بر مبنای درستی یا نادرستی آن شرط، عملیات مختلفی انجام می‌شود. (لوزی در فلوچارت)

در C++ دستوری جهت بررسی شرط و انجام کارهایی بر مبنای آن وجود دارد. این دستور که به دستور شرطی معروف است به شکل زیر است:

if (شرط)

    یک دستور یا یک بلوک از دستورات  
else

    یک دستور یا یک بلوک از دستورات

قسمت مربوط به else در صورت عدم نیاز قابل حذف شدن است.

دققت شود که دستور شرطی هیچ سمتی کالنی ندارد، اما در قسمت دستور یا بلوک دستورات، بسته به دستور استفاده شده، ممکن است سمتی کالن نیاز باشد.



# دستور شرطی

دستور شرطی:

در برخی از الگوریتم‌ها نیاز به بررسی یک شرط داریم و بر مبنای درستی یا نادرستی آن شرط، عملیات مختلفی انجام می‌شود. (لوزی در فلوچا(t))

در C++ دستوری جهت بررسی شرط و انجام کارهایی بر مبنای آن وجود دارد. این دستور که به دستور شرطی معروف است به شکل زیر است:

**if** (شرط)

یک دستور یا یک بلوک از دستورات  
**else**

یک دستور یا یک بلوک از دستورات

قسمت مربوط به else در صورت عدم نیاز قابل حذف شدن است.

دققت شود که دستور شرطی هیچ سمتی کالنی ندارد، اما در قسمت دستور یا بلوک دستورات، بسته به دستور استفاده شده، ممکن است سمتی کالن نیاز باشد.



# دستور شرطی

دستور شرطی:

در برخی از الگوریتم‌ها نیاز به بررسی یک شرط داریم و بر مبنای درستی یا نادرستی آن شرط، عملیات مختلفی انجام می‌شود. (لوزی در فلوچا(t))

در C++ دستوری جهت بررسی شروط و انجام کارهایی بر مبنای آن وجود دارد. این دستور که به دستور شرطی معروف است به شکل زیر است:

**if** (شرط)

یک دستور یا یک بلوک از دستورات  
**else**

یک دستور یا یک بلوک از دستورات

قسمت مربوط به **else** در صورت عدم نیاز قابل حذف شدن است.

دققت شود که دستور شرطی هیچ سمتی کالنی ندارد، اما در قسمت دستور یا بلوک دستورات، بسته به دستور استفاده شده، ممکن است سمتی کالن نیاز باشد.



# دستور شرطی

## دستور شرطی:

در برخی از الگوریتم‌ها نیاز به بررسی یک شرط داریم و بر مبنای درستی یا نادرستی آن شرط، عملیات مختلفی انجام می‌شود. (لوزی در فلوچا(t))

در C++ دستوری جهت بررسی شروط و انجام کارهایی بر مبنای آن وجود دارد. این دستور که به دستور شرطی معروف است به شکل زیر است:

if (شرط)

    یک دستور یا یک بلوک از دستورات  
else

    یک دستور یا یک بلوک از دستورات

قسمت مربوط به else در صورت عدم نیاز قابل حذف شدن است.

دققت شود که دستور شرطی هیچ سمتی کالانی ندارد، اما در قسمت دستور یا بلوک دستورات، بسته به دستور استفاده شده، ممکن است سمتی کالان نیاز باشد.



## مثال ۵

برنامه‌ای بنویسید که یک عدد صحیح را از ورودی دریافت کند و مشخص کند عدد وارد شده زوج است یا فرد.

الگوریتم:

- ۱ شروع
- ۲  $n$  را بگیر
- ۳ قرار بده  $2$
- ۴ اگر  $r = 0$  آنگاه  
    ۵ «زوج» را چاپ کن
- ۶ در غیر راینصورت
- ۷ «فرد» را چاپ کن
- ۸ پایان اگر
- ۹ پایان



دانشگاه  
شہید بهشتی

## مثال ۵

برنامه ای بنویسید که یک عدد صحیح را از ورودی دریافت کند و مشخص کند عدد وارد شده زوج است یا فرد.

## الگوریتم:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int n , r;
6     cout<<"Enter a number";
7     cin>>n;
8     r=n%2;
9     if (r==0)
10        cout<<"Even";
11     else
12        cout<<"Odd";
13     return 0;
14 }
```

- |   |                              |
|---|------------------------------|
| ۱ | شروع                         |
| ۲ | $n$ را بگیر                  |
| ۳ | $r \leftarrow n - [n/2] * 2$ |
| ۴ | اگر $r = 0$ آنگاه            |
| ۵ | «زوج» را چاپ کن              |
| ۶ | در غیر راینصورت              |
| ۷ | «فرد» را چاپ کن              |
| ۸ | پایان اگر                    |
| ۹ | پایان                        |

## مثال ۶

برنامه‌ای بنویسید که یک عدد را از ورودی دریافت کند و قدرمطلق آن را محاسبه و چاپ کند.

الگوریتم:

- ۱ شروع
- ۲  $x$  را بگیر
- ۳ اگر  $x < 0$  آنگاه
- ۴     قرار بده  $y \leftarrow -x$
- ۵ در غیر اینصورت
- ۶     قرار بده  $y \leftarrow x$
- ۷ پایان اگر
- ۸  $y$  را چاپ کن
- ۹ پایان



دانشگاه  
شهید بهشتی



## برنامه:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double x,y;
6     cout << "Enter x : ";
7     cin >> x ;
8     if ( x < 0 )
9         y = -1 * x;
10    else
11        y = x ;
12    cout << y;
13    return 0;
14 }
```

## مثال ۶

برنامه ای بنویسید که یک عدد را از ورودی دریافت کند و قدر مطلق آن را محاسبه و چاپ کند.

## الگوریتم:

- ۱ شروع
- ۲  $x$  را بگیر
- ۳ اگر  $x < 0$  آنگاه
- ۴  $y \leftarrow -1 \times x$  قرار بده
- ۵ در غیر راینصورت
- ۶  $y \leftarrow x$  قرار بده
- ۷ پایان اگر
- ۸  $y$  را چاپ کن
- ۹ پایان

## مثال ۷

برنامه‌ای بنویسید که سه عدد را از ورودی دریافت کند و بیشترین عدد را محاسبه و چاپ کند.

الگوریتم:

۱ شروع  
۲  $C \leftarrow A$  و  $B \leftarrow A$  را بگیر  
 $MAX \leftarrow A$  ۳

۴ اگر  $B > MAX$  آنگاه  
 $MAX \leftarrow B$  ۵  
قرار بده

۶ پیمان اگر  
اگر  $C > MAX$  آنگاه ۷  
 $MAX \leftarrow C$  ۸  
قرار بده

۹ پیمان اگر  
۱۰  $MAX$  را چاپ کن ۱۱ پیمان



دانشکده  
علوم پیامبری

## مثال ۷

برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double A,B,C,MAX ;
6     cout<<"Enter three numbers:" ;
7     cin>>A>>B>>C;
8     MAX=A;
9     if ( B > MAX )
10        MAX=B;
11     if ( C > MAX )
12        MAX =C;
13     cout<<"The Maximum is "<<
14         MAX;
15     return 0;
}

```

برنامه ای بنویسید که سه عدد را از ورودی دریافت کند و بیشترین عدد را محاسبه و چاپ کند.

الگوریتم:

- ۱ شروع
- ۲  $A, B, C$  را بگیر
- ۳  $MAX \leftarrow A$
- ۴ اگر  $B > MAX$  آنگاه
- ۵  $MAX \leftarrow B$  قرار بده
- ۶ پایان اگر
- ۷ اگر  $C > MAX$  آنگاه
- ۸  $MAX \leftarrow C$  قرار بده
- ۹ پایان اگر
- ۱۰  $MAX$  را چاپ کن
- ۱۱ پایان

## مثال ۸

برنامه‌ای بنویسید که سه عدد را از ورودی دریافت کند و مشخص کند آیا مثلث قائم‌الزاویه‌ای با آن طول اضلاع وجود دارد یا نه.

الگوریتم:

- ۱ شروع
- ۲  $C^2 = A^2 + B^2$  آنگاه | ۱۰ اگر  $C^2 = A^2 + B^2$  را بگیر
- ۳ قرار بده | ۱۱ پایان اگر  $C^2 = A^2 + B^2$  را بگیر
- ۴ اگر  $C^2 = A^2 + B^2$  آنگاه | ۱۲ پایان اگر  $C^2 = A^2 + B^2$  را بگیر
- ۵ قرار بده | ۱۳ اگر  $A^2 = B^2 + C^2$  آنگاه  $C^2 = A^2 + B^2$  را پاپ کن
- ۶ پایان اگر درغیراینصورت | ۱۴ قرار بده | ۱۵ درغیراینصورت
- ۷ اگر  $B^2 = A^2 + C^2$  آنگاه | ۱۶ اگر  $B^2 = A^2 + C^2$  آنگاه  $B^2 = A^2 + C^2$  را پاپ کن
- ۸ قرار بده | ۱۷ پایان اگر  $B^2 = A^2 + C^2$  را پاپ کن
- ۹ پایان اگر



دانشکده علم رسانی  
دانشگاه شهید بهشتی

## برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double A,B,C,flag =0;
6     cout<<"Enter three
7         numbers:";
8     cin>>A>>B>>C;
9     if (A*A ==B*B +C*C)
10        flag=1;
11     if ( B*B==A*A +C*C)
12        flag =1;
13     if ( C*C==B*B+A*A)
14        flag=1;
15     if ( flag==1)
16         cout<<"YES";
17     else
18         cout<<"NO";
19     return 0;
}
  
```

## مثال ۸

برنامه ای بنویسید که سه عدد را از ورودی دریافت کند و مشخص کند آیا مثلث قائم الزاویه ای با آن طول اضلاع وجود دارد یا خیر.

## الگوریتم:

- |  |   |
|--|---|
| ۱ شروع   | ۱۰ اگر $C^2 = A^2 + B^2$ آنگاه              |
| ۲ $C$ و $B$ و $A$ را بگیر                      | ۱۱ قرار بده   ۱۲ پایان اگر                  |
| ۳ قرار بده $\circ$                             | ۱۳ $flag = 1$ آنگاه $A^2 = B^2 + C^2$ آنگاه |
| ۴ اگر $flag = 1$ آنگاه                         | ۱۴ $flag \leftarrow 1$ YES را چاپ کن        |
| ۵ قرار بده   ۶ پایان اگر                       | ۱۵ در غیر این صورت                          |
| ۷ اگر $flag = 1$ آنگاه $B^2 = A^2 + C^2$ آنگاه | ۱۶ $flag \leftarrow 1$ NO را چاپ کن         |
| ۸ قرار بده   ۹ پایان اگر                       | ۱۷ پایان اگر                                |



## مثال ۹

برنامه‌ای بنویسید که سه عدد را از ورودی دریافت کند و آنها را به ترتیب صعودی در خروجی چاپ کند.

## الگوریتم:

۱ شروع	۱۰ درغیراینصورت
۲ $C \geq B \text{ و } A \geq C$ را بگیر	۱۱ $y \geq A$ آنگاه
۳ $A \geq B$ آنگاه	۱۲ $x \leftarrow B$ قرار بده
۴ $x \leftarrow A$ قرار بده	۱۳ $y \leftarrow C$ درغیراینصورت
۵ درغیراینصورت	۱۴ $x \leftarrow A$ قرار بده
۶ $x \leftarrow A$ قرار بده	۱۵ $y \leftarrow B$ پایان اگر
۷ $y \leftarrow B$ پایان اگر	۱۶ $x \leftarrow C$ پایان اگر
۸ $x \geq C$ آنگاه	۱۷ $z \leftarrow x$ اول $x$ و سپس $y$ و سپس $z$
۹ $x \leftarrow C$ پایان	۱۸ $y \leftarrow z$ را چاپ کن



دانشکده  
علوم پیامبری

## برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     double A,B,C,x,y,z;
6     cout<<"Enter three
         numbers:";
7     cin>>A>>B>>C;
8     if (A>=B)
9         { x=B; y=A; }
10    else
11        { x=A; y=B; }
12    if (x>=C)
13        { z=y; y=x; x=C; }
14    else
15        if (y>=C)
16            { z=y; y=C; }
17        else
18            z=C;
19    cout<<x<<","<<y<<","<<z;
20    return 0;
21 }
```

مثال ۹

برنامه‌ای بنویسید که سه عدد را از ورودی دریافت کند و آنها را به ترتیب صعودی در خروجی پاپ کند.

الگوریتم:

۱ شروع	۱۰ درغیرابینصورت
۲ C و B و A را بگیر	۱۱ اگر $y \geq C$ آنگاه
۳ آنگاه $A \geq B$ اگر	۱۲ قرار بده $y \leftarrow z$
۴ قرار بده $x \leftarrow B$	۱۳ درغیرابینصورت
۵ درغیرابینصورت	۱۴ قرار بده $y \leftarrow A$
۶ قرار بده $x \leftarrow A$	۱۵ پایان اگر
۷ قرار بده $y \leftarrow B$	۱۶ پایان اگر
۸ پایان اگر	۱۷ اول x و سپس y و سپس z
۹ اگر $x \geq C$ آنگاه	۱۸ قرار بده $z \leftarrow y$
۱۰ پایان	۱۹ $x \leftarrow C$ و $y \leftarrow x$



# دستور switch

دستور switch :

در هالتی که می‌خواهیم بر مبنای مقدار یک متغیر یا یک عبارت مماسباتی تعداد محدودی حالت را بررسی و بر مبنای آن کار فاصی انجام دهیم، می‌توانیم از دستور switch استفاده کنیم.

این کار با استفاده از دستور if نیز امکان‌پذیر است ولی در برخی حالات، استفاده از دستور switch راحت‌تر است.

ساختار گرامری دستور switch :

```
1 switch( expression )
2 {
3     case constant-expression1:
4         statement(s);
5         break;
6     case constant-expression2:
7         statement(s);
8         break;
9     ...
10    default:
11        statement(s);
12 }
```



# switch دستور

switch دستور :

در هالتی که می خواهیم بر مبنای مقدار یک متغیر یا یک عبارت مماسباتی تعداد محدودی حالت را بررسی و بر مبنای آن کار فاصی انجام دهیم، می توانیم از دستور switch استفاده کنیم.

این کار با استفاده از دستور if نیز امکان پذیر است ولی در برخی حالات، استفاده از دستور switch راحت تر است.

```

1 switch( expression )
2 {
3     case constant-expression1:
4         statement(s);
5         break;
6     case constant-expression2:
7         statement(s);
8         break;
9     ...
10    default :
11        statement(s);
12 }
```

switch دستور ساختار گرامری :



# دستور switch

دستور switch :

در هالتی که می‌خواهیم بر مبنای مقدار یک متغیر یا یک عبارت مماسباتی تعداد محدودی حالت را بررسی و بر مبنای آن کار فاصی انجام دهیم، می‌توانیم از دستور switch استفاده کنیم.

این کار با استفاده از دستور if نیز امکان‌پذیر است ولی در برخی حالات، استفاده از دستور switch امتحان‌تر است.

```
1 switch( expression )
2 {
3     case constant-expression1:
4         statement(s);
5         break;
6     case constant-expression2:
7         statement(s);
8         break;
9     ...
10    default:
11        statement(s);
12 }
```

ساختار گرامری دستور switch :



## دستور switch

دستور switch :

ساختار گرامری دستور switch :

عملکرد این دستور به این صورت است که مقدار عبارت expression بروزی شده و در صورتی که این مقدار برابر با constant-expression1 باشد، دستورات بعد از آن دستور تا دستور break اجرا شده و سپس اجرای برنامه به اولین دستور بعد از دستور switch منتقل می‌شود. این کار برای سایر عبارت‌ها نیز تکرار می‌شود.

در صورتی که expression با هیچ یک از عبارات جلوی case‌ها برابر نباشد، دستورات قسمت default اجرا می‌شود.

```
1 switch( expression )
2 {
3     case constant-expression1:
4         statement(s);
5         break;
6     case constant-expression2:
7         statement(s);
8         break;
9     ...
10    default :
11        statement(s);
12 }
```



# switch دستور

```

1 switch( expression )
2 {
3     case constant-expression1:
4         statement(s);
5         break;
6     case constant-expression2:
7         statement(s);
8         break;
9     ...
10    default :
11        statement(s);
12 }
```



دستور : switch

ساختار گرامری دستور : switch

عملکرد این دستور به این صورت است که مقدار عبارت expression بروزی شده و در صورتی که این مقدار برابر با constant-expression1 باشد، دستورات بعد از آن دستور تا دستور break اجرا شده و سپس اجرای برنامه به اولین دستور بعد از دستور switch منتقل می‌شود. این کار برای سایر عبارت‌ها نیز تکرار می‌شود.

در صورتی که expression با هیچ یک از عبارات جلویی case ها برابر نباشد، دستورات قسمت default اجرا می‌شود.

## دستور switch

دستور switch :

ساختار گرامری دستور switch :

عملکرد این دستور به این صورت است که مقدار عبارت expression بروزی شده و در صورتی که این مقدار برابر با constant-expression1 باشد، دستورات بعد از آن دستور تا دستور break اجرا شده و سپس اجرای برنامه به اولین دستور بعد از دستور switch منتقل می‌شود. این کار برای سایر عبارت‌ها نیز تکرار می‌شود.

در صورتی که expression با هیچ یک از عبارات جلوی case‌ها برابر نباشد، دستورات قسمت default اجرا می‌شود.

```

1 switch( expression )
2 {
3     case constant-expression1:
4         statement(s);
5         break;
6     case constant-expression2:
7         statement(s);
8         break;
9     ...
10    default :
11        statement(s);
12 }
```



# دستور switch

نکات زیر در خصوص استفاده از دستور `switch` باید رعایت شود:

حاصل عبارت `expression` باید یک مقدار صحیح یا از یک نوع مقدار قابل شمارش باشد.

حاصل عبارات `constant-expression` باید از همان نوع `expression` و یک مقدار ثابت یا متغیر باشد.

دستور `break` در انتهای دستورات `case` افتیالی است. قانون کلی این است که در صورت ورود به یک حالت `case`، دستورات بعدی تا رسیدن به یک دستور `break` اجرا می شود.

استفاده از بخش `default` نیز افتیالی است. این بخش در صورتی اجرا می شود که عبارت `expression` با هیچگدام از عبارات مقابل `case`ها برابر نباشد.



دانشکده  
علوم پیامبری

# دستور switch

نکات زیر در خصوص استفاده از دستور `switch` باید رعایت شود:

حاصل عبارت `expression` باید یک مقدار صحیح یا از یک نوع مقدار قابل شمارش باشد.

حاصل عبارات `constant-expression` باید از همان نوع `expression` و یک مقدار ثابت یا متغیر باشد.

دستور `break` در انتهای دستورات `case` افتیالی است. قانون کلی این است که در صورت ورود به یک حالت `case`، دستورات بعدی تا رسیدن به یک دستور `break` اجرا می شود.

استفاده از بخش `default` نیز افتیالی است. این بخش در صورتی اجرا می شود که عبارت `case` با هیچگدام از عبارات مقابل `expression` برابر نباشد.



دانشکده  
علوم پیامبری

# دستور switch

نکات زیر در خصوص استفاده از دستور `switch` باید رعایت شود:

حاصل عبارت `expression` باید یک مقدار صحیح یا از یک نوع مقدار قابل شمارش باشد.

حاصل عبارات `constant-expression` باید از همان نوع `expression` و یک مقدار ثابت یا متغیر باشد.

دستور `break` در انتهای دستورات `case` افتیالی است. قانون کلی این است که در صورت ورود به یک حالت `case`، دستورات بعدی تا رسیدن به یک دستور `break` اجرا می‌شود.

استفاده از بخش `default` نیز افتیالی است. این بخش در صورتی اجرا می‌شود که عبارت `expression` با هیچگدام از عبارات مقابل `case`ها برابر نباشد.



دانشکده  
علوم پیامبری



# دستور switch

نکات زیر در خصوص استفاده از دستور `switch` باید رعایت شود:

حاصل عبارت `expression` باید یک مقدار صحیح یا از یک نوع مقدار قابل شمارش باشد.

حاصل عبارات `constant-expression` باید از همان نوع `expression` و یک مقدار ثابت یا متغیر باشد.

دستور `break` در انتهای دستورات `case` افتیالی است. قانون کلی این است که در صورت ورود به یک حالت `case`، دستورات بعدی تا رسیدن به یک دستور `break` اجرا می‌شود.

استفاده از بخش `default` نیز افتیالی است. این بخش در صورتی اجرا می‌شود که عبارت `expression` با هیچگدام از عبارات مقابل `case`ها برابر نباشد.



دانشگاه  
علوم پزشکی  
شاهرود

## switch دستور

مثال:

```
1 #include <iostream>          15     cout<<"Well done";  
2 using namespace std;        16     break;  
3 int main ()                17     case 'D' :  
4 {                           18     cout<<"You passed";  
5     char grade;             19     break;  
6     cout<<"Enter grade:";  20     case 'F' :  
7     cin>>grade;            21     cout<<"Better try again";  
8     switch(grade)           22     break;  
9     {                      23     default :  
10       case 'A' :           24     cout<<"Invalid grade";  
11         cout<<"Excellent!"; 25     }  
12         break;            26     cout<<"Your grade is "<<grade;  
13       case 'B' :           27     return 0;  
14       case 'C' :           28 }
```



# ساختار ملچه تکرار

## ساختار ملچه تکرار:

در حل بسیاری از مسائل نیاز داشتیم با استفاده از ملچه‌های تکرار، تعدادی دستور را پندين باز اجرا کنیم.

با دانستن دستور شرطی در زبان C++ و معرفی دستوری که معادل «برو به مرحله ....» باشد، می‌توان

ملچه‌ها را در زبان برنامه‌نویسی پیاده کرد. البته چنین دستوری نیز در C++ وجود دارد اما به دلیل

ساختاریافته نبودن این شیوه برای ملچه‌های تکرار، از دستورات ملچه تکرار در C++ استفاده فواهیم کرد.

در زبان C++ سه نوع ملچه تکرار وجود دارد. (ملچه while، ملچه for و ملچه do).



دانشکده  
علوم پیامبری

# سافتار ملقه تکرار

## سافتار ملقه تکرار:

در حل بسیاری از مسائل نیاز داشتیم با استفاده از ملقه‌های تکرار، تعدادی دستور را پندين باز اجرا کنیم.  
با دانستن دستور شرطی در زبان C++ و معرفی دستوری که معادل «برو به مرحله ....» باشد، می‌توان ملقه‌ها را در زبان برنامه‌نویسی پیاده کرد. البته پنین دستوری نیز در C++ وجود دارد اما به دلیل سافتاریافته نبودن این شیوه برای ملقه‌های تکرار، از دستورات ملقه تکرار در C++ استفاده فواهیم کرد.

در زبان C++ سه نوع ملقه تکرار وجود دارد. (ملقه while و ملقه for و ملقه do while).



دانشکده  
علوم ریاضی

# سافتار ملقه تکرار

## سافتار ملقه تکرار:

در حل بسیاری از مسائل نیاز داشتیم با استفاده از ملقه‌های تکرار، تعدادی دستور را پندين باز اجرا کنیم.

با دانستن دستور شرطی در زبان C++ و معرفی دستوری که معادل «برو به مرحله ....» باشد، می‌توان ملقه‌ها را در زبان برنامه‌نویسی پیاده کرد. البته پنین دستوری نیز در C++ وجود دارد اما به دلیل سافتاریافته نبودن این شیوه برای ملقه‌های تکرار، از دستورات ملقه تکرار در C++ استفاده فواهیم کرد.

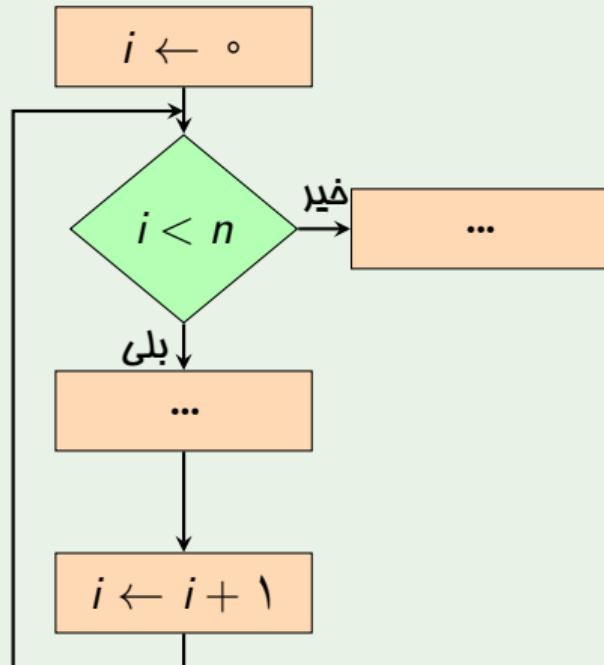
در زبان C++ سه نوع ملقه تکرار وجود دارد. (ملقه for، ملقه while و ملقه do).



دانشکده  
علوم ریاضی

ساختار ملکه تکرار  
for دستور ملکه

دستور ملکه for



## دستور ملقة for

( دستور ; شرط ; دستور )

یک دستور یا یک بلوک از دستورات

دقت کنید که در پرانتز (وبروی عبارت for دو سمتی کالن وجود دارد که سه عبارت را از هم جدا می‌کند.  
حتی با نبود دستورات و شرط سمتی کالن را باید گذاشت.

دستور اول در داخل پرانتز، معادل دستوری است که قبل از ملقة می‌آید.

دستور دوهم، شرط ملقة یعنی همان شرطی که برقرار بودن آن به تکرار ملقة و برقرار نبودن آن به پایان ملقة منجر می‌شود قرار می‌گیرد.

دستور سوم، دستوری است که پس از هر بار تکرار ملقة اجرا می‌شود.

دستور یا یک بلوک از دستورات مربوط به دستور ملقة for را بدنه ملقة می‌نامند. بدنه ملقة در صورت درستی شرط تکرار می‌شود.



# ساختار حلقه تکرار

## for دستور حلقه

دستور حلقه : for

( دستور ; شرط ; دستور )

یک دستور یا یک بلوک از دستورات

دقت کنید که در پرانتز و بروی عبارت for دو سمتی کالن وجود دارد که سه عبارت را از هم جدا نمی‌کند.

حتی با قبول دستورات و شرط سمتی کالن را باید گذاشت.

دستور اول در داده این پرانتز، معادل دستوری است که قبل از حلقه می‌آید.

دستور دوهم، شرط حلقه یعنی همان شرطی که برقرار بودن آن به تکرار حلقه و برقرار نبودن آن به پایان حلقه منجر می‌شود قرار می‌گیرد.

دستور سوم، دستوری است که پس از هر بار تکرار حلقه اجرا می‌شود.

دستور یا یک بلوک از دستورات مربوط به دستور حلقه for را بدنه حلقه می‌نامند. بدنه حلقه در صورت درستی شرط تکرار می‌شود.



ساختار حلقه تکرار  
for دستور حلقه

دستور حلقه for

( دستور ; شرط ; دستور )

یک دستور یا یک بلوک از دستورات

دقت کنید که در پرانتز و بروی عبارت for دو سمی کالن وجود دارد که سه عبارت را از هم جدا می‌کند.  
هتی با نبود دستورات و شرط سمی کالن را باید گذاشت.

دستور اول در داخل پرانتز، معادل دستوری است که قبل از حلقه می‌آید.

دستور دوهم، شرط حلقه یعنی همان شرطی که برقرار بودن آن به تکرار حلقه و برقرار نبودن آن به پایان حلقه منجر می‌شود قرار می‌گیرد.

دستور سوم، دستوری است که پس از هر بار تکرار حلقه اجرا می‌شود.

دستور یا یک بلوک از دستورات مربوط به دستور حلقه for را بدنه حلقه می‌نامند. بدنه حلقه در صورت درستی شرط تکرار می‌شود.



## دستور ملقه for

( دستور ; شرط ; دستور )

یک دستور یا یک بلوک از دستورات

دقت کنید که در پرانتز و بروی عبارت for دو سمی کالن وجود دارد که سه عبارت را از هم جدا می‌کند.

هتی با نبود دستورات و شرط سمی کالن را باید گذاشت.

دستور اول در دادفل پرانتز، معادل دستوری است که قبل از ملقه می‌آید.

دستور دوم، شرط ملقه یعنی همان شرطی که برقرار بودن آن به تکرار ملقه و برقرار نبودن آن به پایان ملقه منجر می‌شود قرار می‌گیرد.

دستور سوم، دستوری است که پس از هر بار تکرار ملقه اجرا می‌شود.

دستور یا یک بلوک از دستورات مربوط به دستور ملقه for را بدنه ملقه می‌نامند. بدنه ملقه در صورت درستی شرط تکرار می‌شود.



ساختار ملقه تکرار  
for دستور ملقة

دستور ملقة for

( دستور ; شرط ; دستور )

یک دستور یا یک بلوک از دستورات

دقت کنید که در پرانتز و بروی عبارت `for` دو سمی کالن وجود دارد که سه عبارت را از هم جدا نمی‌کند.  
هتی با نبود دستورات و شرط سمی کالن را باید گذاشت.

دستور اول در دادفل پرانتز، معادل دستوری است که قبل از ملقة می‌آید.

دستور دوم، شرط ملقة یعنی همان شرطی که برقرار بودن آن به تکرار ملقة و برقرار نبودن آن به پایان ملقة منجر می‌شود قرار می‌گیرد.

دستور سوم، دستوری است که پس از هر بار تکرار ملقة اجرا می‌شود.

دستور یا یک بلوک از دستورات مربوط به دستور ملقة `for` را بدنه ملقة می‌نامند. بدنه ملقة در صورت درستی شرط تکرار می‌شود.



ساختار ملقه تکرار  
for دستور ملقه

دستور ملقه for

( دستور ; شرط ; دستور )

یک دستور یا یک بلوک از دستورات

دقت کنید که در پرانتز و بروی عبارت for دو سمی کالن وجود دارد که سه عبارت را از هم جدا می‌کند.  
هتی با نبود دستورات و شرط سمی کالن را باید گذاشت.

دستور اول در دادفل پرانتز، معادل دستوری است که قبل از ملقه می‌آید.

دستور دوم، شرط ملقه یعنی همان شرطی که برقرار بودن آن به تکرار ملقه و برقرار نبودن آن به پایان  
ملقه منجر می‌شود قرار می‌گیرد.

دستور سوم، دستوری است که پس از هر بار تکرار ملقه اجرا می‌شود.

دستور یا یک بلوک از دستورات مربوط به دستور ملقه for را بدنه ملقه می‌نامند. بدنه ملقه در صورت  
درستی شرط تکرار می‌شود.



ساختار ملقه تکرار  
for دستور ملقه

دستور ملقه for

( دستور ; شرط ; دستور )

یک دستور یا یک بلوک از دستورات

دقت کنید که در پرانتز و بروی عبارت for دو سمی کالن وجود دارد که سه عبارت را از هم جدا نمی‌کند.  
هتی با نبود دستورات و شرط سمی کالن را باید گذاشت.

دستور اول در دادفل پرانتز، معادل دستوری است که قبل از ملقه می‌آید.

دستور دوم، شرط ملقه یعنی همان شرطی که برقرار بودن آن به تکرار ملقه و برقرار نبودن آن به پایان  
ملقه منجر می‌شود قرار می‌گیرد.

دستور سوم، دستوری است که پس از هر بار تکرار ملقه اجرا می‌شود.

دستور یا یک بلوک از دستورات مربوط به دستور ملقه for را بدنه ملقه می‌نامند. بدنه ملقه در صورت  
درستی شرط تکرار می‌شود.



## ساختار ملکه تکرار

for دستور ملکه

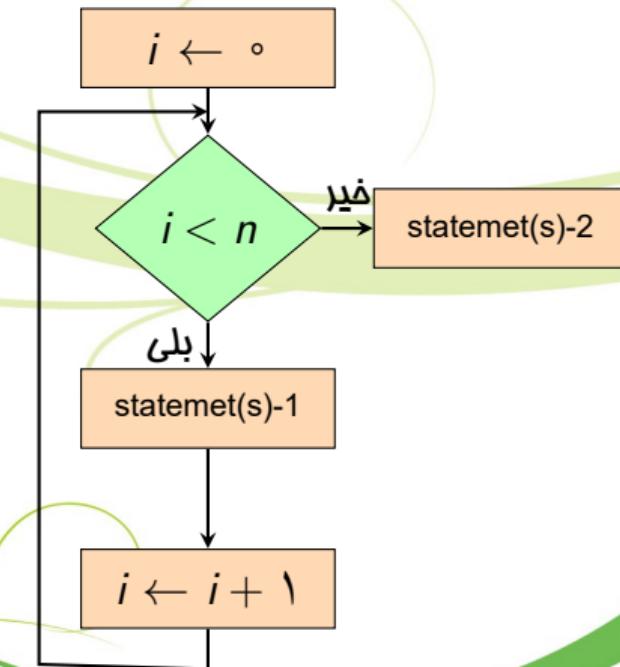
```

1   for ( i=0; i<n; i++)
2   {
3       statemet(s)-1
4   }
5   statemet(s)-2

```



دانشگاه  
علوم پزشکی  
شاهرود



## ساختار ملقة تکرار

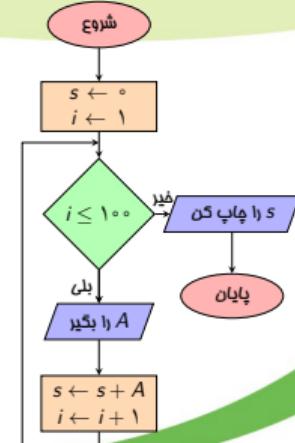
for دستور ملقة

برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i;
6     float A , s=0 ;
7     for ( i=1; i<=100;i++)
8     {
9         cout<<"\n Enter a number:";
10        cin>>A;
11        s+=A;
12    }
13    cout<<s;
14    return 0;
15 }
```

**مثال ۱۰**  
 برنامه ای بنویسید که ۱۰۰ عدد را از ورودی دریافت کند و مجموع آنها را محاسبه و پاپ کند.



## ساختار حلقه تکرار

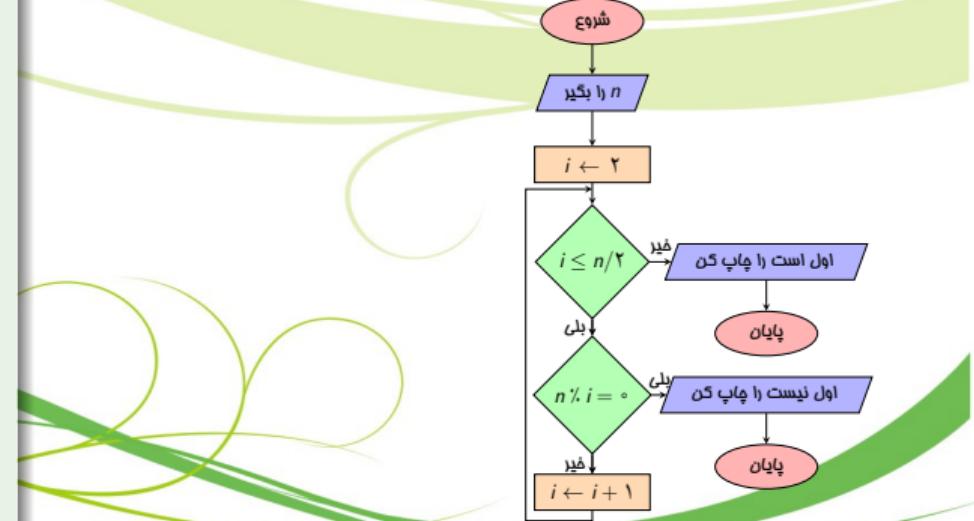
for دستور حلقه

برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i , n;
6     cout<<"Enter n:" ;
7     cin >> n;
8     if (n ==1)
9     {
10        cout<<" Not Prime ";
11        return 0;
12    }
13    for (i=2 ; i <= n/2 ; i++)
14        if (n%i ==0)
15        {
16            cout<<" Not Prime ";
17            return 0;
18        }
19    cout<<" Prime ";
20    return 0;
21 }
```

مثال ۱۱  
برنامه ای بنویسید که یک عدد صحیح بزرگتر از ۱ را از  
خودی دریافت کند و مشخص کند اول است یا غیر.



# ساختار ملکه تکرار

for دستور ملکه تکرار

مثال ۱۲

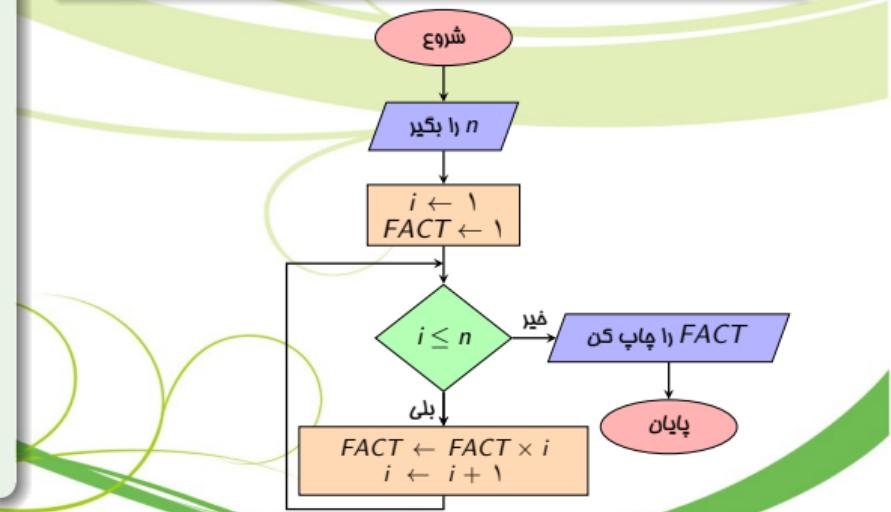
برنامه‌ای بنویسید که یک عدد طبیعی را از ورودی دریافت کند و فاکتوریل آن را محاسبه و پاپ کند.

برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i , n;
6     long int FACT =1 ;
7     cout<<"Enter n:" ;
8     cin >> n;
9     for ( i=1 ; i <=n ; i++)
10        FACT*=i;
11     cout<<" \n Factorial is :" <<
12         FACT;
13     return 0;
}

```



# ساختار ملکه تکرار

## for دستور ملکه تکرار

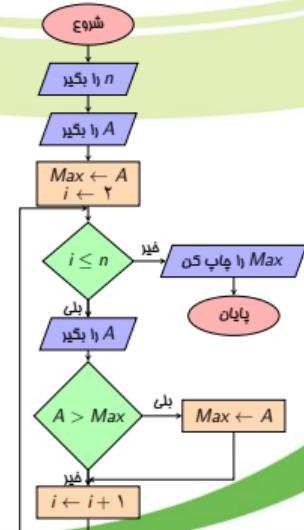
### مثال ۱۳

برنامه‌ای بنویسید که ابتدا عدد طبیعی  $n$  و سپس  $n$  عدد را از ورودی دریافت کند و بزرگترین آنها را محاسبه و پاپ کند.

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i, n;
6     float A, Max;
7     cout<<"Enter n:";
8     cin>>n;
9     cout<<"Enter a number:";
10    cin>>A;
11    Max=A;
12    for(i=2;i<=n ;i++)
13    {
14        cout<<"Enter Next
          number:";
15        cin>>A;
16        if(A > Max)
17            Max=A;
18    }
19    cout<<Max;
20    return 0;
21 }
```

برنامه:



# ساختار ملکه تکرار

for دستور ملکه تکرار

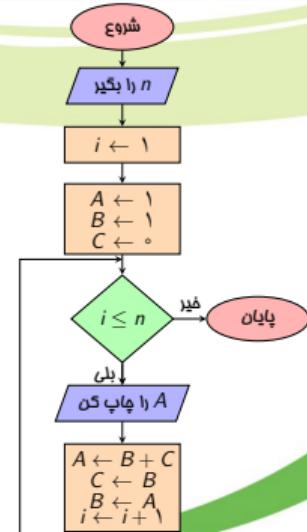
## مثال ۱۴

برنامه‌ای بنویسید که عدد طبیعی  $n$  را دریافت کند و  $n$  جمله اول دنبالهٔ فیبوناچی را محاسبه و چاپ کند.

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i, n, A=1, B=1, C=0;
6     cout<<"Enter n:";
7     cin>>n;
8     for (i=1;i<=n;i++)
9     {
10        cout<<A<<"\n";
11        A=B+C;
12        C=B;
13        B=A;
14    }
15    return 0;
16 }
```

برنامه:



# ساختار ملکه تکرار

for دستور ملکه

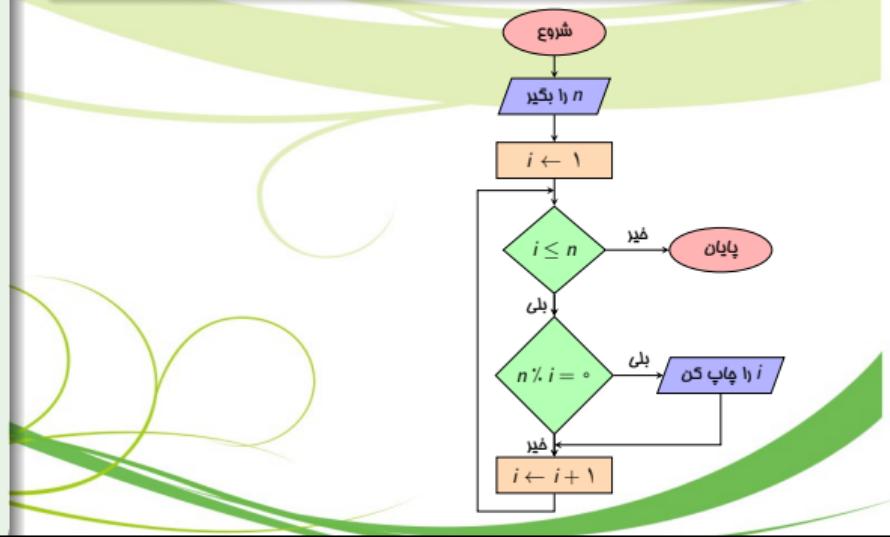
برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i, n;
6     cout<<"Enter n:";
7     cin>>n;
8     for(i=1;i<=n;i++)
9         if(n%i==0)
10            cout<<i<<"\n";
11     return 0;
12 }
```

مثال ۱۵

برنامه ای بنویسید که یک عدد صحیح را دریافت کرده و تمام مقسوم علیه های آن را محاسبه و چاپ کند.



## ساختار ملقه تکرار

دستور ملقة while

دستور ملقة while:

دستور ملقة while با کمی تفاوت نسبت به ملقة for است.

این دستور ملقة، صرفاً شرط تکرار ملقة را دارد و در صورت برقرار بودن شرط دستورات ملقة تکرار می شود.  
به محض اشتباه بودن شرط اجرای برنامه از اولین دستور بعد از بدنه ملقة ادامه فواهد یافت.

while (شرط)

یک دستور یا یک بلوک از دستورات

دانشگاه  
پژوهش  
و تحقیق  
علمی پارسی

## ساختار ملّه تکرار

دستور ملّه while

دستور ملّه while :

دستور ملّه while با کمی تفاوت نسبت به ملّه for است.

این دستور ملّه، صرفاً شرط تکرار ملّه را دارد و در صورت برقرار بودن شرط دستورات ملّه تکرار می شود.

به محض اشتباه بودن شرط اجرای برنامه از اولین دستور بعد از بدنه ملّه ادامه فواهد یافت.

while (شرط)

یک دستور یا یک بلوک از دستورات

دانشگاه  
پژوهش  
و تحقیق  
علمی پارسی

## ساختار ملّه تکرار

دستور ملّه while

دستور ملّه while :

دستور ملّه while با کمی تفاوت نسبت به ملّه for است.

این دستور ملّه، صرفاً شرط تکرار ملّه را دارد و در صورت برقرار بودن شرط دستورات ملّه تکرار می‌شود.  
به محض اشتباہ بودن شرط، اجرای برنامه از اولین دستور بعد از پدنۀ ملّه ادامه خواهد یافت.

while (شرط)

یک دستور یا یک بلوک از دستورات

دانشگاه  
پژوهشی  
دانشگاه  
علمی  
دانشگاه  
پژوهشی

## ساختار ملقه تکرار

دستور ملقه while

دستور ملقه while:

دستور ملقه while با کمی تفاوت نسبت به ملقه for است.

این دستور ملقه، صرفاً شرط تکرار ملقه را دارد و در صورت برقرار بودن شرط دستورات ملقه تکرار می شود.  
به محض اشتباه بودن شرط اجرای برنامه از اولین دستور بعد از پذنه ملقه ادامه فواهد یافت.

## while (شرط)

یک دستور یا یک بلوک از دستورات

دانشگاه  
پژوهشی  
دانشگاه  
علمی  
دانشگاه  
پژوهشی

## ساختار حلقه تکرار

دستور حلقه while

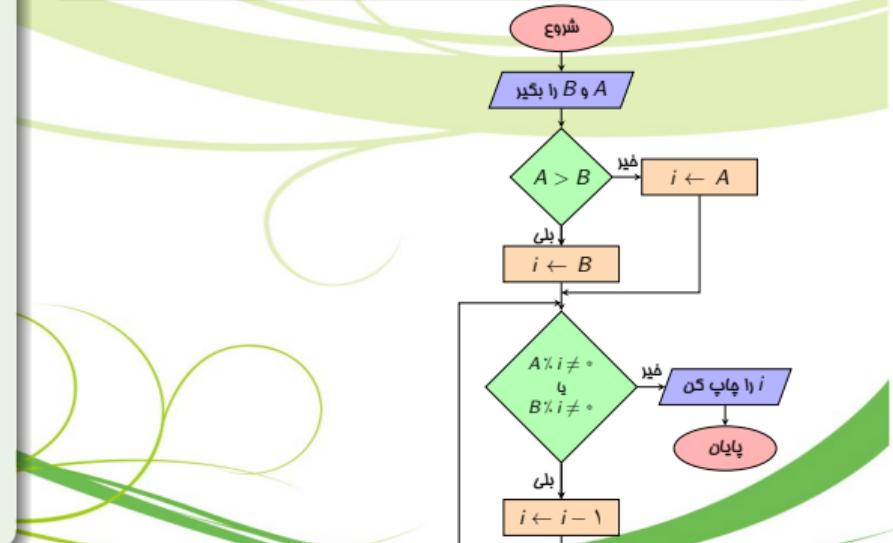
## مثال ۱۶

برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int A,B,i;
6     cout<<"Enter two numbers:";
7     cin>>A>>B;
8     if (A>B)
9         i=B;
10    else
11        i=A;
12    while(A%i !=0 || B%i !=0)
13        i--;
14    cout<<i;
15    return 0;
16 }
```

برنامه ای بنویسید که دو عدد صحیح را از ورودی دریافت کند و ب.م.د آنها را محاسبه و پاپ کند.



## ساختار حلقه تکرار

while دستور حلقه

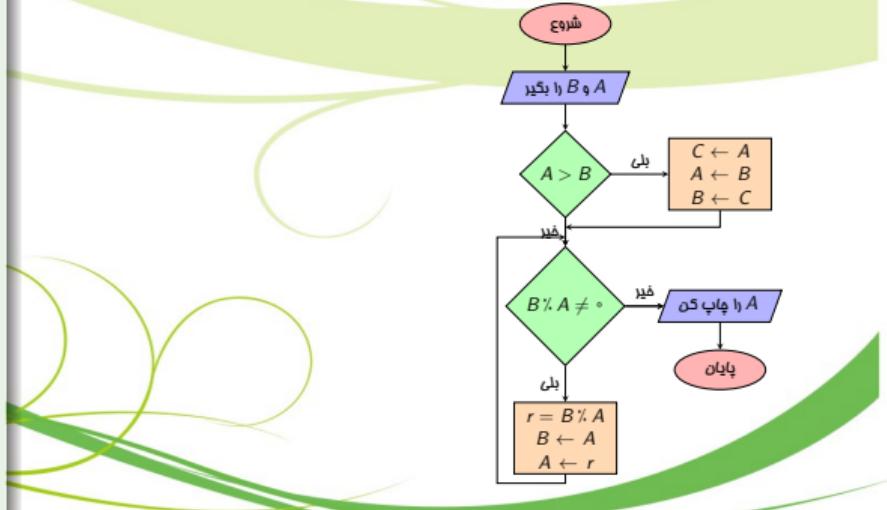
برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int r, A,B,C;
6     cout<<"enter two numbers:";
7     cin>>A>>B;
8     if(A>B)
9     {
10         C=A; A=B; B=C;
11     }
12     while(B%A !=0)
13     {
14         r=B%A; B=A; A=r;
15     }
16     cout<<A;
17     return 0;
18 }
```

مثال ۱۶

برنامه ای بنویسید که دو عدد صحیح را از ورودی دریافت کند و ب.م.د آنها را محاسبه و چاپ کند.



## ساختار ملچه تکرار

دستور ملچه while

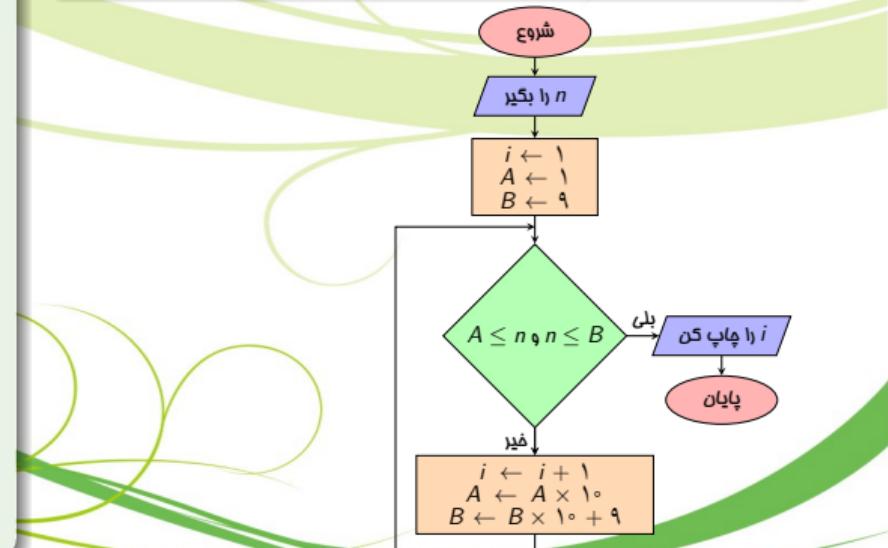
## مثال ۱۷

برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i=1 , n , A=1,B=9;
6     cout<<"Enter n:";
7     cin>>n;
8     while (!(A<=n && n<=B))
9     {
10        i++;
11        A*=10;
12        B=B*10+9;
13    }
14    cout<<i;
15    return 0;
16 }
```

برنامه ای بنویسید که یک عدد طبیعی را دریافت کرده و تعداد ارقام آن را محاسبه و چاپ کند.



## ساختار ملچه تکرار

دستور ملچه while

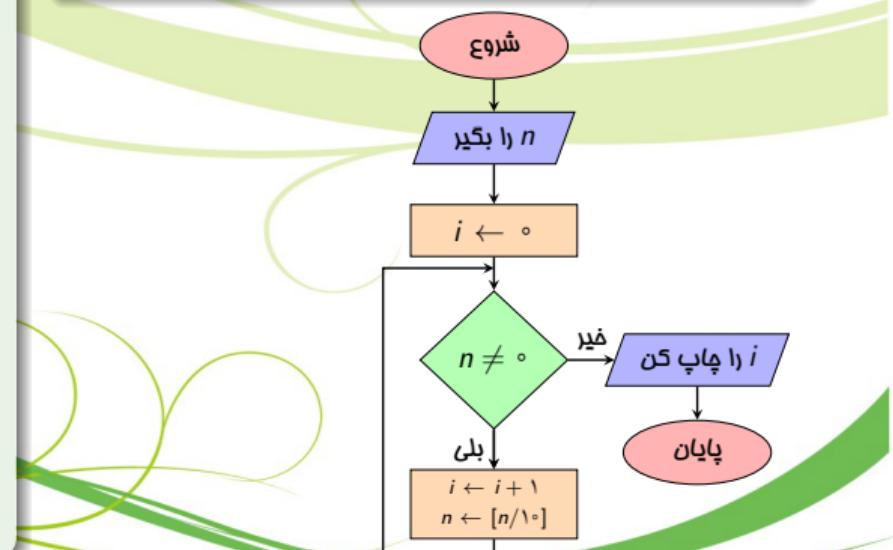
## مثال ۱۷

برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i=0, n ;
6     cout<<"Enter n:" ;
7     cin>>n;
8     while(n!=0)
9     {
10         i++;
11         n/=10;
12     }
13     cout<<i;
14     return 0;
15 }
```

برنامه‌ای بنویسید که یک عدد طبیعی را دریافت کرده و تعداد ارقام آن را محاسبه و پاپ کند.



# ساختار حلقه تکرار

## do .. while

### دستور حلقه do ..

در دستورات قبلی حلقه، شرط تکرار حلقه در ابتدای حلقه بررسی می‌شد و در صورت برقار بودن آن، بدنه حلقه تکرار می‌شد.

اما در برخی الگوریتم‌ها، شرط تکرار حلقه در انتهای بررسی می‌شود. برای پیاده‌سازی چنین الگوریتم‌هایی از حلقه تکرار می‌کنیم.

do

یک دستور یا یک بلوک از دستورات  
while(شرط);

دقت کنید که در انتهای این دستور، سمتی کالن نیاز است.

عملکرد دستور while ... do مشابه دستور while است با این تفاوت که شرط حلقه در انتهای اجرای بدنه حلقه بررسی می‌شود.



# ساختار حلقه تکرار

## do .. while

### دستور حلقه do ..

در دستورات قبلی حلقه، شرط تکرار حلقه در ابتدای حلقه بررسی می‌شد و در صورت برقار بودن آن، بدنه حلقه تکرار می‌شد.

اما در برفی الگوریتم‌ها، شرط تکرار حلقه در انتهای بررسی می‌شود. برای پیاده‌سازی چنین الگوریتم‌هایی از حلقه ... do استفاده می‌کنیم.

do

یک دستور یا یک بلوک از دستورات  
while(شرط);

دقت کنید که در انتهای این دستور، سمی کالن نیاز است.

عملکرد دستور ... while مشابه دستور while است با این تفاوت که شرط حلقه در انتهای اجرای بدنه حلقه بررسی می‌شود.



## ساختار حلقه تکرار

do .. while دستور حلقه

دستور حلقه do ..

در دستورات قبلی حلقه، شرط تکرار حلقه در ابتدای حلقه بررسی می‌شد و در صورت برقار بودن آن، بدنه حلقه تکرار می‌شد.

اما در برفی الگوریتم‌ها، شرط تکرار حلقه در انتهای بررسی می‌شود. برای پیاده‌سازی پنین الگوریتم‌هایی از حلقه ... do ... while استفاده می‌کنیم.

do

یک دستور یا یک بلوک از دستورات

while();  
(شرط);

دقت کنید که در انتهای این دستور، سمی کالن نیاز است.

عملکرد دستور ... while مشابه دستور while است با این تفاوت که شرط حلقه در انتهای اجرای بدنه حلقه بررسی می‌شود.



## ساختار حلقه تکرار

do .. while دستور حلقه

دستور حلقه do ..

در دستورات قبلی حلقه، شرط تکرار حلقه در ابتدای حلقه بررسی می‌شد و در صورت برقار بودن آن، بدنه حلقه تکرار می‌شد.

اما در برفی الگوریتم‌ها، شرط تکرار حلقه در انتهای بررسی می‌شود. برای پیاده‌سازی پنین الگوریتم‌هایی از حلقه ... do ... while استفاده می‌کنیم.

do

یک دستور یا یک بلوک از دستورات

while();  
(شرط);

دقت کنید که در انتهای این دستور، سمی کالن نیاز است.

عملکرد دستور ... while مشابه دستور do ... while است با این تفاوت که شرط حلقه در انتهای

اجرای بدنه حلقه بررسی می‌شود.

دانشکده علم رسانی  
دانشکده علم رسانی

# ساختار حلقه تکرار

do .. while

دستور حلقه do ..

در دستورات قبلی حلقه، شرط تکرار حلقه در ابتدای حلقه بررسی می‌شد و در صورت بروقراز بودن آن، بدنه حلقه تکرار می‌شد.

اما در برقی الگوریتم‌ها، شرط تکرار حلقه در انتهای بررسی می‌شود. برای پیاده‌سازی پنین الگوریتم‌هایی از حلقه ... do ... while استفاده می‌کنیم.

do

یک دستور یا یک بلوک از دستورات

while(شرط);

دقت کنید که در انتهای این دستور، سمتی کالن نیاز است.

عملکرد دستور while ... do مشابه دستور while است با این تفاوت که شرط حلقه در انتهای اجرای بدنه حلقه بررسی می‌شود.



دانشگاه  
علم پژوهی

ساختار ملچه تکرار  
do .. while دستور ملچه

مثال:

```
1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     float number, sum = 0.0;
6     do {
7         cout<<"Enter a number: ";
8         cin>>number;
9         sum += number;
10    }while(number != 0.0);
11    cout<<"Total sum = "<<sum;
12    return 0;
13 }
```

10  
11  
12  
13

آرایه‌ها نیز مشابه سایر متغیرها باید قبل از استفاده تعریف شوند.

اضافه بر نوع آرایه، باید فضای مورد نیاز برای آرایه نیز مشخص شود. مثال: int A[100]; می‌توان چند آرایه را که از یک نوع هستند در یک دستور تعریف قرار داد و با کاما از یکدیگر جدا کرد. همچنین امکان تعریف متغیرهای معمول و آرایه‌ها در یک دستور تعریف امکان‌پذیر است. تعداد درایه‌های یک آرایه باید یک عدد طبیعی باشد و امکان متغیر بودن این مقدار نیست.

اندیس آرایه از صفر شروع می‌شود، یعنی [۰] A درایه چهارم آرایه A است.

برای اجاع به درایه یک آرایه، بررسی نمی‌شود که آیا درایه‌ای با این شماره در آرایه وجود دارد یا خیر. لذا برنامه‌نویس باید مواظب باشد در برنامه چنین اتفاقی نیفتد. این موضوع یکی از کانون‌های فطای منطقی در برنامه‌نویسی است.



آرایه‌ها نیز مشابه سایر متغیرها باید قبل از استفاده تعریف شوند.

اضافه بر نوع آرایه، باید فضای مورد نیاز برای آرایه نیز مشخص شود. مثال: `int A[100];` می‌توان چند آرایه را که از یک نوع هستند در یک دستور تعریف قرار داد و با کاما از یکدیگر جدا کرد. همچنین امکان تعریف متغیرهای معمول و آرایه‌ها در یک دستور تعریف امکان‌پذیر است. تعداد درایه‌های یک آرایه باید یک عدد طبیعی باشد و امکان متغیر بودن این مقدار نیست.

اندیس آرایه از صفر شروع می‌شود، یعنی `[۳]A` درایه چهارم آرایه `A` است.

برای اجاع به درایه یک آرایه، بررسی نمی‌شود که آیا درایه‌ای با این شماره در آرایه وجود دارد یا خیر. لذا برنامه‌نویس باید مواظب باشد در برنامه چنین اتفاقی نیفتد. این موضوع یکی از کانون‌های فطای منطقی در برنامه‌نویسی است.



آرایه‌ها نیز مشابه سایر متغیرها باید قبل از استفاده تعریف شوند.

اضافه بر نوع آرایه، باید فضای مورد نیاز برای آرایه نیز مشخص شود. مثال: `int A[100];` می‌توان چند آرایه را که از یک نوع هستند در یک دستور تعریف قرار داد و با کاما از یکدیگر جدا کرد. همچنین امکان تعریف متغیرهای معمول و آرایه‌ها در یک دستور تعریف امکان‌پذیر است. تعداد درایه‌های یک آرایه باید یک عدد طبیعی باشد و امکان متغیر بودن این مقدار نیست.

اندیس آرایه از صفر شروع می‌شود، یعنی `[۳]A` درایه چهارم آرایه `A` است.

برای اجاع به درایه یک آرایه، بررسی نمی‌شود که آیا درایه‌ای با این شماره در آرایه وجود دارد یا خیر. لذا برنامه‌نویس باید مواظب باشد در برنامه چنین اتفاقی نیفتد. این موضوع یکی از کانون‌های فطای منطقی در برنامه‌نویسی است.



آرایه‌ها نیز مشابه سایر متغیرها باید قبل از استفاده تعریف شوند.

اضافه بر نوع آرایه، باید فضای مورد نیاز برای آرایه نیز مشخص شود. مثال: `int A[100];` می‌توان چند آرایه را که از یک نوع هستند در یک دستور تعریف قرار داد و با کاما از یکدیگر جدا کرد. همچنین امکان تعریف متغیرهای معمول و آرایه‌ها در یک دستور تعریف امکان‌پذیر است. تعداد درایه‌های یک آرایه باید یک عدد طبیعی باشد و امکان متغیر بودن این مقدار نیست.

اندیس آرایه از صفر شروع می‌شود، یعنی `[۳]A` درایه چهارم آرایه `A` است.

برای اجاع به درایه یک آرایه، بررسی نمی‌شود که آیا درایه‌ای با این شماره در آرایه وجود دارد یا خیر. لذا برنامه‌نویس باید مواظب باشد در برنامه چنین اتفاقی نیفتد. این موضوع یکی از کانون‌های فطای منطقی در برنامه‌نویسی است.



آرایه‌ها نیز مشابه سایر متغیرها باید قبل از استفاده تعریف شوند.

اضافه بر نوع آرایه، باید فضای مورد نیاز برای آرایه نیز مشخص شود. مثال: `int A[100];` می‌توان چند آرایه را که از یک نوع هستند در یک دستور تعریف قرار داد و با کاما از یکدیگر جدا کرد. همچنین امکان تعریف متغیرهای معمول و آرایه‌ها در یک دستور تعریف امکان‌پذیر است. تعداد درایه‌های یک آرایه باید یک عدد طبیعی باشد و امکان متغیر بودن این مقدار نیست.

اندیس آرایه از صفر شروع می‌شود، یعنی `[۳]A` درایه پنجم آرایه `A` است.

برای اجاع به درایه یک آرایه، بررسی نمی‌شود که آیا درایه‌ای با این شماره در آرایه وجود دارد یا خیر. لذا برنامه‌نویس باید مواظب باشد در برنامه چنین اتفاقی نیفتد. این موضوع یکی از کانون‌های فطای منطقی در برنامه‌نویسی است.



آرایه‌ها نیز مشابه سایر متغیرها باید قبل از استفاده تعریف شوند.

اضافه بر نوع آرایه، باید فضای مورد نیاز برای آرایه نیز مشخص شود. مثال: `int A[100];` می‌توان چند آرایه را که از یک نوع هستند در یک دستور تعریف قرار داد و با کاما از یکدیگر جدا کرد.

همچنین امکان تعریف متغیرهای معمول و آرایه‌ها در یک دستور تعریف امکان‌پذیر است. تعداد درایه‌های یک آرایه باید یک عدد طبیعی باشد و امکان متغیر بودن این مقدار نیست.

اندیس آرایه از صفر شروع می‌شود، یعنی `[۳]A` درایه پنجم آرایه `A` است.

برای ارجاع به درایه یک آرایه، بررسی نمی‌شود که آیا درایه‌ای با این شماره در آرایه وجود دارد یا نیز. لذا برنامه‌نویس باید مواظب باشد در برنامه چنین اتفاقی نیفتد. این موضوع یکی از کانون‌های خطاها منطقی در برنامه‌نویسی است.



دانشگاه  
بهشتی  
دانشکده علم پزشکی

## مثال ۱۸

الگوریتمی بنویسید که ابتدا عدد  $n$  و سپس  $n$  عدد را بگیرد و آن‌ها را به ترتیب عکس داده شده، چاپ کند.

## الگوریتم:

- |    |                      |
|----|----------------------|
| ۱  | شروع                 |
| ۲  | $n$ را بگیر          |
| ۳  | $i \leftarrow 0$     |
| ۴  | اگر $n < i$ آنگاه    |
| ۵  | $A[i]$ را بگیر       |
| ۶  | $i \leftarrow i + 1$ |
| ۷  | به مرحله ۴ برو       |
| ۸  | پایان اگر            |
| ۹  | اگر $0 > i$ آنگاه    |
| ۱۰ | $i \leftarrow i - 1$ |
| ۱۱ | $A[i]$ را چاپ کن     |
| ۱۲ | به مرحله ۹ برو       |
| ۱۳ | پایان اگر            |
| ۱۴ | پایان                |



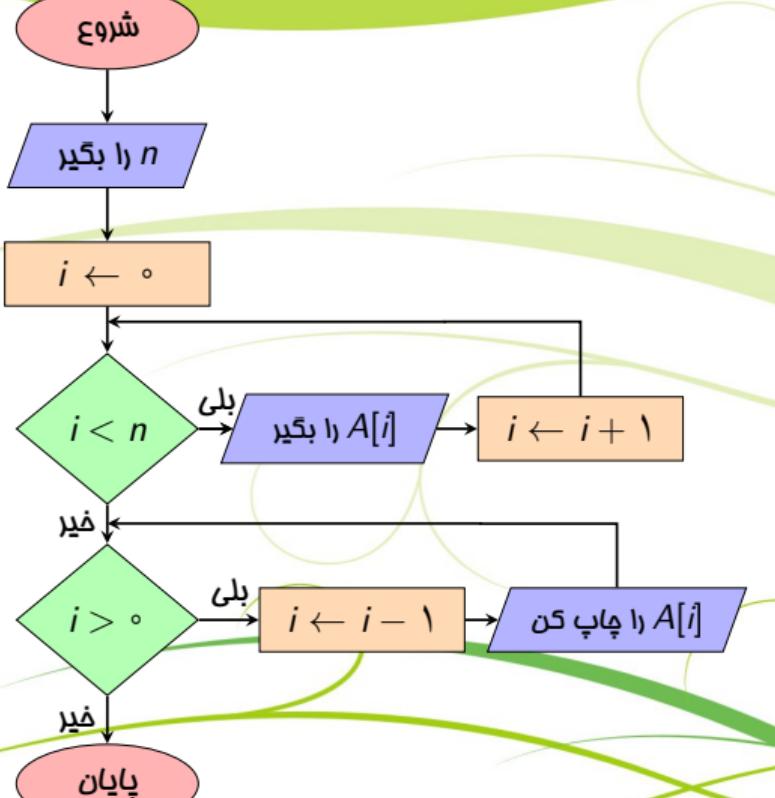
دانشگاه  
یزد  
دانشگاه  
علم پزشکی

## مثال ۱۸

الگوریتمی بنویسید که ابتدا عدد  $n$  و سپس  $n$  عدد را بگیرد و آن‌ها را به ترتیب عکس داده شده، پاپ کند.

## الگوریتم:

- |    |                      |
|----|----------------------|
| ۱  | شروع                 |
| ۲  | $n$ را بگیر          |
| ۳  | $i \leftarrow 0$     |
| ۴  | اگر $i < n$ آنگاه    |
| ۵  | $A[i]$ را بگیر       |
| ۶  | $i \leftarrow i + 1$ |
| ۷  | به مرحله ۴ برو       |
| ۸  | پایان اگر            |
| ۹  | اگر $i > 0$ آنگاه    |
| ۱۰ | $i \leftarrow i - 1$ |
| ۱۱ | $A[i]$ را پاپ کن     |
| ۱۲ | به مرحله ۹ برو       |
| ۱۳ | پایان اگر            |
| ۱۴ | پایان                |



## برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i, n;
6     float A[200];
7     cout<<"Enter An Integer
8         (<=200):";
9     cin>>n;
10    for (i=0; i<n;i++)
11    {
12        cout<<"\n Enter Next
13            number:";
14        cin>>A[i];
15    }
16    while(i>0)
17    {
18        i--;
19        cout<<A[i]<<"," ;
20    }
21    return 0;
22 }
```

## مثال ۱۸

الگوریتمی بنویسید که ابتدا عدد  $n$  و سپس  $n$  عدد را بگیرد و آن‌ها را به ترتیب عکس داده شده، چاپ کند.

## الگوریتم:

۱	شروع
۲	$n$ را بگیر
۳	$i \leftarrow 0$
۴	اگر $i < n$ $\leftarrow$ آنگاه
۵	$A[i]$ را بگیر
۶	$i \leftarrow i + 1$
۷	به مرحله ۴ برو
۸	پایان اگر
۹	اگر $i > n$ $\leftarrow$ آنگاه
۱۰	$i \leftarrow i - 1$
۱۱	$A[i]$ را چاپ کن
۱۲	به مرحله ۹ برو
۱۳	پایان اگر
۱۴	پایان

## مثال ۱۹

الگوریتم و فلوچارٹ ارائه کنید که ابتدا  $n$  و سپس  $n$  نمره را دریافت کرده و تعداد نمرات بالاتر از میانگین کلاس را چاپ کند.

### الگوریتم:

۱	شروع	
۲	$n$ را بگیر	
۳	$Sum \leftarrow 0$ , $i \leftarrow 0$	
۴	اگر $n < i$ آنگاه	
۵	$A[i]$ را بگیر	
۶	$Sum \leftarrow Sum + A[i]$	
۷	$i \leftarrow i + 1$	
۸	به مرحله ۴ برو	
۹	پایان اگر	
۱۰	$Ave \leftarrow Sum/n$	
۱۱	$counter \leftarrow 0$ , $i \leftarrow 0$	

۱۲	اگر $n < i$ آنگاه	
۱۳	$A[i] > Ave$ اگر	
۱۴	$counter \leftarrow counter + 1$	
۱۵	پایان اگر	
۱۶	$i \leftarrow i + 1$	
۱۷	به مرحله ۱۲ برو	
۱۸	پایان اگر	
۱۹	$counter$ را چاپ کن	
۲۰	پایان	



دانشگاه  
یزد

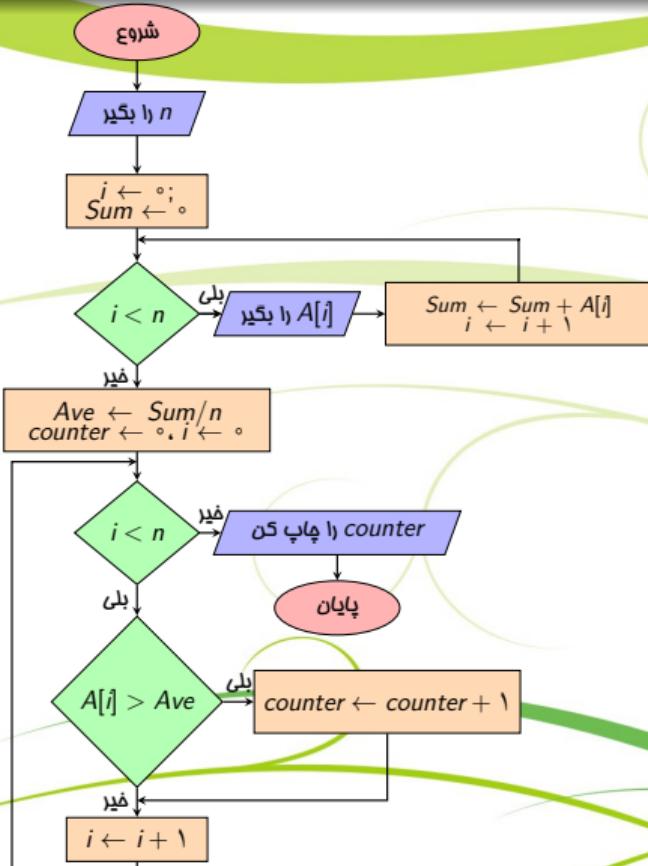


## مثال ۱۹

الگوریتم و فلوچارتی ارائه کنید که ابتدا  $n$  و سپس  $n$  نمره را دریافت کرده و تعداد نمرات بالاتر از میانگین کلاس را چاپ کند.

## الگوریتم:

۱	شروع	
۲	$n$ را بگیر	
۳	$i \leftarrow 0$ ; $Sum \leftarrow 0$	
۴	$i < n$ بله	اگر $i < n$ آنگاه
۵	$A[i]$ را بگیر	$A[i] > Ave$ اگر
۶	$Sum \leftarrow Sum + A[i]$	$counter \leftarrow counter + 1$
۷		
۸	$Ave \leftarrow Sum/n$	
۹	$counter \leftarrow 0$ , $i \leftarrow 0$	
۱۰	$i < n$ بله	اگر $i < n$ آنگاه
۱۱	پایان	$A[i] > Ave$ پایان اگر
۱۲	$i \leftarrow i + 1$	$i \leftarrow i + 1$
۱۳		
۱۴	$Ave < Sum/n$	$Sum \leftarrow Sum + A[i]$
۱۵	$counter \leftarrow counter + 1$	$i \leftarrow i + 1$
۱۶		
۱۷	$i < n$ بله	به مرحله ۱۲ برو
۱۸		پایان اگر
۱۹	$Ave \leftarrow Sum/n$	$counter \leftarrow counter + 1$
۲۰	پایان	$i \leftarrow i + 1$



## برنامه:

```

1 #include <iostream>
2 using namespace std;
3 int main()
4 {
5     int i, n, counter;
6     float A[200], Sum, Ave;
7     cout<<"Enter An Integer
8         (<=200):";
9     cin>>n;
10    for (i=0; i<n;i++)
11    {
12        cout<<"\n Enter Next
13            number:";
14        cin>>A[i];
15        Sum+=A[i];
16    }
17    Ave=Sum/n;
18    counter=0;
19    for(i=0; i<n; i++)
20    {
21        if (A[i]>Ave)
22            counter++;
23    }
24    cout<<"# of grades above
25        average:"<<counter;
26    return 0;
27 }
```

مثال ۱۹

الگوریتم و فلوچارٹی ارائه کنید که ابتدا  $n$  و سپس  $n$  نمره را دریافت کرده و تعداد نمرات بالاتر از میانگین کلاس را چاپ کند.

الگوریتم:

۱ شروع	۱۲ اگر $n < i$ آنگاه	۲ $n$ را بگیر
۳ $Sum \leftarrow 0$	۱۳ $Ave \leftarrow Sum / n$	۴ $i \leftarrow 0$
۵ $i \leftarrow i + 1$	۱۴ $counter \leftarrow 0$	۶ $counter \leftarrow counter + 1$
۷ $if (A[i] > Ave)$	۱۵ $if (A[i] > Ave)$	۸ $Sum \leftarrow Sum + A[i]$
۹ $    counter++;$	۱۶ $i \leftarrow i + 1$	۱۰ $i \leftarrow i + 1$
۱۱ $cout \leftarrow "# of grades above$	۱۷ $if (Ave < Sum / n)$	۱۱ $counter \leftarrow counter + 1$
۱۲ $average:" \ll counter;$	۱۸ $cout \leftarrow cout + "Counter = " \ll counter$	۱۲ $cout \leftarrow cout + "Counter = " \ll counter$
۱۳ $return 0;$	۱۹ $cout \leftarrow cout + "Counter = " \ll counter$	۱۳ $cout \leftarrow cout + "Counter = " \ll counter$
۱۴ $}$	۲۰ $cout \leftarrow cout + "Counter = " \ll counter$	۱۴ $cout \leftarrow cout + "Counter = " \ll counter$



## آرایه‌های دو بعدی

آرایه‌های دو بعدی، (وش تعریف و استفاده مشابه آرایه‌های یک بعدی است. تنها تفاوت آنها این است که در تعریف آرایه، باید تعداد سطر و ستون‌های آنها مشخص شود.

مثال: برای تعریف آرایه دو بعدی Matrix از نوع اعشاری و با ۳ سطر و ۷ ستون، دستور زیر استفاده می‌شود:

```
float Matrix [3][7];
```

برای دسترسی به هر درایه باید شماره سطر و ستون آنها بیان شود. برای ارجاع به درایه سطر اول و ستون چهارم از Matrix [0][3] استفاده می‌کنیم.



دانشگاه  
علوم پیامبری  
شاهرود



## آرایه‌های دو بعدی

آرایه‌های دو بعدی، (وش تعریف و استفاده مشابه آرایه‌های یک بعدی است. تنها تفاوت آنها این است که در تعریف آرایه، باید تعداد سطر و ستون‌های آنها مشخص شود.

مثال: برای تعریف آرایه دو بعدی Matrix از نوع اعشاری و با ۳ سطر و ۷ ستون، دستور زیر استفاده می‌شود:

```
float Matrix [3][7];
```

برای دسترسی به هر درایه باید شماره سطر و ستون آنها بیان شود. برای ارجاع به درایه سطر اول و ستون چهارم از Matrix [0][3] استفاده می‌کنیم.



دانشگاه  
علوم پیامبری  
شاهرود



## آرایه‌های دو بعدی

آرایه‌های دو بعدی، (وش تعریف و استفاده مشابه آرایه‌های یک بعدی است. تنها تفاوت آنها این است که در تعریف آرایه، باید تعداد سطر و ستون‌های آنها مشخص شود.

مثال: برای تعریف آرایه دو بعدی Matrix از نوع اعشاری و با ۳ سطر و ۷ ستون، دستور زیر استفاده می‌شود:

```
float Matrix [3][7];
```

برای دسترسی به هر درایه باید شماره سطر و ستون آنها بیان شود. برای ارجاع به درایه سطر اول و ستون چهارم از Matrix [0][3] استفاده می‌کنیم.



دانشگاه  
علم پژوهی



بآرزوی موفقیت

و بهروزی

[mfarshi@yazd.ac.ir](mailto:mfarshi@yazd.ac.ir)