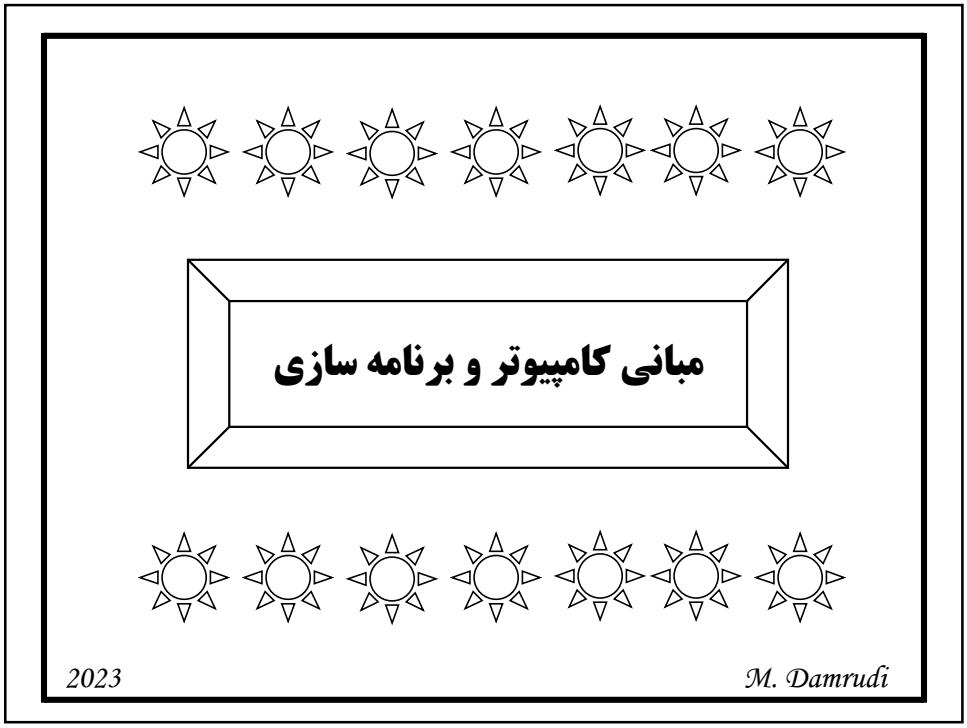
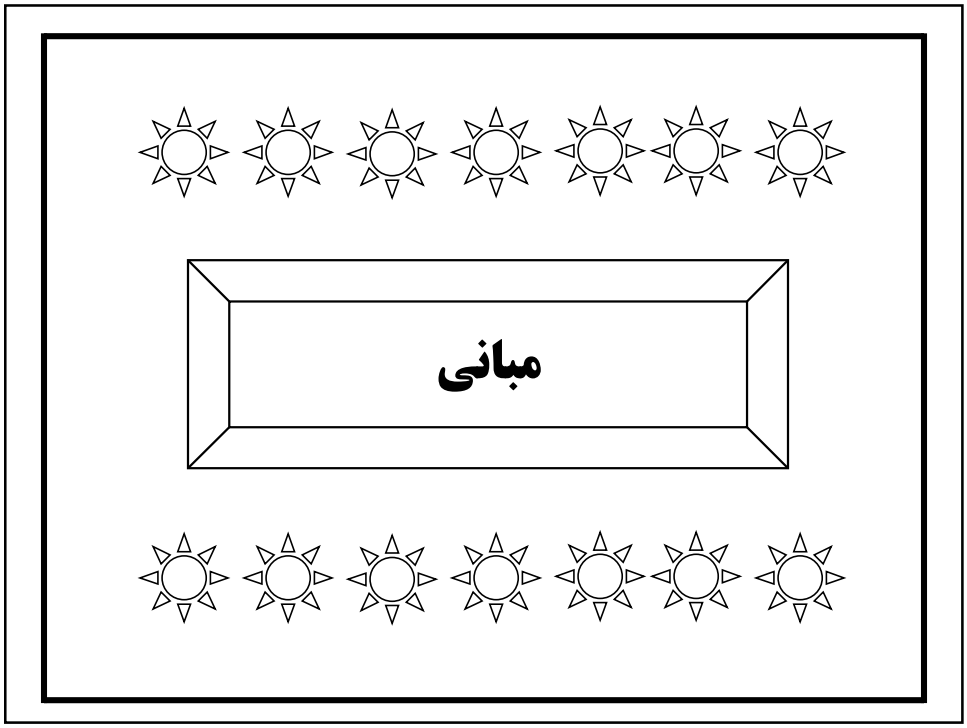

مبانی کامپیوتر و برنامه سازی
2023
M. Damrudi


مبانی

دستگاهها

تک منظوره (Single purpose)
تلفن (یک کاربرد خاص)

همه منظوره (General purpose)
کامپیوتر (قادر به انجام کارهای مختلف)

کامپیوتر چیست؟

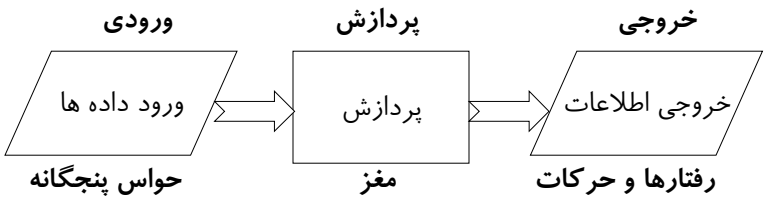
کامپیوتر از قطعات الکترونیکی و الکترومکانیکی تشکیل شده است و دارای ویژگیهای زیر است:

۱- قابل برنامه ریزی است.

۲- در مقابل عملیات ما عکس العمل نشان می دهد.

۳- دارای حافظه است.

۴- قادر به پردازش داده ها است.



مزایای کامپیوتر

۱- سرعت انجام عملیات
۲- صحت و دقت در انجام کار
۳- قابلیت اطمینان
۴- قابلیت ذخیره سازی حجم انبوهی از اطلاعات در فضایی کم

مزایای انسان نسبت به کامپیوتر

۱- خلاقیت
۲- هوشمندی
۳- ابتکار
۴- مدیریت انعطاف پذیر

کاربردهای کامپیوتر

۱- علمی و تحقیقاتی (کامپیوتر در تمام رشته ها، رباتهای پزشک، ...)

۲- صنعتی و مهندسی (داروسازی، خودروسازی، غذایی، ...)

۳- تجاری و خدماتی (قبوض، انبارداری، تجارت الکترونیکی، ...)

۴- پزشکی (رباتهای پزشک، تولید اندامهای خودکار، ...)

۵- آموزشی (آموزش از راه دور(دانشگاه مجازی)، آموزش خلبانی،...)

۶- هنری (آگهی ها، کاتالوگ، کارت ویزیت، فیلم علمی-تخیلی، ...)

تاریخچه تکاملی کامپیوتر

نسل: تحول تکنولوژی کامپیوتر در مقاطع مختلف زمانی است که مرتبا قطعات الکترونیکی کوچکتر، سریعتر، قابلیت اطمینان بالاتر وقیمت تولیدشان پایین‌تر می‌شود.

۱- نسل صفر – قطعات مکانیکی (چرتکه،ماشین حساب پاسکال)

۲- نسل اول – اولین قطعه الکتریکی (لامپ خلا)

۳- نسل دوم – اولین قطعه الکترونیکی (ترانزیستور)

۴- نسل سوم – مدارات مجتمع، عناصر الکترونیکی(IC)

۵- نسل چهارم – بیش ازصدها هزار قطعه(IC با تراکم خیلی زیاد)

۶- نسل پنجم – چیپ های هوشمند (کامپیوترهای هوشمند)

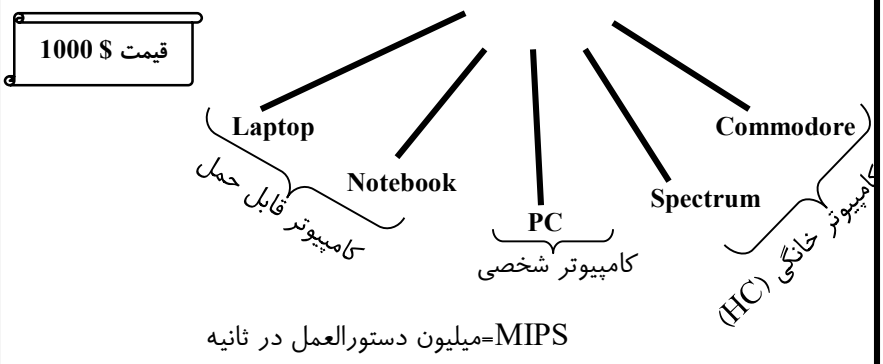
۷- نسل ششم – شبیه سازی عملکرد مغز(کپی برداری از مغز انسان)

انواع کامپیوترها

بر اساس افزایش قدرت پردازش- میزان حافظه- قیمت میانگین

کوچکترین و متداول ترین نوع کامپیوترها

۱- ریزکامپیوترها (Micro computers)



MIPS=میلیون دستورالعمل در ثانیه

انواع کامپیوترها

PC: به دلیل کاهش قیمت، شبکه های پر قدرت کامپیوتری را ایجاد می کنند تا اطلاعات و تجهیزات کامپیوترهای موجود را اشتراکی به کار ببرند.

Notebook و laptop: افزایش کارایی و کاهش قیمت

نوت بوک های جدید با استفاده از ۲ باطری حداکثر با۱۰ ساعت شارژ کار می کنند.

پردازنده های جدید شش و هشت هسته را به کار می برند.

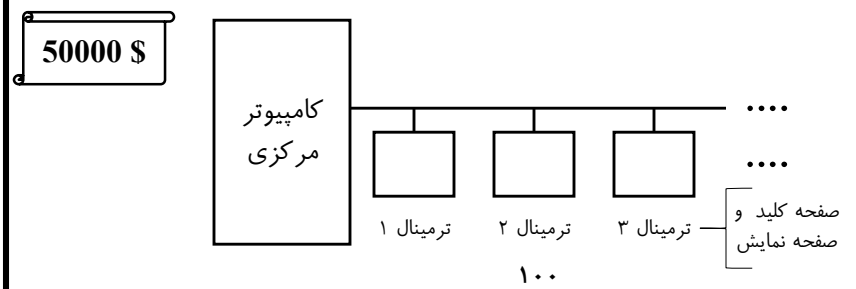
نوت بوک‌های اولیه دارای پردازنده وکارت‌گرافیک ضعیف تری نسبت به لپ‌تاپ ها بودند اما وزن سبک تر و شارژ طولانی تری داشتند اما با گذشت زمان مرز بین آنها کم رنگ تر شد.

انواع کامپیوترها

بر اساس افزایش قدرت پردازش- میزان حافظه- قیمت میانگین

حجم اطلاعات قابل پردازش و تنوع کارها متوسط است. از یک کامپیوتر مرکزی و تعدادی ترمینال (حدود ۱۰۰ و بالاتر) متصل به آن تشکیل شده است. ترمینال: مجموعه صفحه کلید و صفحه نمایش (دانشگاهی، دولتی، تجاری)

۲- کامپیوترهای کوچک (Mini computers)

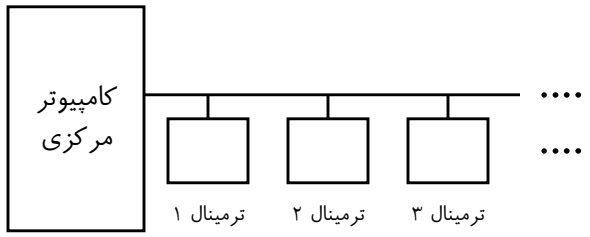


انواع کامپیوترها

حجم اطلاعات قابل پردازش و تنوع کارها بسیار زیاد است. ساختار مشابه کامپوترهای کوچک است ولی قدرت پردازش بیشتر و تعداد ترمینال های قابل اتصال تا ۵۰۰ عدد امکان دارد. (دانشگاه ها، وزارتخانه ها،واحدهای بزرگ تجاری)
IBM 370

۳- کامپیوترهای بزرگ (Mainframe computers)

1000000 $



انواع کامپیوترها

سریعترین، قدرتمندترین، بزرگترین و گرانترین نوع کامپیوترها هستند. در پروژه های عظیم علمی، تحقیقاتی، نظامی، فضایی، هسته ای و دفاعی به کار می روند. سریعترین ابر رایانه جهان جاگوار است جهت انجام تحقیقات علمی مانند تغییرات جوی و مواد فضایی غیر قابل مشاهده که قادر به پردازش یک کادریلیون 10^{15}محاسبه ریاضی در ثانیه است.

۴- ابر کامپیوترها (Super computers)

>10000000 $

IBM Blue Gene
324000 MIPS
میلیارد دستورالعمل در ثانیه

CRAY-xt3
54000 MIPS
میلیارد دستورالعمل در ثانیه

امکان پردازش موازی فرایندها: امکان اجرای موازی قسمتهایی از یک برنامه توسط پردازنده های مختلف و سپس ادغام نتایج آنها با یکدیگر را امکان پذیر می کنند.

ربات

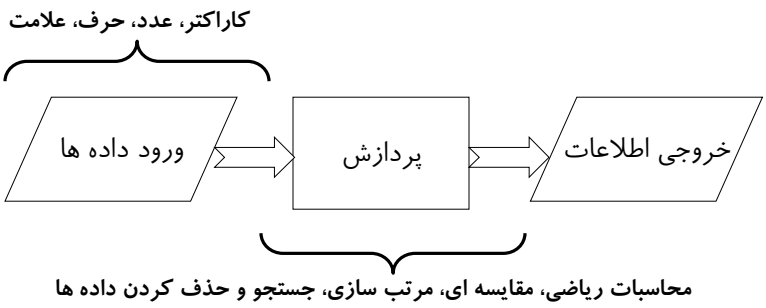
ربات ماشین هوشمندی است که می تواند در شرایط خاصی کار تعریف شده ای را انجام دهد و قادر به تصمیم گیری در شرایط متفاوت است. ربات دارای سه قسمت اصلی است:

- مغز که معمولا یک کامپیوتر است
- محرک ها و بخش مکانیکی شامل موتور، پیستون، تسمه، چرخ، چرخ دنده
- سنسور که می تواند از انواع بینایی، صوتی، تعیین دما، تشخیص نور، تماسی یا حرکتی باشد

ربات های مسیریاب، آتش نشان، مین یاب، امدادگر، فوتبالیست، جنگجو

طبقه بندی علوم کامپیوتر

سیستم مجموعه عناصر منظم و مرتبط با هم است که برای رسیدن به هدف مشخصی بصورت هماهنگ با یکدیگر در تعامل هستند.



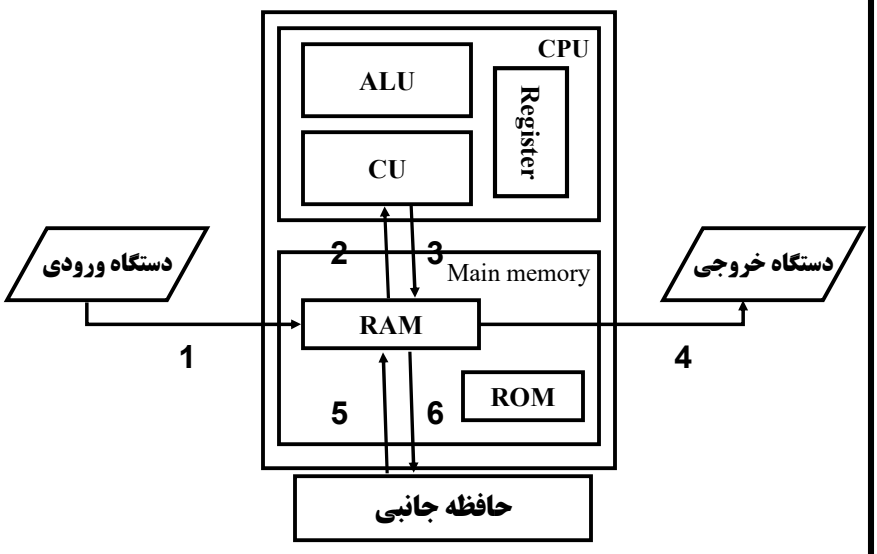
طبقه بندی علوم کامپیوتر

۱- سخت‌افزار (Hardware) : قطعات فیزیکی و قابل‌لمس یک سیستم کامپیوتری از مدارات الکترونیکی و بخشهای مکانیکی تشکیل شده است. (خازن و مقاومت،.)

۲- نرم‌افزار(Software): مدیریت، هدایت،کنترل واستفاده ازسخت‌افزار بوسیله نرم‌افزار عملی می‌شود. مجموعه‌ای از دستورالعملها که به ترتیب خاصی توسط برنامه‌نویس نوشته می‌شود.

۳- میان‌افزار (Firmware): قطعه‌ای سخت‌افزاری است که در آن برنامه‌های مربوط به تست و راه‌اندازی کامپیوتر ذخیره شده است(مثل ROM BIOS).

سیستم کامپیوتر



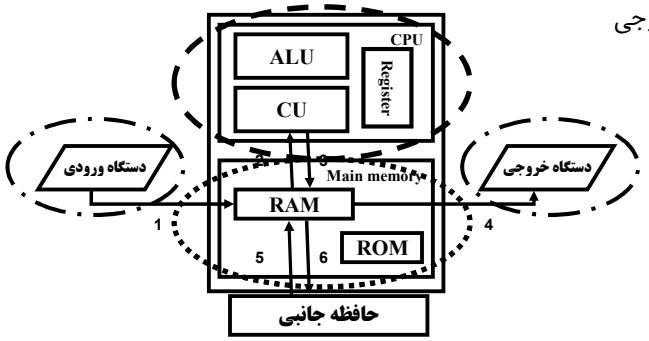
اجزای اصلی سخت‌فزار

۱- ریزپردازنده

۲- حافظه

۳- دستگاههای ورودی

۴- دستگاههای خروجی



ریزپردازنده (واحد پردازش مرکزی)

مغز سیستم

۱– واحد حساب و منطق (ALU) : انجام عملیاتهای محاسباتی و منطقی

۲– واحد کنترل(CU): انجام کار تمام قسمتهای کامپیوتر را کنترل می‌کند.

۳– ثبات (Register): حافظه های سریع داخل CPU **هستند که جهت ذخیره موقتی داده‌ها و آدرسها به‌کار می‌رود.**

روند پردازش داده ها و دستورالعمل ها

۱- داده ها و دستورالعمل ها از طریق دستگاههای ورودی وارد حافظه اصلی می شود.

۲- داده ها و دستورالعمل ها از حافظه RAM جهت پردازش به ریزپردازنده ارسال می شوند و پس از تشخیص عملیات موردنظر توسط واحدکنترل بوسیله واحدحساب ومنطق پردازش می شوند.

۳- نتایج پردازش از CPU به حافظه اصلی منتقل می کند.

۴- نتایج پردازش به دستگاههای خروجی منتقل می شوند.

۵- برنامه ها برای اجرا از حافظه جانبی در RAM بار می شود.

۶- اطلاعات موجود در حافظه RAM در حافظه جانبی نوشته می شود.

حافظه

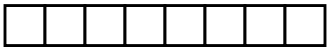
- **قسمتی از کامپیوتر است که داده‌ها و دستورالعملها و نتایج پردازش را به صورت ارقام صفر و یک ذخیره می‌کند.**
- **هر خانه حافظه آدرس مخصوص به خود دارد که منحصر به فرد است.**
- **در هر خانه حافظه می‌توان تنها یک مقدار وارد کرد. با ریختن مقدار جدید، مقدار قبلی پاک می‌شود.**

واحدهای حافظه

BIT
Binary digIT

کوچکترین واحد سخت افزاری حافظه است که می تواند صفر یا یک (دودویی) را ذخیره کند.

Byte
8 bits



واحد اصلی ذخیره سازی اطلاعات در حافظه است که هشت بیت (یک کاراکتر) را می تواند ذخیره کند.کاراتر: ارقام، حروف الفبا، علائم

کدگذاری ASCII

توسط موسسه ملی استاندارد آمریکا طراحی شده است. برای نشان دادن هر کاراکتر از ۸ بیت استفاده شده است. (۲۵۶ =$۲^{۸}$کد)

کدگذاری Unicode

برای زبانهای مختلف و انواع فونت‌هاطراحی شده است. برای نشان دادن هر کاراکتر از ۱۶ بیت استفاده شده است.(۶۵۵۳۶ =$۲^{۱۶}$کد)

تقسیم‌بندی حافظه کامپیوتر

حافظه اصلی به دو دسته حافظه RAM و حافظه ROM تقسیم می‌شود.

۱- Random Access Memory(RAM):
- ✓سرعت دستیابی به این حافظه زیاد است.
- ✓حافظه‌ای ناپایدار است که با قطع جریان برق اطلاعات آن پاک می‌شود.
- ✓می‌توان اطلاعات آن را پاک کرد و اطلاعات جدیدی جایگزین کرد.
- ✓فضای محدودی دارد و برای ذخیره موقتی داده‌ها تا زمان پردازش یا انتقال آنها به کار می‌رود.
- ✓حافظه خواندنی نوشتنی است.
- ✓هرچه بیشتر باشد، سرعت و کارایی سیستم بالاتر می رود.

تقسیم‌بندی حافظه کامپیوتر

حافظه اصلی به دو دسته حافظه RAM و حافظه ROM تقسیم می‌شود.

۲- Read Only Memory (ROM):
- ✓از جنس نیمه هادی است.
- ✓حافظه‌ای پایدار است زیرا با قطع جریان برق اطلاعات آن از بین نمی‌رود.
- ✓کاربر نمی‌تواند اطلاعات آن را پاک کند و یا تغییر دهد.
- ✓اطلاعات مهمی که توسط شرکت سازنده قرار می‌گیرد.
- ✓اطلاعات این حافظه برای تست و راه‌اندازی قسمت‌های سخت افزاری کامپیوتر به کار می‌رود.
- ✓فقط خواندنی است.

حافظه جانبی

- حافظه RAM محدود و موقت است. برای ذخیره دائمی داده‌ها و اطلاعات از حافظه جانبی استفاده می‌شود.
- سرعت دسترسی به داده ها در حافظه جانبی کندتر از حافظه اصلی است، پس داده ها برای اجرا به حافظه اصلی منتقل می شوند.

واحدهای اندازه‌گیری حافظه

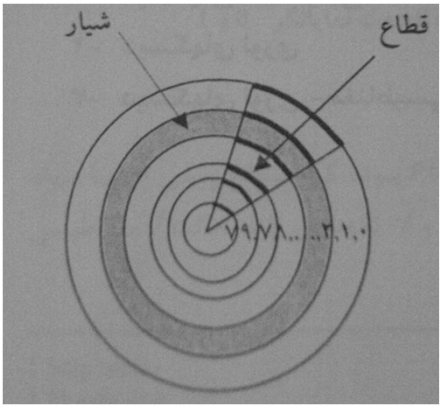
واحد	بایت
کیلوبایت(KB)	$۲^{۱۰}$
مگابایت(MB)	$۲^{۲۰}$
گیگابایت(GB)	$۲^{۳۰}$
ترابایت(TB)	$۲^{۴۰}$
پتابایت(PB)	$۲^{۵۰}$

روشهای دسترسی به اطلاعات

دسترسی ترتیبی
نوار مغناطیسی (جنس پلاستیکی با لایه‌ای از اکسیدآهن)

دستیابی مستقیم(تصادفی)
دیسک‌ها، فلش مموری، حافظه SSD

ساختار ذخیره و بازیابی اطلاعات روی دیسک مغناطیسی



- شیار: سطح دیسک به صورت دوایر متحدالمرکز است که هر دو دایره هم مرکز مجاورهم تشکیل شیار می دهند.
- قطاع: شیارها را به قسمت های مساوی به نام قطاع تقسیم می کنند.
- سیلندر: شیارهای هم شعاع را در دیسک های مختلف تشکیل سیلندر را می دهند.

انواع دیسکها با توجه به محیط ذخیره و بازیابی اطلاعات

- دیسکهای مغناطیسی (جنس، ظرفیت، سرعت)
  - **دیسکهای فلاپی**
  - **دیسکهای سخت**
- دیسکهای نوری (تابش اشعه لیزر وجود یا عدم وجود حفره ها را تشخیص می دهد. اگر شدت نور بالا باشد (۱) و اگر کم باشد (۰) است)
  - **CD-ROM**
  - **WORM**
  - **دیسکهای نوری قابل پاک کردن**
  - **DVD (ظرفیت ۱۰ برابرCD)**
  - **Blu-ray**
- دیسکهای نوری-مغناطیسی

انواع دیسکها با توجه به محیط ذخیره و بازیابی اطلاعات

دیسکهای سخت: به علت سبک بودن و سرعت بالای چرخش این دیسک های آلومینیومی، سرعت دسترسی به آن از فلاپی دیسک ها خیلی بالاتر و در ضمن ظرفیت ذخیره سازی آن خیلی بیشتر از فلاپی دیسک است. ظرفیت یک دیسک سخت معمولی حدود GB ۲۰۰ تا یک ترابایت است.

CD-ROM: ظرفیت ذخیره سازی حدود MB ۷۰۰ و قطر ۴/۷ اینچ است. امکان پاک کردن و یا تغییر اطلاعات ذخیره شده وجود ندارد اما قابلیت چند بارنویسی دارد. ساختار ذخیره سازی به صورت مارپیچی حدود ۶ کیلومتر است.

WORM: اطلاعات یک بار روی آن نوشته می شود و به دفعات خوانده می شود. امکان پاک کردن و تغییر اطلاعات وجود ندارد.

دیسکهای ویدئویی دیجیتالی(DVD): نسل دیگر دیسک های نوری است، ظرفیت ذخیره سازی ۱۰برابر CDهای معمولی است.

Blu-ray: آخرین دیسک نوری است که برای ذخیره داده های ویدیویی با کیفیت بالا طراحی شده است. بسته به انواعی که استفاده می کنید، ظرفیتی بین ۲۵ تا تقریبا ۱۰۰ گیگابایت دارد. برخلاف CD و DVD که از نور لیزر قرمز استفاده می کنند، از نور لیزر آبی استفاده می کند.

دیسکهای نوری-مغناطیسی: تلفیق تکنولوژیهای نوری و مغناطیسی که هم خاصیت قابل پاک شدن و بازنویسی دیسک های مغناطیسی و هم چگالی و ظرفیت بالای دیسک های نوری را دارد.

دستگاههای ورودی

صفحه کلید (Keyboard)

متداولترین نوع دستگاه ورودی است. متداولترین نوع دستگاه ورودی جهت وارد کردن داده ها و برنامه ها است. دارای حداقل ۱۰۱ کلید می‌باشد. کلیدها براساس کاربرد به دسته‌های مختلفی تقسیم می‌شوند.

ماوس (Mouse)

حرکت گوی پلاستیکی داخل ماوس سبب حرکت اشاره گر ماوس و ارسال کد به برنامه می شود و عملیات مربوطه اجرا می شود. جهت ترسیم اشکال در برنامه های گرافیکی نیز به کار می رود. انواع مختلفی مانند مکانیکی، نوری و بی سیم دارد.

اسکنر(Scanner)

متون و تصاویر را جهت اصلاح یا بایگانی وارد حافظه کامپیوتر نمود.

دیجیتایزر(Digitizer)

از یک قلم الکترونیکی و یک صفحه گرافیکی تشکیل شده است. در طراحی به کمک کامپیوتر جهت انتقال نقشه های موجود به حافظه کامپیوتر و تغییر و اصلاح آنها به کار می رود.

قلم نوری (Light Pen)

از یک قلم حساس به نور و یک صفحه نمایش تشکیل شده و می‌توانیم اشکالی را مستقیما بر روی صفحه نمایش ترسیم کنیم.

دسته فرمان (Joystick)

از یک دسته فرمان برای تغییر موقعیت مکان نما بر روی صفحه نمایش به کار می‌رود در بازیهای کامپیوتری استفاده می‌شود.

صفحه نمایش لمسی (Touch Screen)

یک صفحه نمایش حساس که حرکت انگشت دست باعث حرکت اشاره‌گرمی‌شود.

میکروفن

این ابزار ورودی داده‌های صوتی را به حافظه منتقل می‌کند.

Track ball

مشابه ماوس است با این تفاوت که گوی آن در بالا است و با کف دست در تمام جهات می‌چرخد.

دستگاههای خروجی

به دو گروه عمده تقسیم می‌شوند.

Hard copy

با استفاده از چاپگرها تهیه می‌شوند.

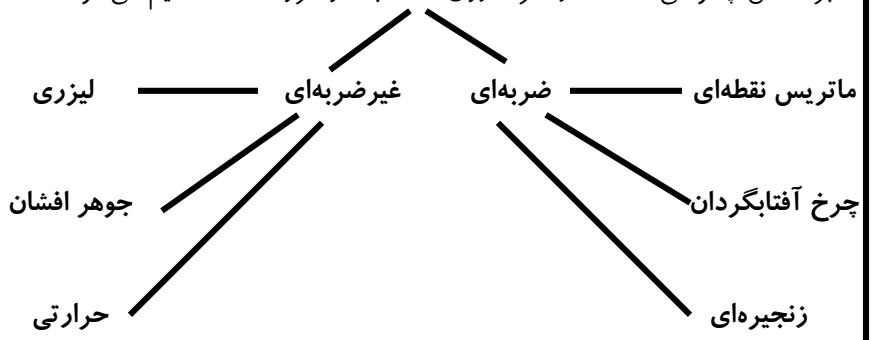
Soft copy

CRT- LCD-LED

بر روی صفحه نمایش ایجاد می‌شود.

چاپگرها(Printers)

بر اساس چگونگی انتقال کاراکترها روی کاغذ به دو گروه عمده تقسیم می‌شوند.



رسام(Plotter)

برای چاپ نقشه‌ها و اشکال گرافیکی روی کاغذ به کار می‌رود.

پروژکتور(Projector)

با استفاده از این وسیله آنچه روی مانیتور قابل مشاهده است به صورت بزرگتری بر روی یک تابلوی سفید منعکس می‌نماید.

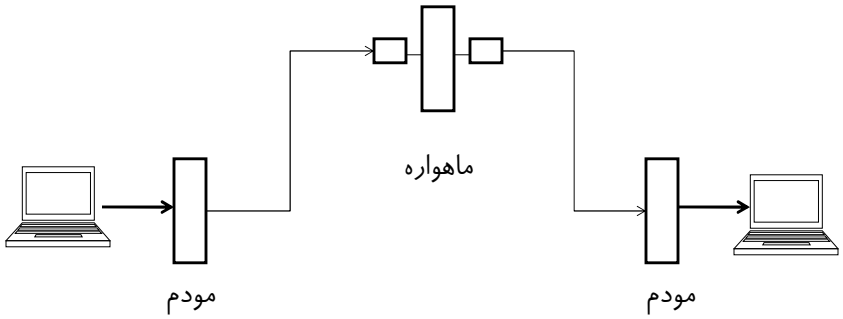
بلندگو(Speaker)

در برنامه‌های مختلف مانند فیلم‌ها، موسیقی‌ها و ... به کار می‌رود.

مودم(Modem)

برای ارسال اطلاعات یک کامپیوتر (سیگنالهای دیجیتال) به شبکه‌های کامپیوتر از جمله اینترنت باید آن را به خطوط و ماهواره های مخابراتی (سیگنالهای آنالوگ) متصل کرد. هم به عنوان ورودی و هم به عنوان خروجی به کار می‌رود.

ارسال و دریافت اطلاعات توسط مودم



مراقبت از سخت‌افزار

گرما

قرار دادن در محلی که در اطراف آن جریان هوا وجود داشته باشد.

گرد و غبار

حداقل سالی یک دفعه گرد و غبار را باید پاک کرد.

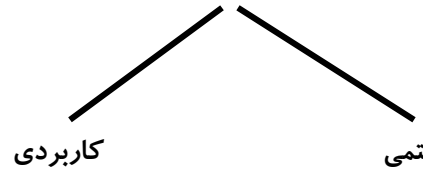
نرم‌افزار کامپیوتر

نرم‌افزار هماهنگ‌کننده و ناظر بر فعالیتهای سخت‌افزار است.

برنامه مجموعه دستورالعملهایی است که به ترتیب خاصی نوشته می‌شود و توسط ریزپردازنده اجرا شده و هدف مشخصی را دنبال می‌کند.

انواع نرم‌افزارها

نرم‌افزارها به دو گروه عمده تقسیم می‌شوند.



نرم‌افزارهای کاربردی

نرم‌افزارهای کاربردی توسط کاربران یا شرکتهای خاصی در زمینه‌های مختلف علمی، مهندسی، تجاری، آموزشی و ... نوشته می‌شوند و به شش گروه تقسیم می‌شوند:

- واژه‌پردازها
- صفحه‌گسترده
- پایگاه‌داده‌ها
- نرم‌افزارهای انیمیشن و مالتی‌مدیا
- نرم‌افزارهای تخصصی
- نرم‌افزارهای طراحی و گرافیکی

واژه‌پردازها

برای نوشتن یک متن، ویرایش آن و پردازش کلمات از واژه‌پردازها استفاده می‌شود.

Microsoft Word

صفحه گسترده

جهت پردازش و کار با آمارهای پرحجم، جداول بزرگ، مسائل مربوط به محاسبات و مقایسه‌های آماری بکار می‌رود و از فرمولها و نمودارهای مختلف استفاده می‌کنند. در زمینه حسابداری و تهیه نمودارها در گزارشهای مدیریت کاربرد دارد.

Microsoft Excel

پایگاه داده‌ها

جهت نمایش، ذخیره و پردازش حجم گسترده اطلاعات به کار می‌رود.

فیلد، رکورد، فایل، پایگاه‌داده‌ها (بانک اطلاعات)

Microsoft Access, SQL Server

نرم‌افزارهای طراحی و گرافیکی

با این نرم‌افزارها می‌توان به سرعت و دقت مدلهای دوبعدی را در بخشهای معماری، مکانیک، عمران و ... انجام داد. برای پوستر، سربرگ، نشریات، تبلیغات و ... به کار می‌رود.

Auto Cad, Corel Draw

نرم‌افزارهای انیمیشن و مالتی‌مدیا

برای ایجاد تصاویر متحرک در فضای دوبعدی و سه‌بعدی به کار می‌روند. برای نمایش فیلم و موسیقی به کار می‌روند.

3D Max, 3D Studio, Flash

CD player, Windows Media Player

نرم‌افزاهای تخصصی

به رشته‌های زیر تقسیم می‌شود:

پزشکی، مهندسی، علوم، تجاری، اداری

نرم‌افزارهای سیستمی

نرم‌افزارهای سیستمی برنامه‌هایی هستند جهت فعال کردن، کنترل کردن و سرویس دادن به کامپیوتر و کاربر به کار می‌روند. این نرم‌افزارها به چهار دسته تقسیم می‌شوند:

- سیستم‌عامل
- مترجم‌ها
- نرم‌افزارهای کمکی
- نرم‌افزارهای ایمن‌سازی کامپیوتر

سیستم عامل

اولین و مهمترین نرم افزاری که روی کامپیوتر نصب می شود. Windows 10 ،xp ،vista ،Linux ،Unix

وظایف سیستم عامل:

- استفاده از کامپیوتر را ساده می کند.
- مدیریت منابع سیستم (ریزپردازنده، حافظه، ورودی-خروجی)
- ایجاد ارتباط بین سخت افزار، سایر نرم افزارها و کاربران

مترجم

ترجمه دستورات به زبان ماشین: کامپایلرها و اسمبلرها

نرم افزارهای کمکی

استفاده از کامپیوتر را ساده تر می کند.

امکان مدیریت بهتر به کاربران را می دهد.

برنامه های NU، NC ، TeraCopy

نرم افزارهای ایمن ساز کامپیوتر

- این نرم افزارها برای جلوگیری از تخریب/تغییر داده ها و برنامه ها توسط ویروس(مثل چرنوبیل) به کارمی روند.
- ویروس یاب ها عمل شناسایی و پاکسازی ویروس ها را انجام می دهند.

مانند: Toolkit، Norton Antivirus، Windows Defender، Avira، Kaspersky ،NOD32

- برای جلوگیری از حمله/تغییر/دستکاری نفوذگرها از دیوارهای آتش استفاده می شود.

شبکه

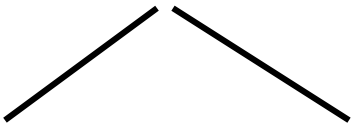
با اتصال چند کامپیوتر با یک ساختار یا طرح مشخص کامپیوترهای توانمندی ایجاد می‌شوند که آنها را شبکه کامپیوتری می نامند.
با استفاده از شبکه های کامپیوتری می‌توان تبادل داده‌ها را انجام داد و اطلاعات یا تجهیزات گران‌قیمت مانند چاپگر را به اشتراک گذاشت.

برای ایجاد شبکه به قسمتهای زیر نیازمندیم:

۱- کامپیوتر فرستنده جهت ارسال اطلاعات
۲- کارت شبکه (فواصل کوتاه) یا کارت مودم(فواصل دور) برای تبادل اطلاعات
۳- کانال ارتباطی
بین دستگاههای مودم از ماهواره‌های مخابراتی و بین کارت‌های شبکه از کابل کواکسیال استفاده می‌شود.
۴- کامپیوتر گیرنده جهت دریافت اطلاعات

انواع شبکه

از نظر فاصله فیزیکی بین کامپیوترها به دو گروه تقسیم می‌شوند.

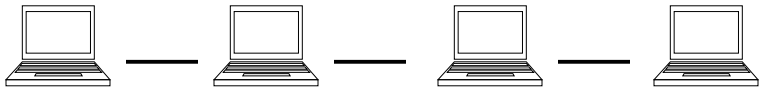


WAN **LAN**

LAN (شبکه محلی)

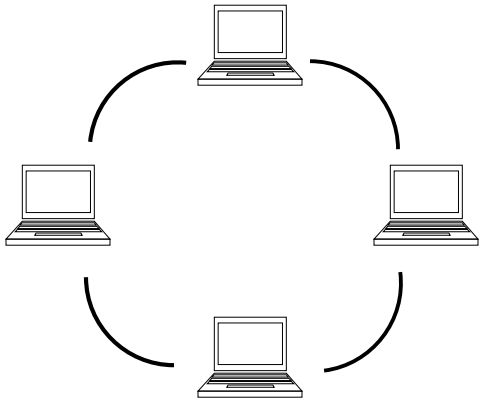
کامپیوترهای واقع در این شبکه در فواصل کوتاهی از همدیگر مانند طبقات مختلف یک اداره قرار دارند.
یکی از راه های تبادل اطلاعات در این شبکه استفاده از کارت شبکه و کابل کواکسیال یا UTP می باشد.
اتصال این کامپیوترها تحت یک ساختار یا طرح مشخصی است که توپولوژی نامیده می شود.

انواع توپولوژیها



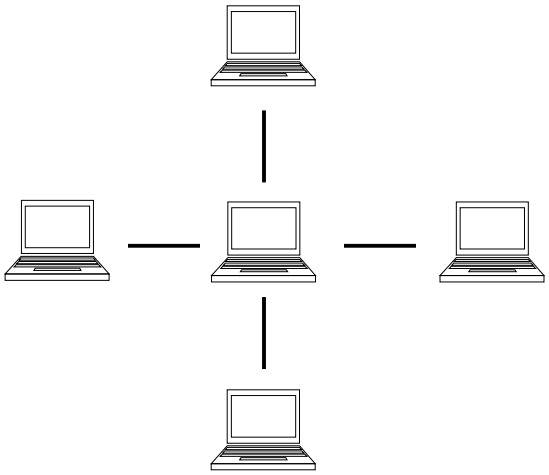
خطی

انواع توپولوژیها



حلقه‌ای

انواع توپولوژیها



ستاره‌ای

WAN (شبکه گسترده)

کامپیوترهای واقع در این شبکه در فواصل زیادی از همدیگر مانند شهرها یا کشورها قرار دارند.

برای تبادل اطلاعات در این شبکه از مودم و ماهوارههای مخابراتی استفاده می‌شود.

دارای انواع مختلف مانند شبکه صدا و سیما، شبکه راه آهن و ... است که بر مبنای پروتکل‌ها و اتصالات و روش‌های آدرس‌دهی کار می‌کنند.

اینترنت

اینترنت شبکه عظیم و پیچیده‌ای از شبکه‌های کامپیوتری مرتبط را در سطح جهان تشکیل می‌دهد.

اینترنت سرویسهای متعددی دارد که بر اساس قوانین و استانداردهای خاصی در دسترس قرار می‌گیرد.

سرویس وب به عنوان جالب‌ترین و محبوب‌ترین سرویس اینترنت روشی برای دستیابی به اطلاعات روی اینترنت است.

اطلاعات روی تعداد بسیار زیادی کامپیوتر در جهان به اشتراک گذاشته شده است.

پروتکلی که وب در شبکه اینترنت برای ارائه اطلاعات استفاده می‌کند http نام دارد. و پروتکل SMTP برای ارسال و دریافت email استفاده می شود.

مرورگرها مانند Microsoft edge و Google chrome، Mozilla Firefox برای در اختیار قرار دادن اطلاعات به کار می‌رود.

موتورهای جستجو مانند Google برای پیداکردن اطلاعات در وب سایتها به کار می‌رود. در وب میلیاردها صفحه اطلاعات قابل دسترسی است.

محاسبات کوانتومی

محاسبات کوانتومی حوزه ای از علوم کامپیوتر است که بر توسعه فناوری های مبتنی بر اصول نظریه کوانتومی متمرکز است. محاسبات کوانتومی از رفتارهای منحصر به فرد فیزیک کوانتومی برای حل مسائلی استفاده می کند که برای محاسبات کلاسیک بسیار پیچیده هستند.

محاسبات کلاسیک از بیت‌های دودویی – ۱ها و ۰ها استفاده می‌کند - محاسبات کوانتومی از ۱ها، ۰ها و هر دو ۱ و ۰ به طور همزمان استفاده می‌کند. کامپیوتر کوانتومی قدرت پردازشی بیشتری دارد زیرا بیت ها می توانند همزمان در چندین حالت باشند.

رایانه‌های کوانتومی از ناحیه‌ای تشکیل شده‌اند که qbitها را در خود جای می‌دهد، روشی که سیگنال‌ها را به qbitها منتقل می‌کند، و رایانه‌ای کلاسیک که برنامه‌ای را اجرا می‌کند و دستورالعمل‌ها را ارسال می‌کند.

یک qbit یا بیت کوانتومی معادل بیت در محاسبات کلاسیک است. همانطور که بیت واحد اصلی اطلاعات در یک کامپیوتر کلاسیک است، qbit نیز واحد پایه اطلاعات در یک کامپیوتر کوانتومی است.

مبنا

پرکاربردترین سیستم عددی دیجیتال، شماره‌گذاری باینری است.

دستگاه اعداد باینری (دودویی) و دستگاه اعداد دسیمال (ده دهی) قوانین مشابهی دارند، با این تفاوت که در دستگاه دسیمال از توان‌های ۱۰ استفاده می‌شود و در دستگاه باینری از توان‌های ۲ استفاده می‌شود.

سیستم‌های دیجیتالی و کامپیوتری برای نمایش یک وضعیت، تنها از دو مقدار «۰» منطقی و «۱» منطقی استفاده می‌کنند. «۰» و «۱» در مبنای ۲ هستند و به آنها ارقام باینری گویند.

در اعداد باینری (۱۰۰۱۱۰) سمت چپ‌ترین بیت، پر ارزش‌ترین بیت است که به آن MSB (Most Significant Bit) گفته می‌شود. همچنین سمت راست‌ترین بیت، کم ارزش‌ترین بیت است که LSB (Least Significant Bit) نامیده می‌شود.

مبنا

۱۰ ⇨ ۲

روش اول

۳۷	۲				
۲	۱۸	۲			
۱۷	۱۸	۹	۲		
۱۶	۰	۸	۴	۲	
۱		۱	۴	۲	۲
			۰	۲	۱
				۰	

$(۳۷)_{۱۰} = (۱۰۰۱۰۱)_{۲}$

مبنا

۱۰ ⇨ ۲ روش دوم

$(۳۷)_{۱۰} = (?)_{۲}$

64	32	16	8	4	2	1
0	1	0	0	1	0	1

۳۷= ۳۲ + ۴ + ۱

مبنا

۲ ⇨ ۱۰ روش اول

۵ ۴ ۳ ۲ ۱ ۰

$(۱\ ۰\ ۰\ ۱\ ۰\ ۱)_{۲} = (۳۷)_{۱۰}$

$1*2^5+0*2^4+0*2^3+1*2^2+0*2^1+1*2^0=32+4+1$

0 0 0

مبنا

۲ ⇨ ۱۰ روش دوم

۳۲ ۱۶ ۸ ۴ ۲ ۱

$(۱\ ۰\ ۰\ ۱\ ۰\ ۱)_۲ = (۳۷)_{۱۰}$

۱+۴+۳۲=۳۷

مکمل ۲

روش اول

2's complement

$(-۳۷)_{۱۰} = (۰۱۱۰۱۱)_۲$

$(۳۷)_{۱۰} = (۱۰۰۱۰۱)_۲$

۰۱۱۰۱۰
\+ ۱
۰۱۱۰۱۱

تمام یک ها را به صفر تغییر داده سپس با یک جمع می کنیم.

مکمل ۲

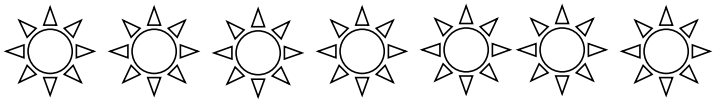
2's complement

روش دوم

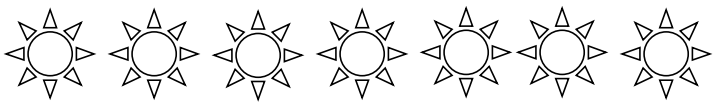
$(-۳۷)_{۱۰} = (۰۱۱۰۱۱)_{۲}$

$(۳۷)_{۱۰} = (۱۰۰۱۰۱)_{۲}$

از سمت راست تا اولین یک را تغییر ندهید، ادامه اعداد را تغییر دهید:
از صفر به یک و از یک به صفر



الگوریتم



الگوریتم

الگوریتم: به مجموعه ای از دستورالعملها که مراحل حل یک مساله را با زبان دقیق و جزئیات کافی بیان کرده، دارای ترتیب مراحل و پایان پذیری مشخصی باشد، الگوریتم می گویند.

زبان دقیق: بدون ابهام.

جزئیات کافی: تمام دستورات قابل تفسیر باشند.

ترتیب مراحل: ترتیب دستورات از لحاظ اجرا مهم است.

پایان پذیری: در زمان معین الگوریتم خاتمه یابد.

ویژگیهای الگوریتم

- متناهی بودن: تعداد مراحل متناهی (محدود) باشند.
- روشن و فاقد ابهام بودن: مراحل اجرا دقیقا روشن باشند.
- مشخص بودن ورودیها: الگوریتم چند ورودی دارد.
- مشخص بودن خروجیها: الگوریتم چند خروجی دارد. چه رابطه ای با ورودیها دارد.
- موثر بودن: تاثیرگذاری دستورات در برنامه.

دستورالعملهای الگوریتم

- شروع
- پایان
- ورودی
- خروجی
- محاسباتی و جایگزینی
- شرطی
- حلقه

الگوریتم ۱

الگوریتمی بنویسید که سه نمره یک دانشجو را دریافت کرده، معدل آن را محاسبه کرده و نتیجه را نمایش دهد.

۱- شروع

۲- a، b و c را از ورودی دریافت کن

۳- avg←(a+b+c)/3

۴- مقدار avg را نمایش بده

۵-پایان

الگوریتم ۲

الگوریتمی بنویسید که شعاع یک کره را دریافت کرده، مساحت و حجم آن را محاسبه کرده و نمایش دهد.

۱- شروع

۲- r را از ورودی دریافت کن

۳- s← 4*3.14*r*r

۴- v← 4/3*3.14*r*r*r

۵- مقدار s و v را نمایش بده

۶- پایان

الگوریتم ۳

الگوریتمی بنویسید که چهار عدد a، b، c، و d را دریافت کرده، حاصل عبارت $S=a^3+b^3+c^3+d^3$ را محاسبه کرده و نمایش دهد.

۱- شروع

۲- a، b، c، وd را از ورودی دریافت کن

۳- $S\leftarrow a^3+b^3+c^3+d^3$

۴- مقدار s را نمایش بده

۵- پایان

جدول ردیابی

برای ردیابی الگوریتم باید از جدول ردیابی استفاده کرد.
در این جدول برای هر متغیر یک ستون در نظر گرفته می شود. ردیابی از ابتدای الگوریتم شروع شده و با در نظر گرفتن مقادیر اولیه متغیرها تا مرحله پایان ادامه داده می شود. مقادیر جدید متغیرها در زمان اجرای الگوریتم در ستون مربوطه نوشته می شود. سپس خروجی نهایی بررسی می شود که درست است یا خیر.

جدول ردیابی الگوریتم زیر را برای سه عدد ۱۸، ۸، ۱۳ ترسیم کرده و خروجی را اعلام کنید.

۱- شروع
۲- a، b و c را از ورودی دریافت کن
۳- avg←(a+b+c)/3
۴- مقدار avg را نمایش بده
۵-پایان

a	b	c	avg
18	8	13	13

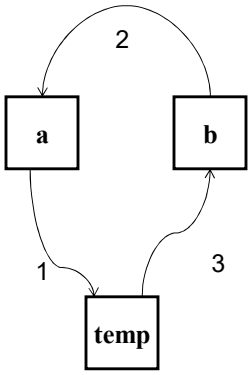
الگوریتم ۴

الگوریتمی بنویسید که حقوق ناخالص یک کارمند را دریافت کرده، ۳٪ بیمه، ۴٪ حق مسکن از آن کم کرده، حقوق خالص را به دست نمایش دهد.

۱- شروع
۲- w را از ورودی دریافت کن
۳- b← 3*w/100
۴- m← 4*w/100
۵- net← w-(b+m)
۶- مقدار net را نمایش بده
۷-پایان

الگوریتم ۵

الگوریتمی بنویسید که دو عدد را دریافت کرده، محتوای آن دو عدد را جابجا کرده و نتیجه را نمایش دهد.



۱- شروع
۲- a، b را از ورودی دریافت کن
۳- temp ← a
۴- a← b
۵- b ← temp
۵- مقدار a و b را نمایش بده
۶- پایان

ساختار کنترل (شرط)

هرگاه در طول الگوریتم نیاز به استفاده از شرط یا شروط داشته باشیم، از اگر استفاده می‌کنیم.

۱- اگر شرط آنگاه دستورات

۲- اگر شرط آنگاه دستورات۱ درغیراینصورت دستورات۲

الگوریتم ۶

الگوریتمی بنویسید که یک عدد دریافت کرده، نشان دهد، مثبت، منفی یا صفر است.

۱- شروع
۲- a را از ورودی دریافت کن
۳- اگر a<0 آنگاه نمایش بده 'a is negative'
۴- اگر a>0 آنگاه نمایش بده 'a is positive'
۵- اگر a=0 آنگاه نمایش بده 'a is zero'
۶- پایان

الگوریتم ۷

الگوریتمی بنویسید که یک عدد دریافت کرده، نشان دهد، زوج است یا فرد.

۱- شروع
۲- a را از ورودی دریافت کن
۳- اگر a mod 2 =0 آنگاه نمایش بده 'Even'
در غیر اینصورت نمایش بده 'Odd'
۴- پایان

الگوریتم ۸

الگوریتمی بنویسید که دو عدد دریافت کرده، عدد بزرگتر را نمایش دهد.

۱- شروع
۲- a و b را از ورودی دریافت کن
۳- max← a
۴- اگر b>max آنگاه max← b
۵- مقدار max را نمایش بده
۶- پایان

الگوریتم ۹

الگوریتمی بنویسید که سه عدد را دریافت کرده، اگر a+c>b باشد مقدار a و در غیراینصورت مقدار b را نمایش دهد.

۱- شروع
۲- a و b و c را از ورودی دریافت کن
۳- اگر a+c>b آنگاه مقدار a را نمایش بده
درغیراینصورت مقدار b را نمایش بده
۴- پایان

الگوریتم ۱۰

الگوریتمی بنویسید که ضرایب یک معادله درجه ۲ را دریافت کرده، ریشه‌های آن را درصورت وجود نمایش دهد.

۱- شروع
۲- a و b و c را از ورودی دریافت کن
۳- d ← b*b-4*a*c
۴- اگر d>=0 آنگاه
x1 ←(-b-sqrt(d))/(2*a)
x2 ←(-b+sqrt(d))/(2*a)
x1 و x2 را نمایش بده
درغیر اینصورت نمایش بده 'no root'
۵- پایان

الگوریتم ۱۱

الگوریتمی بنویسید که سه عدد را دریافت کرده، نشان دهد، آیا این سه عدد می‌توانند اضلاع یک مثلث باشند.

۱- شروع
۲- a و b و c را از ورودی دریافت کن
۳- اگر (a+b)>c و (a+c)>b و (b+c)>a آنگاه
نمایش بده 'Yes'
درغیراینصورت نمایش بده 'No'
۴- پایان

الگوریتم ۱۲

الگوریتمی بنویسید که دو عدد به همراه یک عملگر را از ورودی دریافت کرده، عملیات را روی اعداد انجام داده، نتیجه را نمایش دهد.

۱- شروع
۲- a و b و c را از ورودی دریافت کن
۳- اگر c=‘*’ آنگاه s← a * b
۴- اگر c=‘-’ آنگاه s← a - b
۵- اگر c=‘/’ آنگاه s← a / b
۶- اگر c=‘+’ آنگاه s← a + b
۷- مقدار s را نمایش بده
۴- پایان

الگوریتم ۱۳

الگوریتمی بنویسید که سه عدد دریافت کرده، عدد بزرگتر را نمایش دهد.

۱- شروع
۲- a و b و c را از ورودی دریافت کن
۳- max← a
۴- اگر b>max آنگاه max← b
۵- اگر c>max آنگاه max← c
۶- max را نمایش بده
۷- پایان

حلقه

❖ هنگامیکه یک یا چند دستورالعمل باید چندین بار اجرا شوند، از حلقه ها استفاده می شود.

❖ حلقه به دو صورت وجود دارد:

۱- شمارشی (تعداد مراحل اجرا مشخص است)

۲- غیرشمارشی (بر اساس یک شرط می باشد)

الگوریتم ۱۴

الگوریتمی بنویسید که مجموع اعداد ۱ تا ۱۰۰ را محاسبه کرده و نمایش دهد.

۱- شروع
۲- $I \leftarrow 1$ ، $s \leftarrow 0$
۳- $s \leftarrow s + I$
۴- $I \leftarrow I + 1$
۵- اگر $I<=100$ آنگاه برو به مرحله ۳
۶- مقدار s را نمایش بده
۷- پایان

الگوریتم ۱۵

الگوریتمی بنویسید که عدد صحیحی را از ورودی دریافت کرده و فاکتوریل آن را محاسبه کرده و نمایش دهد.

۱- شروع
۲- n را دریافت کن
۳- I← 1 ، f ← 1
۴- f ← f * I
۵- I ← I + 1
۶- اگر I<=n آنگاه برو به مرحله ۴
۷- مقدار f را نمایش بده
۸- پایان

الگوریتم ۱۶

الگوریتمی بنویسید که دو عدد صحیح مثبت را از ورودی دریافت کرده و حاصلضرب آن دو را بدون استفاده از عملگر ضرب محاسبه کرده و نمایش دهد.

۱- شروع
۲- a و b را دریافت کن
۳- I ← 1 ، s ← 0
۴- s ← s + a
۵- I ← I + 1
۶- اگر I<=b آنگاه برو به مرحله ۴
۷- مقدار s را نمایش بده
۸- پایان

الگوریتم ۱۷

الگوریتمی بنویسید که عدد صحیح مثبت n را از ورودی دریافت کرده مجموع سری زیر را محاسبه کرده و نمایش دهد.

$$S = 1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n}$$

۱- شروع
۲- n را دریافت کن
۳- s ← 0
۴- I ← 1
۵- s← s + 1/I
۶- I ← I + 1
۷- اگر I<=n آنگاه برو به مرحله ۵
۸- مقدار s را نمایش بده
۹- پایان

الگوریتم ۱۸

الگوریتمی بنویسید که مجموع مضارب ۵ کوچکتر از ۱۰۰ را نمایش دهد.

۱- شروع
۲- s ← 0
۳- I ← 0
۴- s← s + I
۵- I ← I + 5
۶- اگر I<=100 آنگاه برو به مرحله ۴
۷- مقدار s را نمایش بده
۸- پایان

الگوریتم ۱۹

الگوریتمی بنویسید که یک عدد را دریافت کرده، مقلوب آن را نمایش دهد.
مقلوب عدد ۱۲۳، عدد ۳۲۱ است.

۱- شروع
۲- n را دریافت کن
۳- m ← 0
۴- b← n mod 10
۵- m ← m * 10 +b
۶- n← n div10
۷- اگر n>0 آنگاه برو به مرحله ۴
۸- مقدار m را نمایش بده
۹- پایان

الگوریتم ۲۰

الگوریتمی بنویسید که عدد صحیح مثبت n را از ورودی دریافت کرده مجموع سری زیر را محاسبه کرده و نمایش دهد.

$$S = \frac{1}{2} + \frac{1}{4} + ... + \frac{1}{n}$$

۱- شروع
۲- n را دریافت کن
۳- s ← 0
۴- I ← 2
۵- s← s + 1/I
۶- I ← I + 2
۷- اگر I<=n آنگاه برو به مرحله ۵
۸- مقدار s را نمایش بده
۹- پایان

الگوریتم ۲۱

الگوریتمی بنویسید که عدد صحیح مثبت n را از ورودی دریافت کرده مجموع سری زیر را محاسبه کرده و نمایش دهد.

$$S = 1 - \frac{1}{3} + \frac{1}{5} - .... + \frac{1}{n}$$

۱- شروع
۲- n را دریافت کن
۳- s ← 0
۴- k ← 1 ، I ← 1
۵- s← s + k/I
۶- I ← I + 2
۷- k=k * (-1)
۸- اگر I<=n آنگاه برو به مرحله 5
۹- مقدار s را نمایش بده
۱۰- پایان

الگوریتم ۲۲

الگوریتمی بنویسید که دو عدد دریافت کرده، خارج قسمت و باقیمانده صحیح تقسیم را بدون استفاده از عمل تقسیم نمایش دهید.

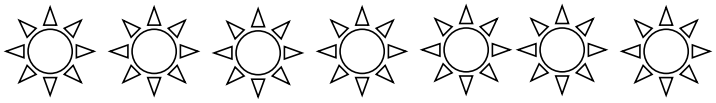
۱- شروع
۲- a و b را دریافت کن
۳- q ← 0
۴- a ← a - b
۵- q ← q + 1
۶- اگر a>=b آنگاه برو به مرحله ۴
۷- a و q را نمایش بده
۸- پایان

تمرین

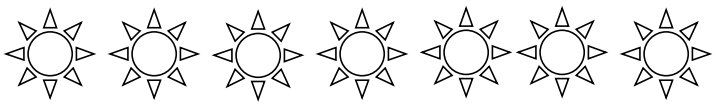
الگوریتمی بنویسید که

- عددی را از ورودی دریافت کرده، قدر مطلق آن را نمایش دهد.
- عددی را از ورودی دریافت کرده، بخشپذیری آن بر ۳ و ۵ را بررسی نماید.
- دو عدد را دریافت کرده و بدون استفاده از متغیر کمکی تعویض نماید.
- یک عدد بین ۱ تا ۷ دریافت کرده معادل روز هفته را نمایش دهد.
- دو عدد را دریافت کرده، اعلام کند آیا این دو عدد متوالی هستند یا خیر؟
- دو عدد را دریافت کرده، اعداد مابین این دو عدد را نمایش دهد. a<b
- مضارب ۷ بین ۱ تا ۱۰۰ را نمایش دهد.
- اعداد فرد کوچکتر از ۲۰ را چاپ کند.
- ۱۰۰ عدد را دریافت کرده، میانگین آن‌ها را محاسبه کرده و نمایش دهد.
- نمره دانشجوئی را از ورودی دریافت کرده، با توجه به مقدار نمره یکی از خروجی های زیر را نمایش دهد.

Grade	output
17 – 20	A
14 - 17	B
12 – 14	C
10 – 12	D
0 – 10	F

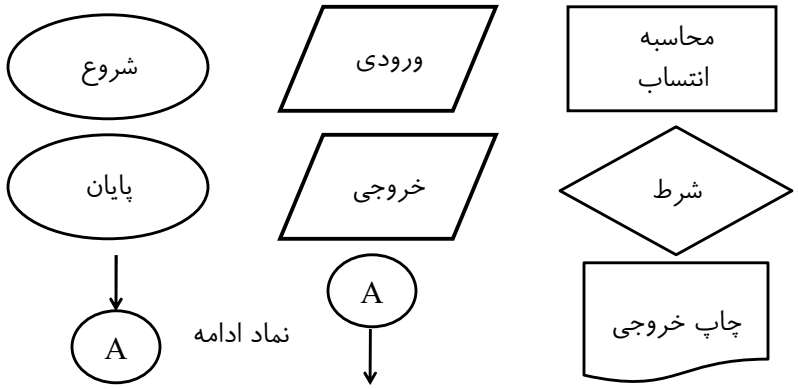


فلوچارت



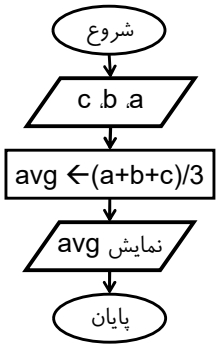
فلوچارت

فلوچارت: فلوچارت، ساده ترین و واضح ترین روش تصویری بیان الگوریتم است.
از نمادهای خاصی(اشکال هندسی) برای نمایش مراحل اجرای الگوریتم استفاده می شود. هر عملی با یک نماد نشان داده می شود.



فلوچارت ۱

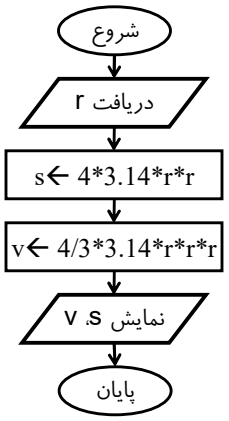
فلوچارتی بنویسید که سه نمره یک دانشجو را دریافت کرده، معدل آن را محاسبه کرده و نتیجه را نمایش دهد.



۱- شروع
۲- a، b و c را از ورودی دریافت کن
۳- avg←(a+b+c)/3
۴- مقدار avg را نمایش بده
۵-پایان

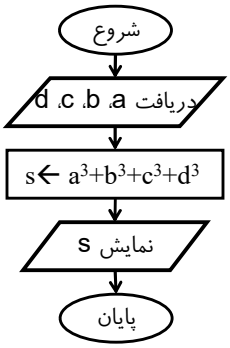
فلوچارت ۲

فلوچارتی بنویسید که شعاع یک کره را دریافت کرده، مساحت و حجم آن را محاسبه کرده و نمایش دهد.



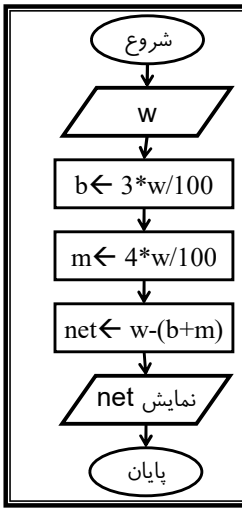
فلوچارت ۳

فلوچارتی بنویسید بنویسید که چهار عدد a، b، c، و d را دریافت کرده، حاصل عبارت $S=a^3+b^3+c^3+d^3$ را محاسبه کرده و نمایش دهد.



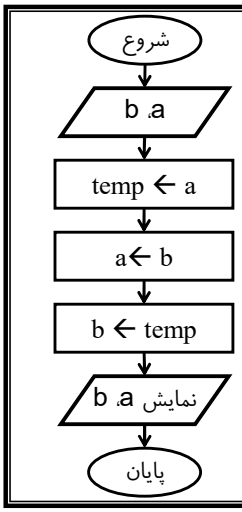
فلوچارت ۴

فلوچارتی بنویسید که حقوق ناخالص یک کارمند را دریافت کرده، ۳٪ بیمه، ۴٪ حق مسکن از آن کم کرده، حقوق خالص را به دست آورید.



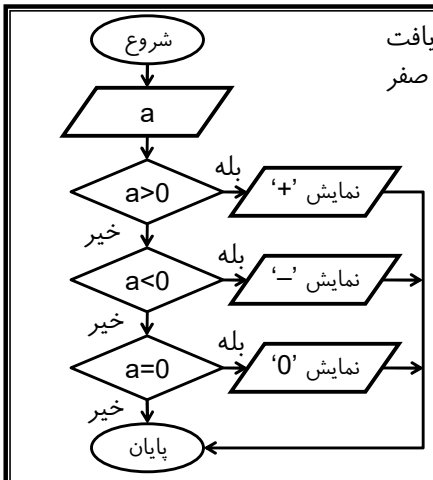
فلوچارت ۵

فلوچارتی بنویسید که دو عدد را دریافت کرده، محتوای آن دو عدد را جابجا کرده و نتیجه را نمایش دهد.



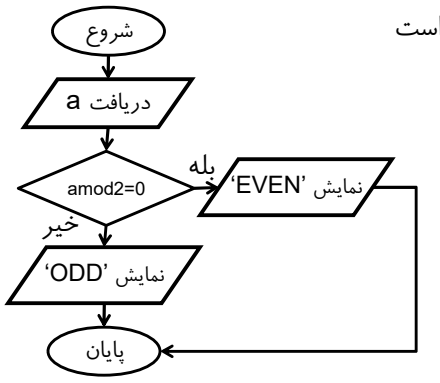
فلوچارت ۶

فلوچارتی بنویسید که یک عدد دریافت کرده، نشان دهد، مثبت، منفی یا صفر است.



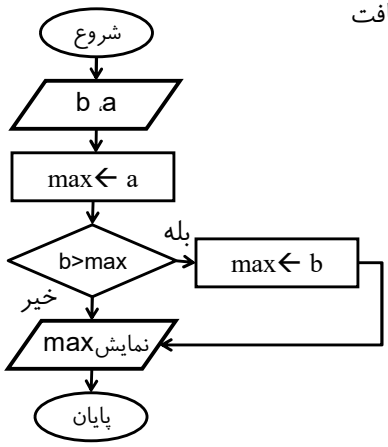
فلوچارت ۷

فلوچارتی بنویسید که یک عدد دریافت کرده، نشان دهد، زوج است یا فرد.



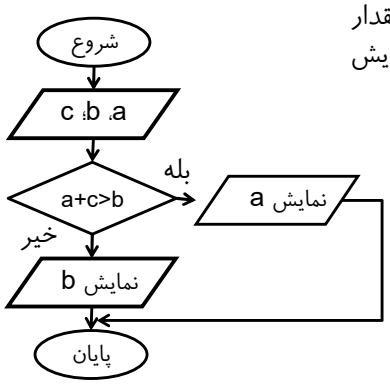
فلوچارت ۸

فلوچارتی بنویسید که دو عدد دریافت کرده، عدد بزرگتر را نمایش دهد.



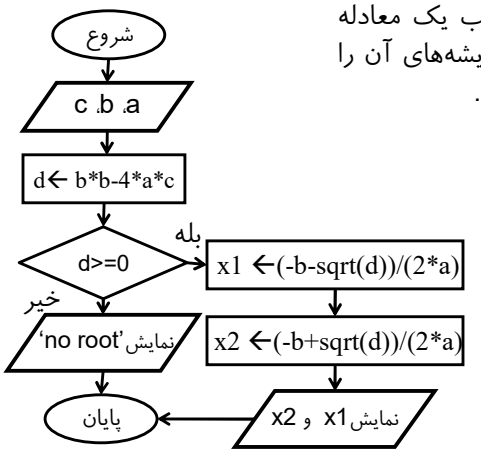
فلوچارت ۹

فلوچارتی بنویسید که سه عدد را دریافت کرده، اگر a+c>b باشد مقدار a و در غیراینصورت مقدار b را نمایش دهد.



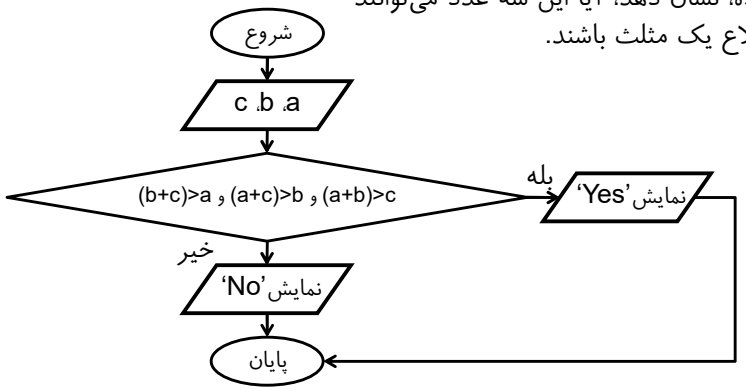
فلوچارت ۱۰

فلوچارتی بنویسید که ضرایب یک معادله درجه ۲ را دریافت کرده، ریشه‌های آن را درصورت وجود نمایش دهد.



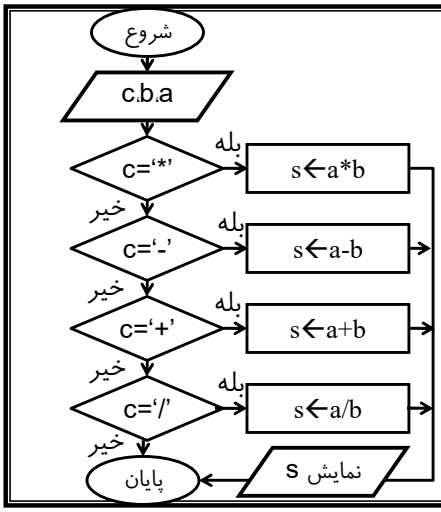
فلوچارت ۱۱

فلوچارتی بنویسید که سه عدد دریافت کرده، نشان دهد، آیا این سه عدد می‌توانند اضلاع یک مثلث باشند.



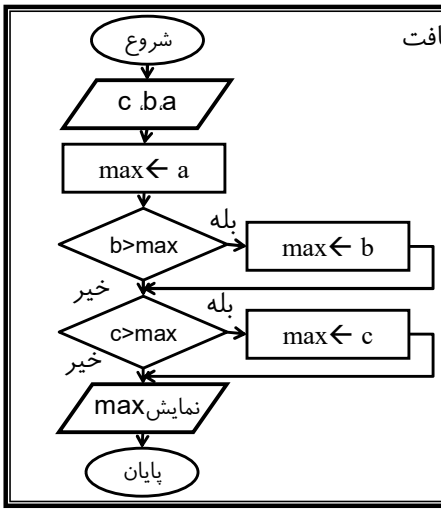
فلوچارت ۱۲

فلوچارتی بنویسید که دو عدد به همراه یک عملگر را از ورودی دریافت کرده، عملیات را روی اعداد انجام داده، نتیجه را نمایش دهد.



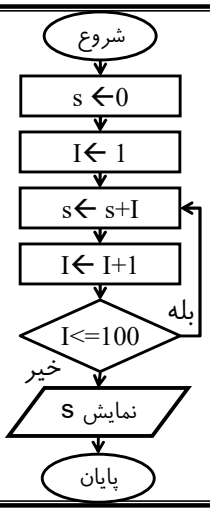
فلوچارت ۱۳

فلوچارتی بنویسید که سه عدد دریافت کرده، عدد بزرگتر را نمایش دهد.



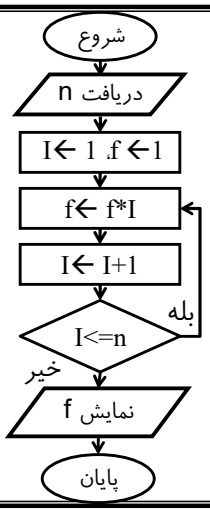
فلوچارت ۱۴

فلوچارتی بنویسید که مجموع اعداد ۱ تا ۱۰۰ را محاسبه کرده و نمایش دهد.



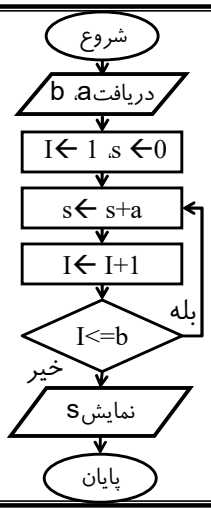
فلوچارت ۱۵

فلوچارتی بنویسید که عدد صحیحی را از ورودی دریافت کرده و فاکتوریل آن را محاسبه کرده و نمایش دهد.



فلوچارت ۱۶

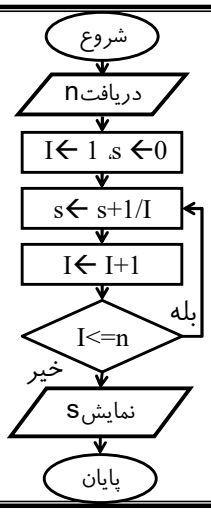
فلوچارتی بنویسید که دو عدد صحیح مثبت را از ورودی دریافت کرده و حاصلضرب آن دو را بدون استفاده از عملگر ضرب محاسبه کرده و نمایش دهد.



فلوچارت ۱۷

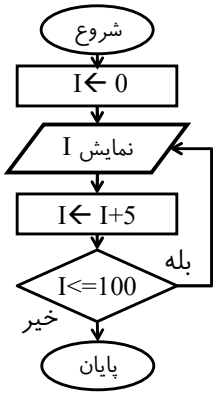
فلوچارتی بنویسید که عدد صحیح مثبت n را از ورودی دریافت کرده مجموع سری زیر را محاسبه کرده و نمایش دهد.

$$S = 1 + \frac{1}{2} + \frac{1}{3} + ... + \frac{1}{n}$$



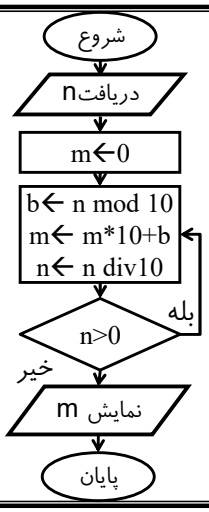
فلوچارت ۱۸

فلوچارتی بنویسید که مضارب ۵ کوچکتر از ۱۰۰ را نمایش دهد.



فلوچارت ۱۹

فلوچارتی بنویسید که یک عدد را دریافت کرده، مقلوب آن را نمایش دهد.

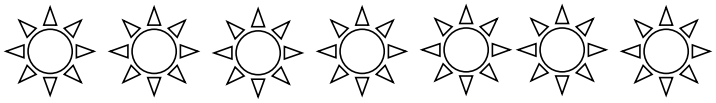


تمرین

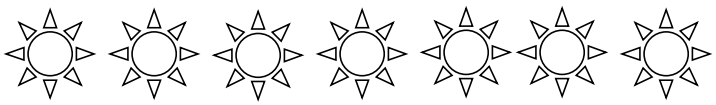
فلوچارتی رسم کنید که

- عددی را از ورودی دریافت کرده، قدر مطلق آن را نمایش دهد.
- عددی را از ورودی دریافت کرده، بخشپذیری آن بر ۳ و ۵ را بررسی نماید.
- دو عدد را دریافت کرده و بدون استفاده از متغیر کمکی تعویض نماید.
- یک عدد بین ۱ تا ۷ دریافت کرده معادل روز هفته را نمایش دهد.
- دو عدد را دریافت کرده، اعلام کند آیا این دو عدد متوالی هستند یا خیر؟
- دو عدد را دریافت کرده، اعداد مابین این دو عدد را نمایش دهد. a<b
- مضارب ۷ بین ۱ تا ۱۰۰ را نمایش دهد.
- اعداد فرد کوچکتر از ۲۰ را چاپ کند.
- ۱۰۰ عدد را دریافت کرده، میانگین آن‌ها را محاسبه کرده و نمایش دهد.
- نمره دانشجوئی را از ورودی دریافت کرده، با توجه به مقدار نمره یکی از خروجی های زیر را نمایش دهد.

Grade	output
17 – 20	A
14 - 17	B
12 – 14	C
10 – 12	D
0 – 10	F



برنامه نویسی C++



یک برنامه ساده

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main()
{
cout<<"Hello C++";
return 0;
}
```

یک برنامه ساده

```
/*
A simple program.
This program contains all of the
basic elements
*/

#include "stdafx.h"
#include <iostream>
using namespace std;

int main()
//main is where program execution begins.
{
cout<<"Hello C++";
return 0;
}
```

یک برنامه ساده

```
/*
A simple program.
This program contains all of the
basic elements
*/

#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
//main is where program execution begins.
{
cout<<"Hello C++";
getch();
return 0;
}
```

توضیح Comment

۱–Multiline comment : توضیحات چند خطی نامیده می‌شوند که با یک /* شروع و با */ به پایان می‌رسد.

۲– Single-line comment: توضیح یک خطی نامیده می‌شود که با // شروع شده و در همان خط پایان می‌یابد.

محتویات توضیح توسط کامپایلر نادیده گرفته می‌شود.

main( )

۱- همه برنامه‌ها در C++ ترکیبی از یک یا چند تابع می‌باشند.
۲- تنها تابعی که تمام برنامه‌ها دارند تابع main می‌باشد.
۳- main جایی است که اجرای برنامه از آن‌جا شروع می‌شود.

header files

۱- زبان C++ چندین فایل تعریف می‌کند که فایل‌های سرآمد یا header files می‌گویند.

۲- هر فایل سرآمد شامل اطلاعاتی است که برای برنامه ضروری می‌باشند.

۳- iostream برای پشتیبانی از سیستم ورودی و خروجی می‌باشد. برای استفاده از cout از این فایل سرآمد استفاده می‌شود. محتویات فایل iostream.h به برنامه اضافه می شود.

۴- conio.h برای استفاده از تابع getch مورد نیاز می‌باشد.

cout

یک شناسه از پیش تعریف شده است که مخفف console output می‌باشد و برای نمایش روی صفحه نمایش به کار می‌رود.

- ✓ انتهای تمام دستورات در C++ به سمی‌کولون (;) ختم می‌شود.

stdafx.h

یکprecompiled header file است که شامل فایل های پیش نیاز پروژه های C++ در محیط visual C++ است.

getch()

این تابع باعث می‌شود تا زمانیکه از ورودی کاراکتری دریافت نشده است اجرای برنامه متوقف گردد.

return 0

با این دستور تابع main پایان می‌یابد و مقدار صفر به پروسه فراخواننده main که معمولا سیستم‌عامل است بازگردانده می‌شود.

مقدار صفر نشان دهنده آن است که برنامه به طور عادی پایان یافته است. مقادیر دیگر نشان‌دنده آن است که برنامه به علت خطایی پایان یافته است. return یکی از کلیدواژه‌های C++ می‌باشد که برای بازگرداندن مقداری از یک تابع به کار می‌رود.

using namespace std

حوزه نام std بصورت عمومی تعریف می شود. در صورتی که این عبارت استفاده نشود باید قبل از تمام دستورات ورودی و خروجی std:: اضافه گردد.

دومین برنامه ساده

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
int num;
num=99;
cout<<"The value is:";
cout<<num;
getch();
return 0;
}
```

تعریف متغیر

- متغیر محل نامگذاری شده‌ای از حافظه است که می‌توان مقداری در آن قرار داد.
- محتوای یک متغیر قابل تغییر است و ثابت نیست.
- برای تعریف متغیر ابتدا نوع متغیر و سپس نام دلخواهی برای آن تعیین می‌کنیم.

```
type   variable;
```

- با علامت = مقدار ۹۹ را داخل متغیر num قرار داده می‌شود.
- می‌توان چند متغیر را همزمان تعریف نمود.

```
int a,b;
```

نوع داده‌های اصلی

کلمه کلیدی	دامنه	شرح
bool	مقدار درست یا نادرست	true و false
char	۱۲۷ تا ۱۲۸−	کاراکترهای ۸ بیتی 'A'
int	۳۲۷۶۷ تا ۳۲۷۶۸−	عدد صحیح
float	۴/۳E+۳۸ تا ۴/۳E−۳۸	مقدار اعشاری
double	۷/۱E+۳۰۸ تا ۷/۱E−۳۰۸	مقدار اعشاری با دقت مضاعف
void	اعمال نمی‌شود	بیانگر یک عبارت بدون مقدار
wchar_t	۰ تا ۶۵۵۳۵	کاراکترهای عریض (زبان چینی)

سومین برنامه ساده

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
cout<<"First line.\n";
cout<<"Second line.\n";
cout<<"Third line.\n";
getch();
return 0;
}
```

کاراکتر خط جدید
(newline)

خروجی برنامه

First line.
Second line.
Third line.

سومین برنامه ساده

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
cout<<"One\nTwo\nThree";
getch();
return 0;
}
```

خروجی برنامه

One
Two
Three

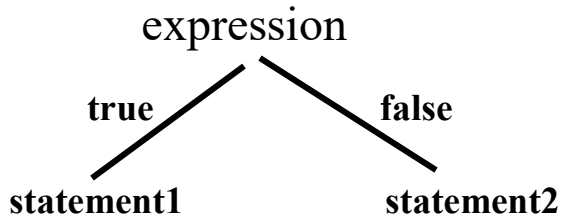
دریافت اطلاعات از کاربر

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
int a,b;
cin>>a>>b;
cout<<"The value a is:"<<a<<"\n";
cout<<"The value b is:"<<b<<"\n";
getch();
return 0;
}
```

به جای "\n" می توان از endl استفاده کرد.

دستور if

```
if (expression) statement1;
      else statement2;
```



استفاده از if

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
float num1,num2;
int choice;
cout<<"Enter values:";
cin>> num1>>num2>>choice;
if (choice==1) cout<<num1+num2;
if (choice==2) cout<<num1-num2;
getch();
return 0;
}
```

استفاده از else

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
float num1,num2;
int choice;
cout<<"Enter values:";
cin>> num1>>num2>>choice;
if (choice==1) cout<<num1+num2;
else cout<<num1-num2;
getch();
return 0;
}
```

استفاده از بلوک کد

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
float num1,num2;
int choice;
cout<<"Enter values:";
cin>>choice;
if (choice==1) {
cin>> num1>>num2;
cout<<num1+num2;
}
getch();
return 0;
}
```

بزرگترین و کوچکترین عدد

```
#include "stdafx.h"
#include <iostream>

using namespace std;

int main()
{
    cout<<"Minimum int, Maximum int\n";
    cout<<INT_MIN<<", "<<INT_MAX <<"\n";
    cout<<"unsigned int \n"<<UINT_MAX <<"\n";
    system("pause");
    return 0;
}
```

Output

```
Minimum int, Maximum int
-2147483648, 2147483647
unsigned int
4294967295
Press any key to continue . . .
```

System("pause") به جای getch()
استفاده شده است.

زوج / فرد

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int num;
    cout<<"Enter a number:";
    cin>>num;
    if(num%2==0)
        cout<<"The number is even.";
    else
        cout<<"The number is odd.";
    system("pause");
    return 0;
}
```

Output

```
Enter a number:11
The number is odd.Press any key to continue . . .
```

دو عدد و یک عملگر

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main(){
    float a,b;
    char op;
    cout<<"Enter numbers:\n";
    cin>> a >> b;
    cout<<"Enter operation:\n";
    cin>>op;
    if(op=='+')
        cout<<a + b;
    if(op=='-')
        cout<<a - b;
    if(op=='*')
        cout<<a * b;
    if(op=='/')
        cout<<a / b;
    system("pause");
    return 0;}
```

Output

```
Enter numbers:
6
4
Enter operation:
/
1.5Press any key to continue . . .
```

استفاده از عملگر ?

```
if(condition) var=exp1;
else var=exp2;
```

```
var=condition ? exp1:exp2;
```

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main(){
    int i;
    cout<<"Enter a number.\n";
    cin>>i;
    i=i>0 ? 1: -1;
    cout<<i<<"\n";
    system("pause");
    return 0;
}
```

عملگر sizeof

ازعملگرهای یکتائی می باشد و مشخص کننده تعداد بایت هائی است که یک نوع داده اشغال می‌کند.

```
int   x;
cout << sizeof x ;
cout << sizeof(float) ;
```

عملگرهای منطقی

عملگر	عمل
&&	AND
||	OR
!	NOT

دستور switch

شکل کلی دستور به صورت زیر است:

```
switch(expression){
      case(val1):
    ┌ instructions
    └ break;
      case(val2):
    ┌ instructions
    └ break;
      ⋮
      default:
    ┌ instructions
}   └
```

به کار بردن break

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

using namespace std;
int main()
{
int num;
cin>> num;
switch (num) {
    case 1:
    cout<< "First Case"; break;
    case 2:
    cout<< "Second Case";break;
    case 3:
    cout<< "Third Case"; break;
    case 4:
    cout<< "Forth Case"; break;
    case 5:
    cout<< "Fifth Case"; break;
    default:
    cout<<"Nothing";
    }
getch();
return 0;
}
```

تمرین

برنامه ای بنویسید که

- عددی را از ورودی دریافت کرده، قدر مطلق آن را نمایش دهد.
- عددی را از ورودی دریافت کرده، بخشپذیری آن بر ۳ و ۵ را بررسی نماید.
- دو عدد را دریافت کرده و بدون استفاده از متغیر کمکی تعویض نماید.
- یک عدد بین ۱ تا ۷ دریافت کرده معادل روز هفته را نمایش دهد.
- دو عدد را دریافت کرده، اعلام کند آیا این دو عدد متوالی هستند یا خیر؟
- شعاع دایره را دریافت کرده، محیط و مساحت دایره را محاسبه کرده و نمایش دهد.
- ضرایب یک معادله درجه۲ را دریافت کرده، ریشه‌های آن را درصورت وجود محاسبه کرده و نمایش دهد.
- یک عدد بین ۱ تا ۱۲ دریافت کرده معادل ماه مربوطه را نمایش دهد.
- نمره دانشجوئی را از ورودی دریافت کرده، با توجه به مقدار نمره یکی از خروجی های زیر را نمایش دهد.

Grade	output
17 – 20	A
14 - 17	B
12 – 14	C
10 – 12	D
0 – 10	F

استفاده از عملگرهای افزایشی و کاهشی

i=i+1;	=	i++;
i=i-1;	=	i--;

استفاده از Assignment operators

*= += -= /=

x=x + y <= x+=y

استفاده از عملگرهای افزایشی و کاهشی

می‌توان عملگرها را پیش از متغیرها هم به کار برد. اما معنای متفاوتی دارند.

j=i++; ⇨	ابتدا مقدار i به j نسبت داده شده و سپس i یک واحد اضافه می‌شود.
j=++i; ⇨	ابتدا مقدار i یک واحد اضافه می‌شود و سپس مقدار i به j نسبت داده می‌شود.

مثال

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int i, j, k, l;
i=15;
j=20;
k=i++;
l=++j;
cout<<i<<"\n"<<j<<"\n"<<k<<"\n"<< l<<"\n";
system("pause");
return 0;
}
```

```
Output
16
21
15
21
Press any key to continue . . .
```

مثال

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int i, j, k, l;
i=15;
j=20;
k=i--;
l=--j;
cout<<i<<"\n"<<j<<"\n"<<k<<"\n"<<l<<"\n";
system("pause");
return 0;
}
```

Output

```
14
19
15
19
Press any key to continue . . .
```

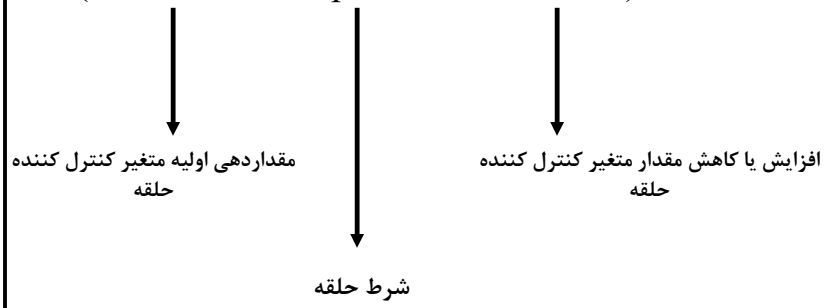
تقدم عملگرها

1	()
2	! ~ ++ -- sizeof
3	* / %
4	+ -
5	<< >>
6	< <= > >=
7	== !=
8	&
9	^
10	|
11	&&
12	||
13	?
14	= += -= *= /= %=
15	,

حلقه for

حالت کلی این دستور به صورت زیر می‌باشد.

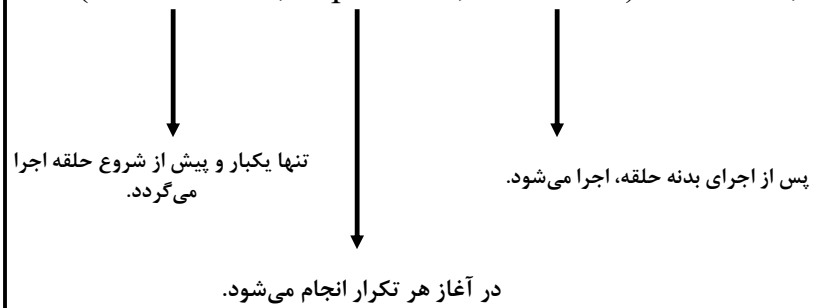
```
for (initialization; expression; increment) statement;
```



حلقه for

حالت کلی این دستور به صورت زیر می‌باشد.

```
for (initialization; expression; increment) statement;
```



برنامه ای بنویسید که اعداد بین ۱ تا ۱۰ را نمایش دهد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num;
for (num=1; num<11; num= num+1)
    cout<< num<<" ";
system("pause");
return 0;
}
```

Output

```
1 2 3 4 5 6 7 8 9 10 Press any key to continue . . .
```

حلقه for

کاهش و افزایش متغیر کنترل کننده می‌تواند بیش از یک واحد باشد.

```
for (num=10; num>0; num= num-4)
```

```
for (num=0; num<11; num= num+4)
```

عدد صحیح و مثبت n را دریافت کرده فاکتوریل آنرا محاسبه ونمایش دهد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int  n, i ;
long fact = 1 ;
cout << "Enter a positive integer number\n";
cin >> n;
for( i=1; i<=n; ++i)
    fact *= i;
cout << fact<<"\n" ;
system("pause");
return 0;
}
```

Output

```
Enter a positive integer number
5
120
Press any key to continue . . .
```

دوعدد را دریافت کرده و عدد اول را به توان عدد دوم برساند و نتیجه را نمایش دهد. x^y

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int x,y,p=1;
cin>>x>>y;
for(int n=1;n<=y;n++)
    {
        p=p*x;
    }
cout<<"Power is:"<<p<<"\n";
system("pause");
return 0;
}
```

Output

```
2
3
Power is:8
Press any key to continue . . .
```

به کار بردن for

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num;
for (num=11; num<11; num= num+1)
    cout<< num<<" ";
system("pause");
return 0;
}
```

Output

?

به کار بردن بلوک کد در for

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num, sum, prod;
sum=0;
prod=1;
for (num=1; num<11; num= num+1){
    sum= sum + num;
    prod= prod * num;
}
cout<< "product:"<< prod<< "\tsum:"<< sum <<"\n";
system("pause");
return 0;
}
```

Output

```
product:3628800 sum:55
Press any key to continue . . .
```

کاهش به جای افزایش در for

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num, sum, prod;
sum=0;
prod=1;
for (num=10; num>0; num= num-1){
    sum= sum + num;
    prod= prod * num;
}
cout<< "product:"<< prod<< "\tsum:"<< sum <<"\n";
system("pause");
return 0;
}
```

Output

```
product:3628800 sum:55
Press any key to continue . . .
```

حلقه های تو در تو

```
#include "stdafx.h"
#include <iostream>
 using namespace std;

int main()
{
    system("color 5");
    for(int i=1; i<=10; i++)
        for(int j=1; j<=10; j++)
            cout<<i*j<<"\t";
        cout<<"\n";

    system("pause");
    return 0;
}
```

رنگی کردن خروجی

Output

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

```
Press any key to continue . . .
```

مقادیر منطقی در زبان C

==0	=>	false
!=0	=>	true

استفاده از مقادیر منطقی

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
bool p,q;
p=true;
q=true;
cout<<"p=true,"<<"q=true"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
p=true;
q=false;
cout<<"p=true,"<<"q=false"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
p=false;
q=true;
cout<<"p=false,"<<"q=true"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
p=false;
q=false;
cout<<"p=false,"<<"q=false"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
system("pause");
return 0;
}
```

Output

```
p=true,q=true   (p&&q):1 (p||q):1
p=true,q=false  (p&&q):0 (p||q):1
p=false,q=true  (p&&q):0 (p||q):1
p=false,q=false (p&&q):0 (p||q):0
Press any key to continue . . .
```

عدد دریافتی اول است؟

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int num, i, is_prime;
    cin>>num;
    is_prime=1;
    for (i=2; i<=num/2; i++)
        if(!(num%i)) is_prime=0;
    if(is_prime) cout<<"The number is prime.";
    else cout<<"The number is not prime.";
    system("pause");
    return 0;
}
```

فرض می‌شود عدد اول است.

حلقه while

حالت کلی این دستور به صورت زیر می‌باشد.

while (expression) statement;

↓

شرط حلقه

تا زمانیکه شرط برقرار باشد دستورات اجرا می‌شود.
شرط حلقه در ابتدای حلقه کنترل می‌شود.

اعداد فرد بین num و صفر

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int num;
    cin>> num;
    while(num) {
    if (num%2) cout<<num<< " ";
    num--;
    }
    system("pause");
    return 0;
}
```

Output

```
5
5 3 1 Press any key to continue . . .
```

حلقه do

حالت کلی این دستور به صورت زیر می‌باشد.

```
do {
        statements
 } while (expression);
```

شرط حلقه

تا زمانیکه شرط برقرار باشد دستورات اجرا می‌شود.

شرط حلقه در انتهای حلقه کنترل می‌شود.

حلقه حداقل یکبار اجرا خواهد شد.

استفاده از break برای خروج از حلقه

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i;
    for(i=1; i<100;i++){
        cout<< i<< " ";
    if(i==10) break;
    }
    system("pause");
    return 0;
}
```

Output

```
1 2 3 4 5 6 7 8 9 10 Press any key to continue . . .
```

استفاده از continue برای اجرای دور بعدی حلقه

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int x;
    for(x=1; x<100;x++){
         continue;
        cout<< x;
    }
    system("pause");
    return 0;
}
```

Output

```
Press any key to continue . . .
```

حلقه goto

حالت کلی این دستور به صورت زیر می‌باشد.

```
goto label;

label:
```

برچسب

با دیدن goto به خطی می‌رود که برچسب یکسانی دارد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i;
    i=1;
    again:
        cout<<i<<" ";
        i++;
    if(i<10) goto again;
    system("pause");
    return 0;
}
```

Output

```
1 2 3 4 5 6 7 8 9 Press any key to continue . . .
```

نگاهی دقیق‌تر به نوع داده‌ها

با استفاده از توصیف‌کننده‌های نوع (type modifiers) می‌توان نوع داده‌های char, int, float, double را تعدیل کرد. توصیف‌کننده‌های نوع عبارتند از :

signed, unsigned, long, short

long int i; **توصیف کننده نوع پیش از نوع داده آورده می‌شود.**

signed (علامت‌دار)

signed را می‌توان در مورد انواع int و char به‌کار برد.

استفاده از signed برای اعداد صحیح کار زایدی است زیرا به طور پیش فرض اعداد صحیح علامت‌دار هستند.

بسته به نوع پیاده‌سازی یک char می‌تواند علامت‌دار یا بدون علامت باشد. در صورتی که بدون علامت باشد اعداد مثبت بین ۰ تا ۲۵۵ را نگاه می‌دارد. درصورتی که به عنوان signed باشد می‌تواند اعدادی در دامنه ۱۲۸- تا ۱۲۷ را نگاه می‌دارد. signed char x;

long

long را می‌توان در مورد نوع داده‌های int و double به‌کار برد.

long بر روی int طول آن را بر حسب بیت دوبرابر می‌کند. اگر در محیطی طول int ۱۶ بیت باشد با استفاده از long طول آن ۳۲ بیت می‌شود.

long بر روی double طول double را نیز بر حسب بیت دوبرابر می‌کند.

```
long int g;
```

short

short را می‌توان تنها در مورد نوع داده int به‌کار برد.

short بر روی int طول آن را بر حسب بیت نصف می‌کند. اگر در محیطی طول int ۳۲ بیت باشد با استفاده از short طول آن ۱۶ بیت می‌شود.

unsigned (بدون علامت)

unsigned را می‌توان در مورد نوع داده‌های int و char به‌کار برد.

این توصیف‌کننده را همراه توصیف‌کننده‌های short و long نیز می‌توان مورد استفاده قرار داد.

signed & unsigned

۱- تفاوت اعداد صحیح علامت‌دار در نحوه تفسیر بیت با مرحله بالاتر می‌باشد.

۲- اگر عدد صحیح بدون علامت باشد از تمامی بیت‌ها برای نگه‌داری مقادیر استفاده می‌شود.

۳- اگر عدد صحیح علامت‌دار باشد در آن صورت کامپایلر کدی تولید می‌کند که فرض می‌نماید از بیت با مرتبه بالاتر به عنوان علامت استفاده می‌شود. اگر ۰ باشد عدد مثبت و اگر ۱ باشد عدد منفی است.

۴- عموما اعداد منفی با روش two’s complement (مکمل دو) بیان می‌شوند.

ترکیبات ممکن از توصیف‌کننده‌ها

نوع	بیت	دامنه
char	۸	۱۲۸– تا ۱۲۷
unsigned char	۸	۰ تا ۲۵۵
signed char	۸	۱۲۸– تا ۱۲۷
int	۱۶	۳۲۷۶۸– تا ۳۲۷۶۷
unsigned int	۱۶	۰ تا ۶۵۵۳۵
signed int	۱۶	۳۲۷۶۸– تا ۳۲۷۶۷
short int	۱۶	مثل int
unsigned short	۱۶	مثل unsigned int
signed short int	۱۶	مثل short int
unsigned long	۳۲	۰ تا ۴۲۹۴۹۶۷۲۹۵
long	۳۲	۲۱۴۷۴۸۳۶۴۸ – تا ۲۱۴۷۴۸۳۶۴۷

ترکیبات ممکن از توصیف‌کننده‌ها

نوع	عرض
long int	۳۲
unsigned long	۳۲
signed long int	۳۲
float	۳۲
double	۶۴
long double	۸۰

نکته

ممکن است در محیط کاری شما **long** و **short** اصلا تاثیری نداشته باشد. بستگی به کامپایلر دارد.

const

اگر پیش از نوع داده یک متغیر توصیف کننده const آورده شود محتوای آن از سوی برنامه تغییر نمی‌کند. (مگر سخت‌افزاری)

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main(){
    const int ten=10;
    cout<<ten*5<<"\n";
    system("pause");
    return 0;
}
```

typedef

typedef OldDataType NewName;

با استفاده از typedef برای یک نوع داده موجود نام تازه‌ای ایجاد کرد.

```
typedef int length;
typedef length depth;
depth d;
typedef short int myint;
```

define

از define جهت تعریف یک شناسه و یک دنباله کاراکتری استفاده می‌شود طوری که در source برنامه هرجا با آن شناسه برخورد شود آن دنباله کاراکتری جانشین آن گردد. این دستور از دستورات preprocessor است و نیازی به ; ندارد.

#define macro-name character-sequence

```
#define UP 1
#define GETFILE "Enter File Name"
#define myfor for(int k=10;k<15;k++)
```

Preprocessor برنامه سیستم است که قبل از ترجمه برنامه توسط کامپایلر تغییراتی در برنامه ایجاد می کند.

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main(){
    #define myfor for(int k=10;k<15;k++)
    myfor
        cout<<k<<"\t";
    system("pause");
    return 0;
}
```

تمرین

برنامه ای بنویسید که
- دو عدد را دریافت کرده، اعداد مابین این دو عدد را نمایش دهد. a<b
- مضارب ۷ بین ۱ تا ۱۰۰ را نمایش دهد.
- اعداد فرد کوچکتر از ۲۰ را چاپ کند.
- ۱۰۰ عدد را دریافت کرده، میانگین آن‌ها را محاسبه کرده و نمایش دهد.
- n جمله اول سری فیبوناچی را نمایش دهد.
- ۵۰ عدد را دریافت کرده، بزرگترین و کوچکترین عدد را نمایش دهد.
- دو عدد m و n را دریافت کرده و یک مستطیل رنگی با یک کاراکتر نمایش دهید.
- یک عدد را دریافت کرده مقلوب عدد را نمایش دهد.
- تعداد و مجموع ارقام یک عدد دریافتی را نمایش دهد.
- بزرگترین و کوچکترین رقم یک عدد دریافتی را نمایش دهد.
- تعدادی عدد را دریافت کرده، مجموع اعداد, میانگین اعداد، بزرگترین عدد و کوچکترین عدد را محاسبه کرده و نمایش دهد. عدد آخر صفر باشد.

آرایه‌ها و رشته‌ها

یک آرایه مجموعه‌ای از متغیرهای به‌هم مرتبط است که به وسیله یک نام مشترک مشخص می‌شود.

آرایه یک بعدی فهرستی از متغیرهایی از یک نوع واحد است که با استفاده از یک نام مشترک به همه آنها مراجعه می‌شود.

به هر متغیر یک آرایه یک عنصر آرایه گفته می‌شود.

حالت کلی آرایه‌ها به صورت زیر می‌باشد:

```
type var_name[size];
```

برای دسترسی به هر عنصر آرایه باید از شماره آن عنصر به عنوان index استفاده نمود.

آرایه‌ها و رشته‌ها

اندیس تمام آرایه‌ها از صفر شروع می‌شود و تا size-1 ادامه دارد.

```
int d[20];
```

مقدار عنصر صفر آرایه را برابر با ۱۰۰ قرار می‌دهد.

```
d[0]=100;
```

برای مقداردهی آرایه‌ها نیاز به حلقه‌های for داریم. مقادیر عناصر را باید تک به تک قرار داد.

مشکلی که وجود دارد این است که بر روی ایندکس آرایه‌ها عمل بررسی دامنه (bound checking) صورت نمی‌پذیرد.

به عنوان برنامه‌نویس خود باید مراقب این مساله باشید.

مثال:

```
int t[20];
t[25]=14;
```

برنامه‌ای بنویسید که ۳۰ عدد صحیح مثبت را دریافت کرده و max آنها را بیابید، نمایش دهید.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i, max=0;
    int n[30];
    for(i=0; i<30;i++)
    {
        cin>>n[i];
        if(n[i]>max) max=n[i];
    }

    cout<<"Maximum is:"<<max;
    system("pause");
    return 0;
}
```

در زمان معرفی یک متغیر می‌توان به آن مقدار اولیه داد.

برنامه‌ای بنویسید که ۵۰ عدد صحیح مثبت را دریافت کرده و آنها را به ترتیب عکس دریافت نمایش دهید.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i;
    int n[50];
    for(i=0; i<50;i++)
        cin>>n[i];

    for(i=49; i>=0;i--)
        cout<<n[i]<<"\t";
    system("pause");
    return 0;
}
```

برنامه‌ای بنویسید که دو آرایه ۵ عنصری را دریافت کرده مجموع دو آرایه را در آرایه سوم قرار دهد.

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main()
{
    int a[5], b[5], c[5], i;
    for(i=0;i<5;i++)
    {
        cin>>a[i];
        cin>>b[i];
    }
    for(i=0;i<5;i++)
    {
        c[i]=a[i]+ b[i];
        cout<<a[i]<<"+"<<b[i]<<"="<<c[i]<<"\t";
    }
    system("pause");
    return 0;
}
```

تمرین

برنامه‌ای بنویسید که
- یک آرایه ۲۰ عنصری را دریافت کرده بزرگترین عنصر را به همراه اندیس آن نشان دهد.
- یک آرایه ۱۰ عنصری را دریافت کرده اعلام کند هر عنصر زوج است یا فرد.
- یک آرایه ۵ عنصری دریافت کرده یک واحد به آنها اضافه کرده و نمایش دهد.
- توسط آرایه، نمودار افقی برای اعداد {۵،۱۳،۸،۱۰،۱۵،۱۱} رسم کند.
- یک آرایه را دریافت و در آرایه دیگر به صورت وارونه کپی کرده، آرایه دوم را نمایش دهد.
- عناصر آرایه را گرفته مکعب آنها را محاسبه و در همان عنصر ذخیره نمایید.
- ۳۰ عدد را دریافت کرده، تعداد اعدادی را که به ۴ ختم می‌شوند، نمایش دهد.
- ۱۰ عدد را از ورودی دریافت کرده و در یک آرایه قرار دهد. سپس یک عدد دیگر را خوانده و در آرایه جستجو کند و وجود یا عدم وجود آنرا در آرایه مشخص کند.
- ۱۰ عدد مرتب شده را از ورودی دریافت کرده و در یک آرایه قرار دهد. سپس یک عدد دیگر را خوانده و در جای مناسب در آرایه قرار دهید.
- ۵ عدد را دریافت و در آرایه قرار دهد، اختلاف مجموع اعداد زوج و فرد را نمایش دهد.
- برنامه ای بنویسید که یک عدد را از مبنای ۱۰ به مبنای دو برده و نمایش دهد(با آرایه).

رشته‌ها (strings)

متداول‌ترین کاربرد آرایه‌های یک بعدی رشته‌ها می‌باشند.

هر رشته به صورت یک آرایه کاراکتری ختم شده به تهی null-terminated تعریف می‌شود.

آرایه‌ای که قرار است یک رشته را نگه دارد باید به کاراکتر تهی ختم شود به این معنی که باید یک بایت بزرگتر در نظر گرفته شود تا کاراکتر تهی در آن قرار گیرد. مقدار کاراکتر تهی صفر است.

```
char name[11];
```

حداکثر ۱۰ کاراکتر را نگه می‌دارد.

هر ثابت رشته‌ای نیز به یک کاراکتر تهی ختم می‌شود که کامپایلر به طور اتوماتیک آن را به انتهای رشته اضافه می‌کند.

چند تابع رشته‌ای استاندارد

```
strcpy(to,from);
```
این تابع رشته موجود در from رابه to کپی می‌کند. محتوای from تغییری پیدا نمی‌کند.

```
char str[80];
strcpy(str,"hello");
cout<<str;
```

این تابع دامنه آرایه مقصد را چک نمی‌کند و باید خود اندازه آن را بررسی کنید. (به همراه کاراکتر ختم‌کننده تهی null)

```
strcpy(str,"");
```
رشته‌ای با طول صفر ایجاد می‌کند. به چنین رشته‌ای رشته تهی (null string) می‌گویند.

```
strcat(to,from);
```
این تابع رشته موجود در from رابه to می‌چسباند . محتوای from تغییری پیدا نمی‌کند.

چند تابع رشته‌ای استاندارد

```
char str[80];
strcpy(str,"hello");
strcat(str,"Dear");
cout<<str;
```
این تابع دامنه آرایه مقصد را چک نمی‌کند و باید اندازه آن را بررسی کنید.

```
strcmp(s1,s2);
```
این تابع دو رشته را مقایسه می‌کند
اگر دو رشته یکسان باشد، تابع مقدار صفر می‌گیرد.
اگر s1<s2، مقداری کوچکتر از صفر برمی‌گرداند.
اگر s1>s2، مقداری بزرگتر از صفر برمی‌گرداند.

```
cout<<strcmp("one","one");
```

```
strlen(str);
```
این تابع طول یک رشته را برحسب تعداد کاراکترهای آن باز می‌گرداند.
کاراکتر ختم کننده تهی را به حساب نمی‌آورد.

مثال

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    char str1[80], str2[80];
    int i;
    cout<<"Enter two strings.\n";
    cin>>str1>>str2;
    cout<<"length of str1 is:"<<strlen(str1)<<"\n";
    cout<<"length of str2 is:"<<strlen(str2)<<"\n";
    i=strcmp(str1,str2);
    if(!i) cout<<"the strings are equal";
    if (strlen(str1)+strlen(str2)<80){
        strcat(str1, str2);
        cout<<str1<<"\n";}
    strcpy(str1,str2);
    cout<<str1<<"\n";
    system("pause");
    return 0;
}
```

cin.get()

این تابع یک کاراکتر را از صفحه کلید می‌گیرد. برای استفاده از این تابع در ابتدای برنامه باید از فایل سرآمد iostream استفاده شود.

```
char  x;
x = cin.get( );
cout << x ;
```

```
cin.get (طول آرایه و نام ارایه) ;
cin.get (S,15);
```

در این روش کامپایلر آنقدر از کاربر حرف می گیرد تا به طول آرایه تعریف شده برسد. ممکن است کلمه مورد نظر کاربر ۳ حرفی باشد برای حل این مشکل از روش زیر استفاده کنیم:

```
cin. get (S,15,'*');
```

کاراکتر جدا کننده در واقع شرط پایان است که خودمان تعیین می کنیم و یادمان باشد همواره ۱کاراکتر از کاراکترهای داده شده کم می شود چون از آنجا شرط تمام است.

gets/puts

- برای دریافت متن از تابع gets می توان استفاده کرد.
- برای نمایش متن از تابع puts می توان استفاده کرد.

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
 char t[10];
 gets(t);
 puts(t);
 system("pause");
 return 0;
}
```

toupper() , tolower()

کاراکتر ورودی را به معادل بزرگ خود تبدیل می‌کند.

```
int toupper(int ch);
```

کاراکتر ورودی را به معادل کوچک خود تبدیل می‌کند.

```
int tolower(int ch);
```

باید از header file ، ctype.h استفاده نمود.

یکی از کاربردهای متداول آن پشتیبانی از interface است.

به این معنی که در دریافت ورودی از کاربر مفید است.

مثال

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
    char command[80];
    int i, j;
    for(;;){
        cout<<"operation:add,sub,mul,div,mod,quit\n";
        cin>>command;
        for(i=0;i<strlen(command);i++)
            command[i]=tolower(command[i]);
        if(!strcmp(command, "quit"))
           break;
        cout<<"Enter two No.\n";
        cin>>i>>j;
        if(!strcmp(command, "add"))
            cout<<i+j<<"\n";
        if(!strcmp(command, "sub"))
            cout<<i-j<<"\n";
        if(!strcmp(command, "mul"))
            cout<<i*j<<"\n";
        if(!strcmp(command, "div"))
            cout<<i/j<<"\n";
        if(!strcmp(command, "mod"))
            cout<<i%j<<"\n" ;}
    system("pause");
    return 0;
}
```

string

این نوع داده برای نگهداری رشته ای از یک یا چند کاراکتر مورد استفاده قرار می گیرد. کلاس `string` با متد هایی که در اختیار می گذارد کار با کاراکتر ها و مدیریت متن ها را آسان تر می کند. برای استفاده از `string` باید هدرفایل `string` را به برنامه اضافه کنیم.

```
string str1 = "This class is fun.";
string str2 = "Learn C++.";

//.size() show length of string (18)
cout<< str1.size();

//.empty() if string is empty return 1 else return 0. (0)
cout<< str1.empty();

//oprator [] :you can access to i th character.(c)
cout<< str1[5];
```

string

```
string str1 = "This class is fun.";
 string str2 = "Learn C++.";

//.append(string ) :append string to end of other string.
//(This class is fun.Learn C++.)
 str1.append(str2);

//.erase(int i,int j) :Delet form charcter i th and reply that j time.
//(This class++.)
 str1.erase(10,15);
 cout<< str1<<"\n";

 //.insert(int n,const sting &) : from character n program start embed new string in other string. (This class+Java is good.+.)
 str1.insert(11,"Java is good.");
 cout<< str1;

 //.find(string) search string form left in other string and return position of it. (2)
 cout<< str1.find("is");
```

string

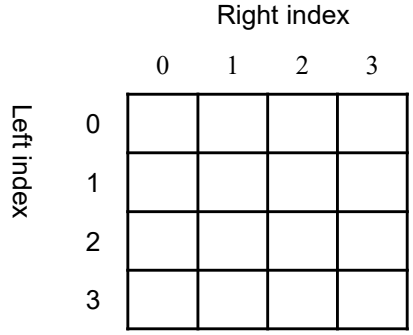
```
//.rfind(string) search string form right in other string and return position of it. (16)
 cout<< str1.rfind("is");

 //operator + : you can connect too string or more by + (ab)
 string str3 = "a";
 string str4 = "b";
 string str5 = "";
 str5 = str3 + str4;
cout<< str5;


 //.compare(string) : do compration operator on to string if to sting equal return 0 if first was biger than other return -1 else return +1. (8)
 cout<< str1.compare(str2);

 //.clear() :assign string to "".
 str1.clear();
 cout<< str1;
```

آرایه های چندبعدی

```
int count[10][12];
```

سطر ستون

یک آرایه دوبعدی از چپ به راست و یک ردیف یک ردیف مورد دستیابی قرار می‌گیرد.



نمایش ذهنی از آرایه‌های دوبعدی

مثال

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
    int two_d[4][5],i,j;
    for(i=0;i<4;i++)
     for(j=0;j<5;j++)
        two_d[i][j]=i*j;
    for(i=0;i<4;i++)
    {
     for(j=0;j<5;j++)
        cout<<two_d[i][j]<<"\t";
     cout<<"\n";
    }
    system("pause");
    return 0;
}
```

Sizeof(int)

4*5*4=80

Output

0	0	0	0	0
0	1	2	3	4
0	2	4	6	8
0	3	6	9	12

Press any key to continue . . .

مقداردهی اولیه آرایه ها

```
type array-name[size]={value-list};
```

```
char k[5]={1,4,9,16,25};
```

0	1	2	3	4
1	4	9	16	25

```
int B[2][3]={{1},{4,9}};
```

0	1	2
1	0	0
4	9	0

```
k[0] =>  1
```

```
k[4] =>  25
```

```
char m[5]={0};
```

0	1	2	3	4
0	0	0	0	0

مثال

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main()
{
    double radians[5][2]={
        1.0, 0.0175,
        2.0, 0.0349,
        3.0, 0.0524,
        4.0, 0.0698,
        5.0, 0.0873
    };
    double degrees;
    int i;
    cin>>degrees;
    for(i=0; i<5; i++)
        if (radians[i][0]==degrees)
        {
            cout<<radians[i][1]<<"\n";;
            break;
        }
    system("pause");
    return 0;
}
```

```
Output
4
0.0698
Press any key to continue . . .
```

آرایه‌های بدون اندازه

```
int pwr[]={1,2,4,8,16,32,64,128};
char prompt[]="Enter your name";
```

آرایه‌هایی که صراحتا ابعاد آنها مشخص نشده است.تعداد کاراکترها و یا اعداد را شمرده و اندازه آرایه را آن تغییر می‌دهد.

```
int sq[][3]={
1, 2, 3,
4, 5, 6,
7, 8, 9
};
```

مزیت این نحوه معرفی آرایه‌ها این است که می‌توان جدول را بدون آنکه ابعاد آرایه را تغییر دهیم، بلندتر یا کوتاهتر نماییم.

جداول رشته‌ای

```
char names[10][40];
```

۱۰ رشته هر کدام با طول حداکثر ۴۰ کاراکتر(با احتساب کاراکتر ختم کننده تهی)

```
cout<<names[0];
```
اولین رشته جدول

```
cin>>names[2];
```
دریافت سومین رشته جدول

ثابت‌های کاراکتری خاص

کد	معنا
\t	tab
\n	newline
\\	back slash کاراکتر

توابع کاراکتری ...is

تابع	کاری که انجام می‌دهد
isalnum(ch)	حروف الفبا یا یک رقم باشد، true برمی‌گرداند.
isalpha(ch)	حروف الفبا باشد، true برمی‌گرداند.
isdigit(ch)	یک رقم باشد، true برمی‌گرداند.
isspace(ch)	فاصله خالی باشد، true برمی‌گرداند.

ctype.h

برنامه‌ای بنویسید که تعداد فاصله‌های یک متن را اعلام می‌کند.

```
#include "stdafx.h"
#include <iostream>
#include <ctype.h>
#include <string.h>

using namespace std;
int main()
{
    char str[]="This is a test";
    int i, spaces;
    spaces=0;
    for(i=0; i<strlen(str); i++)
        if(isspace(str[i]))
            spaces++;
    cout<<spaces<<"\n";
    system("pause");
    return 0;
}
```

Output
3
Press any key to continue . . .

تمرین

برنامه‌ای بنویسید که

- آرایه ۲ بعدی ۱۰ در ۱۰ را با مقادیر جدول ضرب، مقدار دهی کرده و نمایش دهد.
- دو ماتریس ۴*۴ را از ورودی دریافت کرده، تفاضل آن دو را در خروجی نمایش دهد.
- در یک ماتریس ۴*۴ عناصر قطر اصلی را یک و بقیه را صفر قرار دهد و نمایش دهد.
- دو ماتریس ۴*۳ و ۴*۵ را دریافت کرده، حاصلضرب آن دو را نمایش دهد.
- نام ۴۰ دانش‌آموز و نمره آن‌ها را از ورودی دریافت کرده، نام و نمره دانش‌آموزانی که نمره آن‌ها بیشتر از ۱۸ است را نمایش دهد.
- یک رشته راخوانده و تمام حروف بزرگ به کوچک تبدیل نماید و نمایش دهد.
- نمایش کاراکترهای **a** تا **d** به همراه کد اسکی آنها.
- مقداری را به عنوان رمز دریافت کند، در صورتی که رمز درست بود اعلام کند به برنامه خوش آمدید در غیراینصورت اعلام کند رمز اشتباه است.
- یک متن حداکثر ۱۰۰ کاراکتری را در یافت کرده، تعداد تکرار **A**، تعداد تکرار **DA** را نمایش دهد و تمام **K**ها را با **L**جایگزین کند.
- ۵۰ اسم را دریافت کرده، تعداد اسامی که با **B** شروع می‌شود را نمایش دهد.
- دو رشته را از ورودی دریافت کرده، به هم پیوند دهد و در خروجی نمایش دهد.
- رشته ای را از ورودی دریافت کرده، تعداد ارقام موجود در رشته را محاسبه کرده و نمایش دهد.

تابع

```
#include "stdafx.h"
#include <iostream>

using namespace std;
void Areas(int base,int height);
int main()
{
    Areas(10, 20);
    Areas(5, 6);
    Areas(8, 9);
    system("pause");
    return 0;
}
void Areas(int base, int height)
{
    cout<<"Area is:"<<base*height/2<<"\n";
}
```

Prototype

argument

Parameters

تابع

تابع شامل یک یا چند دستور بوده و عمل بخصوصی را انجام می‌دهد.

تابع دارای نامی است و با همین نام فراخوانی می‌شود.

در یک تابع نمی‌توان تابع دیگری ایجاد کرد اما می‌توان تابع دیگری را فراخوانی کرد.

برای استفاده مجدد تابع نوشته شده، بهتر است برای تابع header file ساخته و با پسوند h. در فولدر include که مخصوص header file ها است ذخیره نمود.

prototype شامل سه چیز است: نوع مقدار بازگشتی تابع، تعداد پارامترهای تابع، نوع داده پارامترهای آن تابع

پیش‌الگوی یک تابع (function prototype) برای معرفی آن تابع، پیش از آنکه تعریف شود، استفاده می‌گردد.

تابع

مقدار بازگشتی توابع باید مشخص شود. اگر مشخص نشود اکثرا به عنوان int درنظر گرفته می‌شود. درصورتی‌که تابع دارای مقدار بازگشتی نباشد نوع مقدار بازگشتی void تعریف می‌شود(تابع return ندارد).

به مقداری که به یک تابع ارسال می‌شود argument می‌گویند.

حد بالای آرگومانها توسط کامپایلری که از آن استفاده می‌شود تعیین می‌گردد، اما هر کامپایلر استانداردی حداقل ۲۵۶ آرگومان را قبول می‌کند.

متغیرهایی که آن آرگومانها را دریافت می‌کنند نیز باید معرفی شوند، به این متغیرها parameters گفته می‌شود.

تابع

```
#include "stdafx.h"
#include <iostream>

using namespace std;
float Areas(float base,float height);
int main()
{
    float S,x,y;
    cin>>x>>y;
    S=Areas(x,y);
    cout<<"The area is:"<<S<<"\n";
    system("pause");
    return 0;
}

float Areas(float base, float height)
{
 return base*height/2;
}
```

Output

```
3
4
The area is:6
Press any key to continue . . .
```

توابع ریاضی

math.h استفاده از توابع ریاضی با فایل سرآمد

عملیات ریاضی	تابع در زبان C
cos t	double cos(double t)
e^t	double exp(double t)
x%y	double fmod(double x, double y)
$\log^t$	double log10(double t)
x^y	double pow(double x, double y)
Sin t	double sin(double t)
$\sqrt{t}$	double sqrt(double t)

در C++ می‌توان توابعی به صورت inline تعریف کرد که واقعا فراخوانی نشوند، بلکه در هر نقطه‌ای که فراخوانی شده‌اند کد آنها قرار داده شود. این توابع سریعتر اجرا می‌شوند. پیش از استفاده حتما باید تعریف شود.

```
#include "stdafx.h"
#include <iostream>

using namespace std;

inline int even(int x)
{
    return !(x%2);
}
int main()
{
    if(even(10)) cout<<"10 is even \n";
    if(even(11)) cout<<"11 is even \n";
    if(even(12)) cout<<"12 is even \n";
    system("pause");
    return 0;
}
```

Output

```
10 is even
12 is even
Press any key to continue . . .
```

تابع

زمانی که به انتهای تابع و } یا به دستور return برسیم تابع به روتین فراخواننده خود باز می‌گردد. return می‌تواند مقداری را به روتین فراخواننده‌اش بازگرداند.

یک تابع می‌تواند چندین دستور return داشته باشد.

تابعی که خودش را فرابخواند recursive نامیده می‌شود.

تابع بازگشتی (Recursive)

تابعی که خود را به صورت مستقیم یا از طریق تابع دیگر به صورت غیرمستقیم فراخوانی می کند.

بازگشتی مستقیم:

```
f1(int a)
{
   int b;
   .
   f1(b);
   .
}
```

بازگشتی غیرمستقیم:

```
f1(int a)      f2(int c)
{              {
   int b;         int d;
   .              .
   f2(b);         f1(d);
   .              .
}              }
```

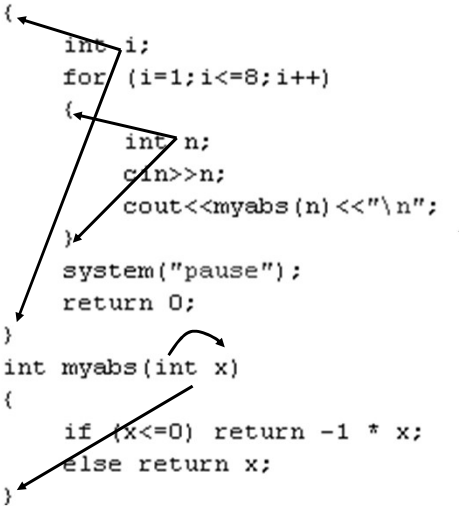
هر تابع بازگشتی دارای یک یا چند مقدار اولیه است که آنرا حالت توقف تابع می نامند. تابع بازگشتی از طریق فراخوانی خودش (حالت بازگشتی) به حالت توقف می رسد.

جملات فیبوناچی تا جمله n

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int fibo(int x);
int main()
{
    int i, n;
    cin>>n;
    for (i=1;i<=n;i++)
        cout<<fibo(i)<<"\t";
     system("pause");
     return 0;
}
int fibo(int x)
{
    if (x<=2) return 1;
    else return fibo(x-1)+fibo(x-2);
}
```

میدان دید (scope)

```
int myabs(int x);
int main()
{
    int i;
    for (i=1;i<=8;i++)
    {
        int n;
        cin>>n;
        cout<<myabs(n)<<"\n";
    }
    system("pause");
    return 0;
}
int myabs(int x)
{
    if (x<=0) return -1 * x;
    else return x;
}
```



روشهای انتقال پارامترهای تابع

call by value, call by reference

value: در این روش مقدار آرگومان به پارامتر صوری آن سابروتین کپی می‌شود بنابراین تغییراتی که روی مقادیر پارامترهای آن سابروتین اعمال می‌شود روی آرگومانهایی که جهت فراخوانی آن سابروتین به کار برده می‌شود تاثیری **نخواهد داشت**.

reference: در این روش به جای مقدار آرگومان آدرس آن به پارامتر مورد نظر کپی می‌شود بنابراین تغییراتی که روی مقادیر پارامترهای آن سابروتین اعمال می‌شود روی آرگومانهایی که جهت فراخوانی آن سابروتین به کار برده می‌شود تاثیر **خواهد داشت**.

call by value

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void f(double x, double y);
int main(){
    double i,j;
    i=6.45;
    j=9.35;
    f(i,j);
    cout<<i<<"\t"<<j<<"\t";
    system("pause");
    return 0;
}
void f(double x, double y){
    x=x+3;
    y=y-3;
    cout<<x<<"\t"<<y<<"\t";;
}
```

Output

```
9.45    6.35    6.45    9.35    Press any key to continue . . .
```

call by reference

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void f(double &x, double &y);
int main(){
    double i,j;
    i=6.45;
    j=9.35;
    f(i,j);
    cout<<i<<"\t"<<j<<"\t";
    system("pause");
    return 0;
}
void f(double &x, double &y){
    x=x+3;
    y=y-3;
    cout<<x<<"\t"<<y<<"\t";
}
```

Output

```
9.45    6.35    9.45    6.35    Press any key to continue . . .
```

تمرین

برنامه‌ای بنویسید که

- مقادیر **k** و **n** را دریافت کرده، و تابع زیر را با فراخوانی تابع فاکتوریل محاسبه کرده و نمایش دهد. **y=n!k!/(k!-(n-k)!)**
- با استفاده از فراخوانی تابع مقدار تابع زیر را محاسبه کرده و نمایش دهد.

$$y = \begin{cases} x^3 + 4x + 5 & x \geq 0 \\ 1 & x < 0 \end{cases}$$

- با استفاده از تابع بازگشتی فاکتوریل عدد دریافتی را محاسبه کرده و نمایش دهد.
- با استفاده از تابع محیط و مساحت مستطیل را محاسبه نماید.
- بزرگترین عدد بین ۲ عدد ورودی با تابع را نمایش دهد.
- با تابع مقدار X^n را برای مقادیر دریافت شده محاسبه نماید.
- مقدار X^n را با تابع بازگشتی محاسبه کرده و نمایش دهد.
- **n** امین جمله سری فیبوناچی را با استفاده از تابع بازگشتی محاسبه کرده و نمایش دهد.
-

کد ASCII

☜ هر کاراگتر یک کد ASCII دارد که می توان در برنامه نویسی از آن استفاده نمود.

☜ کد اسکی ۲۷ مربوط به کلید Esc است.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

int main()
{
    char x;
    for(;;)
    {
        x=getch();
        if(x==27)
            return 1;
    }
    system("pause");
    return 0;
}
```

حلقه بی نهایت

مثال

یک سطر متن انگلیسی که به CTRL Z ختم می شود را دریافت کرده و نمایش می‌دهد.

```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main(){
    char  x;
    while((x = cin.get( )) !=EOF)
        cout << x ;
    system("pause");
    return 0;
}
```

EOF به معنی End of File می‌باشد که در iostream.h تعریف شده و مقدار آن برابر با ۱- می‌باشد. مقدار آن در سیستم عامل DOS عبارتست از ctrl z.

اعدا تصادفی

✓ مقادیر تصادفی در اکثر برنامه‌های کاربردی در زمینه شبیه سازی و بازیهای کامپیوتری نقش مهمی را ایفا می‌نمایند.

✓ برای ایجاد یک عدد تصادفی صحیح بین ۰ و ۳۲۷۶۷ از تابع rand() استفاده می شود.

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
 for(int  j=1; j<=10; j++)
    cout << rand() << "\n";
 system("pause");
 return 0;
}
```

output

```
41
18467
6334
26500
19169
15724
11478
29358
26962
24464
Press any key to continue . . .
```

اعدا تصادفی

✓ هر چند بار برنامهٔ قبل را اجرا نمائیم جواب یکسانی را می گیریم. برای تصادفی کردن اعداد باید از تابع srand() استفاده نمائیم. این تابع به یک آرگومان صحیح از نوع unsigned نیاز دارد که به آن seed گفته می‌شود.

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
 unsigned seed;
 cout << "Enter seed value : " ;
 cin >> seed ;
 srand(seed);
 for(int  j=1; j<=10; j++)
    cout << rand( ) << "\n";
 system("pause");
 return 0;
}
```

output

```
Enter seed value : 4
51
17945
27159
386
17345
27504
20815
20576
10960
6020
Press any key to continue . . .
```

نتیجه پرتاب ۲ تاس

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
 unsigned     seed, d1, d2;
 cout << "Enter seed: " ;
 cin >> seed ;
 srand(seed) ;
 d1= 1+rand( )% 6 ;
 d2= 1+rand( )% 6 ;
 cout << d1 << "          "<< d2 ;
 system("pause");
 return 0;
}
```

output

```
Enter seed: 16
1       3Press any key to continue . . .
```

اعداد تصادفی بین ۰ و ۱

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
 unsigned seed ;
 cout<< "Enter seed: " ;
 cin >> seed ;
 srand(seed) ;
 for(int i=1; i<=10; ++i)
 cout<< rand( ) / 32768.0 <<"\n";
 system("pause");
 return 0;
}
```

output

```
Enter seed: 14
0.00256348
0.827789
0.280518
0.355316
0.677094
0.0376892
0.585052
0.734589
0.373413
0.307465
Press any key to continue . . .
```

struct

هر ساختار از دو یا چند عضو که به همراه هم یک واحد منطقی را می‌سازند، تشکیل می‌شود.

```
struct name{
    type member1;
    type member 2;
    .
    .
    .
    .
    .
}variables;
```

ساختارها شبیه آرایه ها هستند، بدین صورت که یک نوع داده گروهی(جمعی) است که فضای پیوسته از حافظه اصلی را اشغال می نماید. اما عناصر ساختار الزاماً از یک نوع نمی باشند بلکه اعضای یک ساختار می توانند از نوع های مختلف مانند int ،float و ... باشند. اعضای ساختار ربط منطقی دارند. اعضای ساختار را field یا elementگویند.

مثال

نام ساختار

```
struct time
{
int hour ; // 0 to 23
int minute ; // 0 to 59
int second; // 0 to 59
};
```

struct

به دو صورت می توان اعلان یک متغیر از نوع ساختار را نمایش داد:

روش اول

```
struct Info{
int   Id;
char  Id_type;
char  name[80];
} St1, St2, St3;
```

روش دوم

```
struct Info{
int   Id;
char  Id_type;
char  name[80];
};
Info St1, St2, St3;
```

struct

مقدار اولیه برای ساختارها:

```
struct Info{
int   Id;
char  Id_type;
char  name[80];
};
Info St={13, 'p', "Asadi"};
```

به منظور دسترسی به عناصر یک ساختار از عملگر (dot). استفاده می‌گردد.

```
struct Info{
int   Id;
char  Id_type;
char  name[80];
};
Info St;
St.Id=145;
St.Id_type='p';
strcpy(St.name,"Asadi“);
```

struct

عضو یک ساختار خود می‌تواند یک ساختار دیگر باشد. (ساختارهای تو در تو)

```
struct date {
int  month;
int  day;
int  year;
};

struct Info{
int Id;
char name[80];
date entrance; };
```

```
Info x;
x.entrance.month=10;
x.entrance.day=22;
x.entrance.year=1378;
```

struct

عضو یک ساختار خود می‌تواند یک ساختار دیگر باشد.
(روش دیگر)

```
struct Info{
int Id;
char name[80];
  struct date {
      int  month;
      int  day;
      int  year;
      } entrance;
};
```

آرایه ای از ساختارها:

```
struct Info{
int   Id;
char  Id_type;
char  name[80];
};
Info St[30];
```

مثال

نام خانوادگی دانشجویان بزرگتر از ۱۸ سال

```
#include "stdafx.h"
#include <iostream>
using namespace std;
struct students{
  char name[40];
  char family[50];
  int  age;
}st1, st2;
int main()
{
    students st3[100];
    int i;
    for(i=0;i<100;i++)
        cin>>st3[i].name>>st3[i].family>>st3[i].age;
    for(i=0;i<100;i++)
        if(st3[i].age>=18) cout<<st3[i].family<<"\n";
    system("pause");
    return 0;
}
```

تمرین

برنامه‌ای بنویسید که

- چهار تاس برای دو بازیکن (هر کدام دو تاس) در نظر بگیرد. هر بازیکن دو تاس خود را می اندازد. مجموع دو تاس هر بازیکن بیشتر بود آن بازیکن را برنده اعلام کند. در صورتی که مجموع دو تاس هر دو بازیکن برابر بود تساوی اعلام شود.
- با استفاده از ساختار، طول و عرض یک مستطیل را دریافت کرده، مساحت چهار مستطیل را محاسبه کرده و نمایش دهد.
- ساختاری شامل اطلاعات نام، نام خانوادگی، سال ورود، نمره ۱، نمره ۲ و نمره ۳ دانشجویان را در ساختاری ذخیره کند. سپس اطلاعات ۲۰ دانشجو را دریافت کرده و معدل هر یک را به همراه نام خانوادگی نمایش دهد.
- ساختاری شامل اطلاعات نام، نام خانوادگی، سال استخدام، حقوق پایه، بیمه و مالیات کارمندان را در ساختاری ذخیره کند. سپس اطلاعات ۱۰ کارمند را دریافت کرده و حقوق دریافتی هر یک را به همراه نام و نام خانوادگی نمایش دهد.

enum

enum type-name {enumeration-list} variable-list;

می‌توان نوع داده مجتمعی ساخت که از لیستی از ثابت‌های صحیح دارای اسم تشکیل شده باشد.

enum colors {red,green,yellow} mycolor;

ذکر کردن نام enum یا variable list اختیاری است، اما یکی حتما باید ذکر شود.

ثابت های شمارشی رشته نیستند بلکه مقادیر ثابت صحیح با نام هستند.

enum

enum colors {red,green=9,yellow} mycolor;

به متغیر شمارشی فقط مقادیری را می توان نسبت داد که در آن شمارش تعریف شده باشند. (در مثال بالا red, green, yellow)

کامپایلر از سمت چپ مقادیر صحیح را نسبت می دهد، و یک واحد یک واحد اضافه می کند.

در صورتی که یه یک عنصر مقداری نسبت داده شود، عنصر بعدی یک واحد بزرگتر از قبلی خواهد بود.

enum

```
#include "stdafx.h"
#include <iostream>
using namespace std;
enum language{C,CPP,Pascal,BASIC,Ada};

int main()
{
    language p;
    p=BASIC;
    switch(p)
    {
        case C:cout<<"C language";break;
        case CPP:cout<<"C++ language";break;
        case Pascal:cout<<"Pascal language";break;
        case BASIC:cout<<"BASIC language";break;
        case Ada:cout<<"Ada language";break;
    }
    system("pause");
    return 0;
}
```

bit fields

bit-fields عضوی از یک ساختار است که از یک یا چند بیت تشکیل می گردد. با استفاده از آن می توان به کمک یک اسم به یک یا چند بیت موجود در داخل یک byte یا word دسترسی پیدا نمود.

Type name:size;

type: int یا unsigned است. اگر نوع علامت دار باشد، بیت با بالاترین مرتبه به عنوان علامت در نظر گرفته می شود.

کامپایلر عموما میدان های بیتی را در کوچکترین واحدی از حافظه ذخیره می کنند که می تواند آن ها را در خود جای دهد.

```
struct Info{
char name[50];
unsigned department:3;
unsigned month:4;
unsigned vacancy:1;
}I[100];
```

bit fields

لزومی ندارد برای هر بیت نامی مشخص کرد. می توان برای دستیابی به اولین و آخرین بیت یک بایت از میدان بیتی استفاده کرد.

```
struct Info{
unsigned first:1;
unsigned :6;
unsigned last:1;
};
```