



برنامه‌سازی پیشرفته در محیط

Visual C++

مؤلف:

دکتر محمد علی ترکمانی

مهندس الهام ادراکی

سرشناسه	: ترکمانی، محمدعلی، ۱۳۵۴ -
عنوان و نام پدیدآور	: برنامه‌سازی پیشرفته در محیط ++/ مولفان محمدعلی ترکمانی، الهام ادراکی. Visual C.
مشخصات نشر	: مشهد: ارسطو، ۱۳۹۵.
مشخصات ظاهری	: ۲۴۹ص: مصور، جدول، نمودار.
شابک	: 978-600-432-080-1
وضعیت فهرست نویسی	: فیبا
موضوع	: ویزوال سی++ مایکروسافت
موضوع	: ++ Microsoft visual C
موضوع	: سی ++ (زبان برنامه‌نویسی کامپیوتر)
موضوع	: C++ (Computer program language)
شناسه افروده	: ادراکی، الهام، ۱۳۵۶ -
رده بندی کنگره	: ۱۳۹۵ ت۴۹۲س / QAV۶/۷۳
رده بندی دیویی	: ۰۰۵/۲۶
شماره کتابشناسی ملی	: ۴۳۵۹۱۵۳

نام کتاب: برنامه‌سازی پیشرفته در محیط ++ Visual C

موضوع: برنامه نویسی ++ Visual C

موضوع: Visual Studio.net

مؤلفان: دکتر محمدعلی ترکمانی - مهندس الهام ادراکی

ناشر: ارسطو (با همکاری سامانه اطلاع‌رسانی چاپ و نشر ایران)

صفحه‌آرایی، تنظیم و طرح جلد: علی بیات

تیراژ: ۱۰۰۰ جلد

نوبت چاپ: چهارم - ۱۳۹۸

تعداد صفحات: ۲۳۴ ص

چاپ: مدیران

قیمت: ۴۵۰۰۰ تومان

تلفن‌های مرکز پخش: ۰۹۱۷۷۱۶۴۹۴۰ - ۵۰۹۶۱۴۶ - ۰۵۱۱

این اثر مشمول قانون حمایت از مولفان و مصنفان و هنرمندان است. هر کس تمام یا قسمتی از این اثر را بدون اجازه مولف نشر یا پخش یا عرضه کند، مورد پیگرد قانونی قرار خواهد گرفت.

فهرست مطالب

فصل اول: مروری بر برنامه‌سازی ساخت‌یافته در C++ ۱۱

- ۱-۱- زبان C++ ۱۱
- ۱-۲- متغیرها ۱۲
- ۱-۳- ساختار برنامه C++ ۱۳
- ۱-۴- توضیحات در زبان C++ ۱۳
- ۱-۵- دستورات ورودی- خروجی ۱۴
- ۱-۵-۱- دستور printf ۱۴
- ۱-۵-۱-۱- کنترل عمل چاپ در دستور printf ۱۵
- ۱-۵-۱-۲- تعیین کننده‌های فرمت ۱۶
- ۱-۵-۲- دستور scanf ۱۷
- ۱-۶- عملگرها ۱۸
- ۱-۶-۱- عملگرهای محاسباتی ۱۹
- ۱-۶-۲- عملگرهای رابطه‌ای ۱۹
- ۱-۶-۳- عملگرهای منطقی ۲۰
- ۱-۶-۴- عملگرهای بیتی ۲۰
- ۱-۶-۵- عملگرهای ترکیبی ۲۱
- ۱-۶-۶- تقدم عملگرها ۲۱
- ۱-۷- توابع ریاضی ۲۲
- ۱-۸- محاسبه خارج قسمت و باقیمانده تقسیم ۲۲
- ۱-۹- مروری بر دستورات مهم زبان C++ ۲۳
- ۱-۹-۱- پاک کردن صفحه کنسول ۲۳
- ۱-۹-۲- دستور for ۲۴
- ۱-۹-۲-۱- حلقه بینهایت با استفاده از for ۲۵
- ۱-۹-۲-۲- حلقه for تو در تو ۲۵
- ۱-۹-۳- دستور while ۲۶
- ۱-۹-۴- دستور do-while ۲۸
- ۱-۹-۵- دستور break ۲۸

- ۲۹-۶-۱- دستور `continue` ۲۹
- ۲۹-۷-۱- دستور `if` ۲۹
- ۳۰-۸-۱- دستور `switch` ۳۰
- ۳۱-۱۰-۱- آرایه‌ها ۳۱
- ۳۲-۱-۱۰-۱- مقداردهی اولیه به یک آرایه ۳۲
- ۳۲-۱-۱۰-۱-۱- روش اول ۳۲
- ۳۲-۱-۱۰-۱-۲- روش دوم ۳۲
- ۳۳-۱-۱۰-۱-۳- آرایه‌های چند بعدی ۳۳
- ۳۵-۱-۱۰-۱-۴- مقداردهی به آرایه‌های چند بعدی ۳۵
- ۳۶-۱۱-۱- توابع ۳۶
- ۳۸-۱-۱۱-۱- محدوده تعریف متغیرها در زبان `C++` ۳۸
- ۳۸-۱-۱۱-۱-۱- متغیرهای محلی یا `local` ۳۸
- ۳۹-۱-۱۱-۱-۲- متغیر سراسری یا `global` ۳۹
- ۴۰-۱۲-۱- ارسال آرایه به یک تابع (به عنوان پارامتر) ۴۰
- ۴۱-۱۳-۱- آرگومان پیش فرض ۴۱
- ۴۲-۱۴-۱- پیش محاسبه گرها یا ماکروها ۴۲
- ۴۳-۱-۱۴-۱- دستور `define` ۴۳
- ۴۴-۱۵-۱- رشته‌ها ۴۴
- ۴۴-۱-۱۵-۱- تعریف رشته ۴۴
- ۴۴-۲-۱۵-۱- توابع کار روی رشته‌ها ۴۴
- ۴۴-۱۵-۲-۱- `strcpy` ۴۴
- ۴۵-۱۵-۲-۲- `strcmp` ۴۵
- ۴۶-۱۵-۲-۳- `strchr` ۴۶
- ۴۶-۱۵-۲-۴- `strstr` ۴۶
- ۴۶-۱۵-۲-۵- `strlen` ۴۶
- ۴۷-۱۵-۲-۶- `strcat` ۴۷
- ۴۷-۱۵-۲-۷- `gets` ۴۷
- ۴۷-۱۵-۲-۸- `puts` ۴۷
- ۴۷-۱۵-۲-۹- دستور `strlwr` ۴۷
- ۴۸-۱۵-۲-۱۰- دستور `strupr` ۴۸

- ۴۸ ۱۱-۲-۱۵-۱- دستور tolower
- ۴۸ ۱۲-۲-۱۵-۱- دستور toupper
- ۴۸ ۳-۱-۵-۱- انواع تبدیل رشته به عدد و بالعکس
- ۴۸ ۱-۱۵-۳-۱- تابع atoi
- ۴۹ ۲-۱۵-۳-۱- تابع itoa
- ۵۰ ۱-۱۶- تمرین ها
- ۵۱ ۱-۱۶- سنوالات چهارگزینه‌ای
- ۵۲ پاسخ ها

فصل دوم: اشاره گرها، تخصیص پویای حافظه و مراجع ۵۳

- ۵۳ ۱-۲- اشاره گرها
- ۵۸ ۲-۲- عملگرهای NEW و DELETE
- ۶۱ ۱-۲-۲- مقداردهی اولیه
- ۶۱ ۲-۲-۲- اشغال دینامیکی آرایه
- ۶۳ ۳-۲-۲- تابع مرتبط با عملگر new
- ۶۴ ۳-۲- مراجع
- ۶۵ ۱-۳-۲- تبدیل دو ریفرنس متفاوت
- ۶۶ ۴-۲- اشاره گر به تابع
- ۶۸ ۵-۲- انواع فراخوانی توابع در C++
- ۶۹ ۱-۵-۲- فراخوانی با مقدار
- ۶۹ ۲-۵-۲- فراخوانی با آدرس
- ۷۰ ۳-۵-۲- فراخوانی با ارجاع
- ۷۱ ۲-۶- تمرین ها
- ۷۲ ۲-۶- سنوالات چهارگزینه‌ای
- ۷۴ پاسخ ها:

فصل سوم: تعریف ساختار ۷۵

- ۷۵ ۱-۳- دستور STRUCT
- ۷۸ ۱-۳-۱- تعریف استراکچر به صورت آرایه
- ۸۰ ۲-۳-۱- تعریف struct به صورت اشاره گر

۸۱	UNION-۳-۲
۸۳	۳-۳-تمرین‌ها
۸۴	۳-۳-سئوالات چهارگزینه‌ای
۱۵	پاسخ‌ها:

فصل چهارم: برنامه‌سازی شیء‌گرا..... ۸۷

۸۷	۴-۱-تکنیک‌های برنامه‌نویسی
۸۸	۴-۲-خصوصیات برنامه‌نویسی شیء‌گرا
۹۳	۴-۳-انواع زبان‌های شیء‌گرا از نظر نحوه استفاده از شیء‌گرایی
۹۴	۴-۴-انواع زبان‌های شیء‌گرا از نظر تعداد کلاس پدر
۹۴	۴-۵-مفهوم فضای نام و فضای نام STD
۹۴	۴-۶-دستورات ورودی - خروجی در ++C
۹۵	۴-۷-قواعد تعریف کلاس
۹۶	۴-۷-۱-کنترل نحوه دسترسی به اعضای کلاس
۹۶	۴-۷-۱-۱-تعریف اعضای کلاس به صورت عمومی (public)
۹۷	۴-۷-۱-۲-تعریف اعضای کلاس به صورت خصوصی (private)
۹۷	۴-۷-۱-۳-تعریف اعضای کلاس به صورت محافظت شده (protected)
۹۸	۴-۸-توابع عضو کلاس
۹۹	۴-۸-۱-توابع inline
۱۰۰	۴-۹-محدوده تعریف کلاس
۱۰۰	۴-۱۰-تعریف اشیاء
۱۰۵	۴-۱۱-سازنده یک کلاس
۱۰۵	۴-۱۱-۱-نحوه اجرای تابع سازنده در برنامه
۱۰۶	۴-۱۲-تابع تخریب‌گر یک کلاس
۱۰۹	۴-۱۳-مقداردهی اولیه در توابع سازنده
۱۱۱	۴-۱۴-ویژگی‌های توابع سازنده و تخریب‌گر
۱۱۲	۴-۱۵-تعریف چند تابع سازنده برای یک کلاس
۱۱۴	۴-۱۶-توابع عضو ثابت
۱۱۴	۴-۱۷-اعضای کلاس با ویژگی STATIC
۱۱۴	۴-۱۷-۱-اعضای داده‌ای static
۱۱۶	۴-۱۷-۲-توابع عضو استاتیک
۱۲۰	۴-۱۸-اشاره‌گرهایی به اشیاء
۱۲۱	۴-۱۹-اشاره‌گر THIS
۱۲۲	۴-۲۰-تخصیص پویای اشیاء

- ۱۲۴ ۴-۲۱- آرایه‌ای از اشیاء
- ۱۲۶ ۴-۲۱-۱- مقداردهی اولیه به آرایه‌ای از اشیاء با استفاده از فرم ALTERNATIVE
- ۱۲۷ STUDENT OB[4] [2];
- ۱۲۷ ۴-۲۲- اشغال دینامیکی آرایه‌ای از اشیاء
- ۱۲۸ ۴-۲۳- عبور اشیاء به توابع
- ۱۲۸ ۴-۲۳-۱- اشیاء به عنوان آرگومان توابع
- ۱۲۹ ۴-۲۳-۲- عبور آدرس شیء به تابع
- ۱۳۱ ۴-۲۳-۳- ارسال اشیاء به صورت مرجع به توابع
- ۱۳۳ ۴-۲۴- اشیاء به عنوان مقدار بازگشتی توابع
- ۱۳۷ ۴-۲۵- دستور العمل FRIEND
- ۱۳۸ ۴-۲۵-۱- نکاتی در مورد friend
- ۱۴۳ ۴-۲۶- تمرین‌ها
- ۱۴۶ ۴-۲۷- سنوالات چهارگزینه‌ای

۱۵۱ فصل پنجم: وراثت

- ۱۵۱ ۵-۱- تعریف وراثت
- ۱۵۱ ۵-۲- کلاس‌های پایه و مشتق شده
- ۱۵۲ ۵-۳- قواعد ارث‌بری و کنترل دسترسی به اعضای کلاس پایه
- ۱۵۲ ۵-۳-۱- کلاس پایه با پیشوند public
- ۱۵۲ ۵-۳-۲- کلاس پایه با پیشوند protected
- ۱۵۳ ۵-۳-۳- کلاس پایه با پیشوند private
- ۱۵۳ ۵-۴- نحوه نوشتن کلاس‌های پایه، مشتق شده و تعریف توابع کلاس پایه
- ۱۵۴ ۵-۵- وراثت چندگانه
- ۱۵۴ ۵-۶- مثال‌هایی از وراثت
- ۱۶۶ ۵-۷- تمرین‌ها
- ۱۶۷ ۵-۸- سنوالات چهارگزینه‌ای
- ۱۶۸ پاسخ‌ها:

۱۶۹ فصل ششم: تابع مجازی

- ۱۶۹ ۶-۱- تابع مجازی
- ۱۷۱ ۶-۱-۱- اشاره‌گر به نوع پایه و مشتق شده
- ۱۷۴ ۶-۱-۲- کلاس‌های مجرد و توابع مجازی محض

- ۱۷۶ ۳-۱-۶-سازنده‌ها و مخرب‌ها در چند ریختی
- ۱۷۸ ۲-۶-تمرین
- ۱۷۹ ۳-۶-سئوالات چهارگزینه‌ای
- ۱۸۰ پاسخ‌ها:

فصل هفتم: سر بار گذاری عملگرها.....۱۸۱

- ۱۸۱ ۱-۷-تعریف مجدد عملگرها
- ۱۸۲ ۲-۷-محدودیت‌های تعریف مجدد عملگرها
- ۱۸۲ ۳-۷-تعریف مجدد عملگرها به کمک تابع عضو کلاس
- ۱۸۷ ۴-۷-تعریف مجدد عملگرها به کمک تابع دوست
- ۱۸۹ ۵-۷-تعریف مجدد عملگرهای << و >>
- ۱۹۲ ۶-۷-تعریف مجدد عملگرهای ترکیبی
- ۱۹۵ ۷-۷-تعریف مجدد عملگرهای ++ و -- به کمک تابع دوست
- ۱۹۸ ۱-۷-قابلیت‌های تابع دوست در تعریف مجدد عملگرها
- ۱۹۹ ۸-۷-تمرین‌ها
- ۱۹۹ ۹-۷-سئوالات چهارگزینه‌ای
- ۲۰۰ پاسخ‌ها :

فصل هشتم: فایل‌ها.....۲۰۱

- ۲۰۱ ۱-۸-مقدمه
- ۲۰۱ ۲-۸-باز کردن فایل
- ۲۰۲ ۱-۲-۸-نوع بازکردن
- ۲۰۲ ۸-۱-۲-تفاوت r و $r+$
- ۲۰۲ ۲-۸-۲-مقایسه $w+$ و $r+$
- ۲۰۳ ۳-۸-کار با فایل‌های باینری
- ۲۰۳ ۴-۸-دستور FCLOSE
- ۲۰۳ ۵-۸-دستور FCLOSEALL
- ۲۰۴ ۶-۸-دستورات خواندن و نوشتن فایل
- ۲۰۴ ۱-۶-۱-دستور `fprintf`
- ۲۰۴ ۲-۶-۱-دستور `fscanf`
- ۲۰۴ ۳-۶-۱-دستور `fgets`
- ۲۰۵ ۴-۶-۱-دستور `getc`

۲۰۵ ungetc	۸-۶-۵-دستور
۲۰۶ fwrite و fread	۸-۶-۶-دستورات
۲۰۶ EOF	۸-۷-کاراکتر اکتز
۲۰۷ FSEEK	۸-۸-دستور
۲۰۷ FTELL	۸-۹-دستور
۲۰۷ REWIND	۸-۱۰-دستور
۲۰۸ FEOF	۸-۱۱-تابع
۲۱۱ REMOVE	۸-۱۲-فرمان
۲۱۱ RENAME	۸-۱۳-فرمان
۲۱۱	۸-۱۴-تمرین ها
۲۱۲	۸-۱۵-سئوالات چهارگزینه ای
۲۱۲	پاسخ ها

فصل نهم: استثناها و خطاها ۲۱۵

۲۱۵	۹-۱-مقدمه
۲۱۵	۹-۲-خطاها
۲۱۶	۹-۳-مدیریت استثناها
۲۱۶ TRY-CATCH	۹-۴-بلاک های
۲۲۲	۹-۵-تمرین ها
۲۲۴	۹-۶-سئوالات چهارگزینه ای
۲۲۵	پاسخ ها

پیوست ها ۲۲۵

۲۲۵ VISUAL STUDIO .NET	۱: آشنایی با
۲۳۲	پیوست ۲: کتابخانه های مهم
۲۳۳	پیوست ۳: جدول کدهای اسکی

منابع: ۲۳۴

مقدمه:

برنامه‌سازی پیشرفته و برنامه‌سازی شیء‌گرا دروسی هستند که در رشته‌های مهندسی کامپیوتر، IT و ICT ارائه می‌گردند. با توجه به نیاز دانشجویان و اساتید به مرجعی مناسب این کتاب ارائه گردید و این کتاب برخلاف سایر کتابهای موجود در بازار از ویژگیهای زیر برخوردار است:

۱- پوشش سرفصلهای مصوب وزارت علوم

۲- حجم مناسب و قابل ارائه در یک نیمسال تحصیلی

۳- بیان ساده و کاربردی

۴- به روز بودن مطالب: برنامه‌های ارائه شده در این کتاب با آخرین نسخه Visual Studio.net سازگار هستند و به راحتی در کنسول Visual C++.NET اجرا میشوند.

۵- ارائه نمونه سؤال در پایان هر بخش که به دانشجویان کمک میکند بهتر بتوانند خود را برای آزمون‌هایی که در پیش رو دارند آماده نمایند.

امید است این اثر مورد توجه همکاران و دانشجویان گرامی قرار گرفته و به افزایش کیفیت آموزشی دروس برنامه‌سازی پیشرفته و برنامه‌سازی شیء‌گرا کمک نماید. از اساتید و دانشجویان گرامی تقاضا داریم نقطه نظرات خود را از طریق ایمیل m.a.torkamani@gmail.com با مولفین در میان بگذارند تا انشاءالله در ویرایش‌های بعدی اشکالات یا کاستی‌های احتمالی کتاب مورد تجدید نظر قرار گیرد. در پایان وظیفه خود می‌دانم از زحمات آقای مهندس بیات به خاطر طراحی جلد کتاب و همچنین از مدیریت سامانه اطلاع رسانی چاپ و نشر ایران جناب آقای حسین قنبری تشکر و قدردانی نمایم.

محمد علی ترکمانی

تابستان ۱۳۹۵

فصل اول

مروری بر برنامه‌سازی ساخت‌یافته در C++

۱-۱- زبان C++

C++ گونه توسعه یافته زبان C است که در اوایل دهه ۱۹۸۰ توسط بیارنه استراوسروپ ادر آزمایشگاه بل ایجاد گردید. C++ شامل تمامی امکانات زبان C است و علاوه بر آن دارای امکانات برنامه‌نویسی شیء‌گرا (OOP) می‌باشد (شکل ۱-۱). بنابراین می‌توان یک برنامه C را با کامپایلر C++ کامپایل کرد.

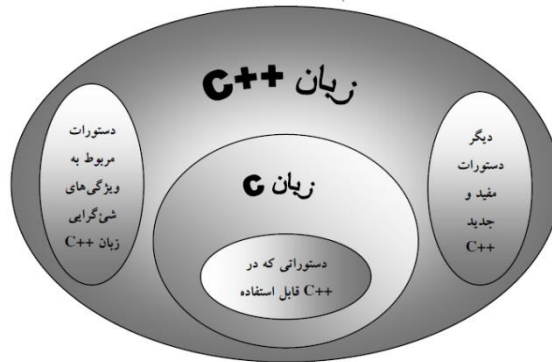
تاکنون کامپایلرهای مختلفی از C++ به بازار عرضه شده است که مهم‌ترین آنها عبارتند از Turbo C++ 3 محصول شرکت Boardland، Visual C++ 6 و Visual C++.Net که محصول شرکت ماکروسافت می‌باشند. Visual C++.Net جزئی از بسته نرم‌افزاری Visual Studio.Net است. در این مجموعه ابزارهای قدرتمند برنامه‌نویسی زیر وجود دارد:

- Visual C++.Net
- .Net#Visual C
- Visual F#.Net
- Visual Basic .Net

در این کتاب از کامپایلر Visual C++.Net 2010 استفاده شده است.

1 Bjarne Stroustrup

2 Object oriented programming



شکل ۱-۱: رابطه برنامه‌نویسی C و C++

۲-۱- متغیرها

در زبان C++ هر متغیر، پیش از آنکه در دستوری از برنامه به کار برده شود، باید تعریف گردد. انواع متغیرها در زبان C++ در جدول ۲-۲ نمایش داده شده است.

جدول ۱-۱: انواع متغیرها در زبان C++

نوع	بایت	محدوده مقادیر
int	4	2,147,483,647 تا -2,147,483,648
unsigned int/ unsigned unsigned long /unsigned long int	4	4,294,967,295 تا 0
bool	1	true یا false
char	1	127 تا -128
unsigned char	1	255 تا 0
short/short int	2	32,767 تا -32,768
unsigned short/ unsigned short int	2	65,535 تا 0
long/ long int	4	2,147,483,647 تا -2,147,483,648
long long	8	9,223,372,036,854,775,808 تا -9,223,372,036,854,775,808 معادل با یک عدد صحیح ۶۴ بیتی است
float	4	3.4E +/- 38 (رقم 7)
double / long double	8	1.7E +/- 308 (رقم 15)
wchar_t	2	دو بایت برای ذخیره کاراکترهای Unicode

۳-۱- ساختار برنامه C++

هر برنامه C++ دارای یک یا چند تابع است که یکی از آنها برنامه اصلی یا main است. در C++ همه دستورات و زیر برنامه‌ها به عنوان یک تابع می‌باشند. شروع برنامه از تابع main می‌باشد. یک بلاک در C++ با آکولاد { } مشخص می‌شود.

به طور کلی می‌توان گفت که هر تابع C++ حداقل شامل اجزای زیر است:

```
int main(int argc, _TCHAR* argv[])
{
    شروع تابع اصلی
    variables declaration; // تعریف متغیرها
    program statements;    // دستورات برنامه
    return 0;
    پایان تابع اصلی
}
```

تعریف توابع دیگر ممکن است قبل یا بعد از تابع اصلی قرار گیرد که در قسمت توابع آنرا شرح می‌دهیم.

درون پرانتز آرگومان‌های ورودی است و قبل از main نیز نوع بازگشتی را نشان می‌دهد.

۴-۱- توضیحات در زبان C++

در زبان C++ توضیحات را باید بعد از // قرار داد.

مثال:

```
//program by M.A.Torkamani
```

نکته: در زبان C توضیحات را باید در بین /* */ قرار دهیم. در زبان C++ نیز استفاده از

این روش قابل قبول است.

```
/* sum of two numbers */
```

¹ Comment

۵-۱-۱- دستورات ورودی - خروجی

۵-۱-۱- دستور printf

این فرمان جهت چاپ اطلاعات روی صفحه نمایش به کار می‌رود.

شکل کلی دستور به صورت زیر است:

```
int printf(const char *control_string, ...);
```

این تابع یک عدد را بر می‌گرداند که این عدد تعداد کاراکترهای چاپ شده روی صفحه است و یا یک عدد منفی است که نشان دهنده رخ دادن خطا است.

`control_string` می‌تواند دو نوع آیتم باشد. نوع اول رشته‌ای است که قرار است چاپ شود و نوع دوم تعیین کننده فرمت است که چگونگی نمایش آرگومان روی صفحه را مشخص می‌کند و با یک علامت درصد شروع می‌شود. ابتدا مثالی از نوع اول ارائه می‌گردد. تعیین کننده فرمت بعداً شرح داده می‌شود.

مثال: برنامه زیر موجب چاپ رشته `hello` روی مانیتور خواهد شد.

```
#include "stdafx.h"
int main(int argc, _TCHAR* argv[])
{
    printf ("hello") ;
    return 0;
}
```

نکته: یک هدر فایل^۲ فایلی با پسوند `.h` است که یک سری توابع در آن تعریف شده است و ما می‌توانیم از آن استفاده کنیم. هدر فایل‌ها به وسیله فرمان `#include` به برنامه ما ضمیمه می‌شوند. هر یک از فرمان‌های زبان `C++` درون یک یا چند فایل که با نام `header file` (یا `library`) شناخته می‌شود، تعریف شده است. برای اینکه برنامه بدون خطا اجرا شود باید `header file` فرمان مورد نظر را قبل از تابع `main` بنویسید. به عنوان مثال دستورات پر کاربرد زبان `Visual C++ 2010` درون کتابخانه `"stdafx.h"` تعریف شده‌اند. برای استفاده از این هدر فایل باید فرمان `#include "stdafx.h"` را قبل از تابع `main` بنویسید.

Format Specifier

header file

تذکر: وقتی که برنامه به اتمام می‌رسد به محیط ++C بر می‌گردد. جهت دیدن جواب‌ها و برگشتن به ++C، یعنی رفت و آمد بین محیط ++C و کنسول (محیط مشکی رنگ شبیه به محیط DOS) از CTRL+F5 استفاده می‌شود.

تذکر: اگر بخواهید وقتی که برنامه به پایان رسید، در محیط ++C باقی بماند و وقتی که یک کلید را فشار دهید به محیط ++C برگردد، باید دستور getch_ را در انتهای برنامه بنویسید. این دستور منتظر می‌ماند تا شما یک کلید را فشار دهید. این دستور در هدر فایل conio.h تعریف شده است:

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
    printf ("hello") ;
    _getch();
    return 0;
}
```

۱-۱-۵-۱- کنترل عمل چاپ در دستور printf

جدول ۱-۲ برخی از کاراکترهای کنترلی مهم که در دستور printf قابل استفاده هستند را نشان می‌دهد.

جدول ۱-۲: کاراکترهای کنترلی در زبان ++C

ردیف	علامت	توضیح
۱	(new line) \n	به ابتدای خط بعدی می‌رود.
۲	(tab) \t	۸ ستون به جلو می‌رود.
۳	"	برای چاپ " به کار می‌رود.
۴	\\	برای چاپ \ به کار می‌رود.
۵	(back space) \b	یک حرف را پاک می‌کند.

مثال: برنامه زیر را اجرا کرده و خروجی آنرا مشاهده نمایید.

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
    int a;
```

```
printf("salam\b ali");
printf("\nsalam\ ali");
printf("\nsalam\t ali");
printf("\nsalam" ali");
printf("\nsalam\n ali");
_getch();
return 0;
}
```

۲-۱-۵-۱- تعیین کننده‌های فرمت

فرض کنید بخواهید مقدار متغیر A را چاپ کنید. اگر فرمان زیر را به کار ببرید، کامپیوتر حرف A را روی صفحه چاپ می‌کند:

```
printf("A");
```

برای اینکه به کامپیوتر بفهمانیم که می‌خواهیم مقدار متغیر A را چاپ کنیم و نه حرف A را از یک تعیین کننده فرمت استفاده کنیم. تعیین کننده‌های فرمت در جدول ۱-۳ ارائه شده‌اند. بنابراین اگر بخواهیم یک حرف را چاپ کنیم، آنرا درکیشن دوتایی قرار می‌دهیم، اما اگر بخواهیم یک متغیر را چاپ کنیم باید از درصد متناسب با نوع آن متغیر استفاده کنیم.

مثال:

```
int A ;
A=2;
printf ("A" ) ;           // حرف A را چاپ می کند.
printf ( "%d" , A);       // 2 را چاپ می کند.
```

مثال: اگر $A=2.5$ و $B=2.5*10000000$ باشد به کار بردن %f ، %e و %g موجب

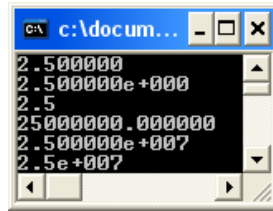
چاپ چه چیزی می‌شود؟

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
    float a=2.5;
    printf("%f", a);
    printf("\n%e", a);
    printf("\n%g", a);
    float b=2.5*10000000;
    printf("\n%f", b);
    printf("\n%e", b);
    printf("\n%g", b);
    _getch();
}
```



```
return 0;
}
```

خروجی برنامه:



جدول ۱-۳: تعیین کننده‌های فرمت

کاراکتر	نوع	فرمت خروجی
C	int	یک کاراکتر
D	int	یک عدد صحیح
o	int	مبنای ۸
u	int	اعداد صحیح بدون علامت
x	int	مبنای ۱۶ با استفاده از حروف abcdef
X	int	مبنای ۱۶ با استفاده از حروف ABCDEF
e	double	نماد علمی (توانی)
f	double, float	double یا float (معمولی)
g	double	توانی یا معمولی، هر کدام که ساده تر باشد نوشته می‌شود
p	Pointer to void	آدرس آرمومان را به صورت مبنای ۱۶ چاپ می‌کند.
s	string	رشته‌ها

۲-۵-۱- دستور scanf

این دستور برای دریافت اطلاعات از ورودی به کار می‌رود. شکل کلی این دستور به صورت

زیر است:

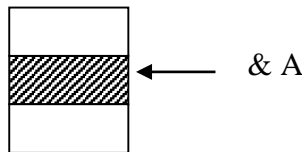
```
int scanf (const char *control_string, ...);
```

تابع `scanf` یک عدد را بر می‌گرداند که این عدد تعداد آیت‌هایی است که با موفقیت به یک

متغیر نسبت داده شده‌اند. در اینجا `control_string` نحوه خواندن متغیرها از لیست

آرگومانهای تابع را نشان می‌دهد. برای `control_string` می‌توان از همان درصدهای ذکر شده در جدول ۲-۴ استفاده نمود.

نکته مهم در مورد `scanf` این است که شکل کلی این دستور شبیه به `printf` است، با این تفاوت که در این دستور به جای کار کردن با خود متغیر، با خانه‌ای از حافظه که متغیر در آن قرار دارد، کار می‌کنیم. برای انجام این کار، از کاراکتر `&` (امپرسن) استفاده می‌کنیم. اگر متغیری به نام `A` داشته باشیم، `&A` به آدرس `A` در حافظه اشاره می‌کند. `&A` در واقع یک اشاره‌گر به خانه‌ای از حافظه است که `A` در آن قرار دارد (شکل ۱-۲).



شکل ۱-۲: اشاره‌گر به آدرس متغیر `A`

مثال:

```
int A;
scanf("%d", &A);
```

نکته: وارد کردن داده‌ها دقیقاً باید مانند آنچه در دستور `scanf_s` است، انجام شود.

مثال:

```
scanf("%f ,%f", &B,&A);
```

در اینجا باید داده‌ها را به صورت زیر وارد کنیم:

7 , 5

در `B` مقدار ۷ و در `A` مقدار ۵ به فرمت اعشاری ذخیره می‌گردد.

۶-۱- عملگرها

در زبان `visual c++` عملگرها به چند دسته تقسیم می‌شوند:

- عملگرهای محاسباتی

- عملگرهای رابطه‌ای
- عملگرهای منطقی
- عملگرهای بیتی
- عملگرهای ترکیبی

۱-۶-۱- عملگرهای محاسباتی

عملگرهای محاسباتی در جدول ۴-۱ نمایش داده شده‌اند.

جدول ۴-۱: عملگرهای محاسباتی

مثال	وظیفه عملگر	عملگر
$x-y$ یا $-x$	تفریق یا منهای یکانی	-
$x+y$	جمع	+
$X*y$	ضرب	*
x/y	خارج قسمت تقسیم	/
$x\%y$	باقیمانده تقسیم صحیح	%
$--x$ یا $x--$	کاهش یک واحد	--
$++x$ یا $x++$	افزایش یک واحد	++

۲-۶-۱- عملگرهای رابطه‌ای

این نوع عملگرها رابطه بین عملوندها را مشخص می‌کنند. جدول ۵-۱ عملگرهای رابطه‌ای را نشان می‌دهد.

جدول ۵-۱: عملگرهای رابطه‌ای

رابطه	عملگر
بزرگتر	>
بزرگتر یا مساوی	>=
کوچکتر	<
کوچکتر یا مساوی	<=
مساوی	==
نامساوی (مخالف با)	!=

تذکر: = برای جایگزینی و == برای مقایسه (در دستورات شرطی) به کار می‌رود.
مثال: عبارت منطقی آیا X با Y برابر است، به صورت X==Y نوشته می‌شود.
تذکر: در زبان C++ نتیجه یک عبارت شرطی درست ۱ و یک عبارت شرطی غلط صفر است.

۳-۶-۱- عملگرهای منطقی

عبارات منطقی عباراتی هستند که دارای ارزش درست یا غلط است. جدول ۶-۱ عملگرهای منطقی به ترتیب اولویت را نمایش می‌دهد.

جدول ۶-۱: عملگرهای منطقی

عملگر	نام	مثال
!	نقیض (not)	!X
&&	و (and)	x>y && m<n
	یا (or)	m<n x>y

۴-۶-۱- عملگرهای بیتی^۱

عملگرهای بیتی، عملگرهایی هستند که برای کار روی بیتها استفاده می‌شوند. جدول ۷-۱ اولویت عملگرهای بیتی را نشان می‌دهد.

جدول ۷-۱: اولویت عملگرهای بیتی

عملگر	وظیفه
&	And
	OR
^	XOR
~	Not
>>	یا شیفت به راست Right Shift
<<	یا شیفت به چپ Left Shift

مثال: شیفت به راست و چپ

```
#include "stdafx.h"
#include "conio.h"
```

```
int main(int argc, _TCHAR* argv[])
{
    unsigned char C=4;
    C=C >> 1;
    printf("\n%d",C);
    C=8;
    C=C << 1;
    printf("\n%d",C);
    _getch();
    return 0;
}
```

خروجی برنامه فوق به ترتیب ۲ و ۱۶ است.

۵-۶-۱- عملگرهای ترکیبی

این عملگرها از ترکیب عملگرهای محاسباتی و علامت = ایجاد می‌شوند. کار این عملگرها یک عمل محاسباتی و یک عمل انتساب است. اولویت این عملگرها از سایر عملگرها کمتر است. جدول ۱-۸ این عملگرها را نشان می‌دهد.

جدول ۱-۸: عملگرهای ترکیبی

دستور	دستور معادل
A += B ;	A = A + B ;
A -= B ;	A = A - B ;
A *= B ;	A = A * B ;
A /= B ;	A = A / B ;
A %= B ;	A = A % B

۶-۶-۱- تقدم عملگرها

جدول ۱-۹ تقدم عملگرها را نشان می‌دهد. در این جدول، عملگرهایی که در یک ردیف هستند دارای اولویت یکسان هستند. این عملگرها هر کدام که زودتر ظاهر شوند اول اجرا می‌شوند.

جدول ۱-۹: تقدم عملگرها

بالاترین اولویت	()
	-- ++ ~ !
	./ *

- +	
<< >>	
<= > >= <	
== !=	
&	
^	
&&	
+= -= *= /= = %=	پایین ترین اولویت

۷-۱- توابع ریاضی

توابع ریاضی نظیر قدر مطلق (abs)، جذر یا ریشه دوم (sqrt) و توان‌رسانی (pow)، sin و cos در هدر فایل math.h تعریف شده‌اند.

تمرین: برنامه‌ای بنویسید که دو عدد را از ورودی گرفته و به وسیله تابع pow، عدد اول را به توان عدد دوم برساند.

۸-۱- محاسبه خارج قسمت و باقیمانده تقسیم

همانطور که ذکر شد برای محاسبه خارج قسمت و باقیمانده تقسیم از عملگرهای / و % استفاده می‌شود. اگر A و B از نوع صحیح باشد، A/B از نوع صحیح و اگر A و B از نوع اعشاری باشد، A/B از نوع اعشاری است.

مثال:

```
int A = 5 , B = 2 ;
float M;
M = (float) A/B;
```

در واقع می‌گوییم که نوع A /B را از نوع float در نظر بگیرد. یعنی عمل تقسیم را به صورت اعشاری انجام بده. اگر (float) را ننویسیم به جای 2.5، عدد صحیح 2 را بر می‌گرداند. به این کار Type Casting می‌گویند. به برنامه زیر توجه کنید:

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
    int A = 5 ,B = 2 ;
    float M,N;
    N = A/B;
    M = (float) A/B;
    printf("\nM=%f\nN=%f",M,N);
    _getch();
    return 0;
}
```

خروجی برنامه:



۹-۱- مروری بر دستورات مهم زبان C++

۹-۱-۱- پاک کردن صفحه کنسول

در Visual C++ .NET برای پاک کردن صفحه کنسول از دستور زیر استفاده می‌شود.

```
system("cls");
```

توجه داشته باشید که برای استفاده از این دستور باید کتابخانه‌ای که این دستور در آن تعریف شده است را در بالای برنامه تعریف کنید:

```
#include <stdlib.h>
```

۲-۹-۱-دستور for

این دستور برای ایجاد حلقه به کار می‌رود. به عبارت دیگر وقتی از این دستور استفاده می‌شود که بخواهیم یک یا چند دستور به بیش از یک مرتبه (به عنوان مثال ۱۰۰ مرتبه) اجرا شوند.

حلقه for دارای یک شاخص حلقه است که شرط خروج از حلقه بر اساس آن شاخص تعیین می‌شود.

شکل کلی دستور به صورت زیر است:

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

که در آن:

initialization: مقدار اولیه شاخص حلقه

condition: شرط پایان حلقه

increment: گام حلقه که میزان افزایش یا کاهش شاخص حلقه را مشخص می‌کند.

مثال:

```
for (i=5 ; i <=100; i + = 7)
```

```
بدنه حلقه  
{
```

در این مثال ۵ مقدار اولیه، ۱۰۰ مقدار نهایی و ۷ گام یا step حلقه است.

مثال: در قطعه برنامه زیر شاخص حلقه فقط یک واحد اضافه شود:

```
for (i=0 ; i <=10; i ++)
```

```
بدنه حلقه  
{
```

تذکر: اگر در بدنه for فقط یک دستورالعمل وجود داشته باشد، نیازی به نوشتن آکولاد نیست.

مثال:

```
for (i = 1; i <50; i++)  
{  
    s=s + i;  
}
```

برنامه فوق را می‌توان به صورت زیر نوشت:

```
for (i = 1; i < 50; i++)  
s = s + i ;
```


مثال: برنامه‌ای بنویسید که مجموع اعداد 5- تا 10 را محاسبه و چاپ نماید.

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
    int i , A=0 ;
    for (i = -5 ; i <= 10 ; i ++ )
        A += i;
    printf ("%d", A );

    _getch();
    return 0;
}
```

۱-۲-۹-۱- حلقه بینهایت با استفاده از for

```
for ( ; ; )
{
}
```

به وسیله دستور break می‌توان از حلقه بینهایت خارج شد. بعداً با این دستور آشنا خواهید شد.

نکته: شاخص حلقه for را می‌توان در خود حلقه تعریف کرد. در این صورت متغیر فقط در داخل حلقه قابل استفاده است.

مثال:

```
for (int i=1;i<=10;i++)
```

۱-۲-۹-۲- حلقه for تو در تو^۱

حلقه for می‌تواند به صورت تو در تو به کار رود:

```
int i , j;
    for (i=1;i<=10;i++)
    {
        for (j=1;j<=10;j++)
```

1 Nsted For