برنامه‌سازی پیشرفته در محیط

Visual C++

مولف:

دکتر محمد علی ترکمانی

مهندس الهام ادراکی

سرشناسه	:	ترکمانی، محمدعلی، ۱۳۵۴ -
عنوان و نام پدیدآور	:	برنامه‌سازی پیشرفته در محیط ++/ مولفان محمدعلی ترکمانی، الهام ادارکی.Visual C
مشخصات نشر	:	مشهد: ارسطو، ۱۳۹۵.
مشخصات ظاهری	:	۲۴۹ص.: مصور، جدول، نمودار.
شابک	:	978-600-432-080-1
وضعیت فهرست نویسی	:	فیپا
موضوع	:	ویژوال سی++ مایکروسافت
موضوع	:	++ Microsoft visual C
موضوع	:	سی ++ (زبان برنامه‌نویسی کامپیوتر)
موضوع	:	C++ (Computer program language)
شناسه افزوده	:	ادراکی، الهام، ۱۳۵۶ -
رده بندی کنگره	:	QA۷۶/۷۳ /س۹۳ت۴ ۱۳۹۵
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 ۱٤
۱-۵-۱-دستور printf ۱٤
۱-۱-۵-۱-کنترل عمل چاپ در دستور printf ۱۵
۱-۱-۵-۱-تعیین کننده‌های فرمت ۱٦
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 ۲٤
۱-۲-۹-۱-حلقه بینهایت با استفا ده از for ۲۵
۱-۲-۹-۱-حلقه for تو در تو ۲۵
۱-۹-۳-دستور while ۲٦
۱-۹-۴-دستور do-while ۲۸
۱-۹-۵-دستور break ۲۸

۶-۹-۱ -دستور continue ۲۹

۷-۹-۱ -دستور if ۲۹

۸-۹-۱ -دستور switch ۳۰

۱۰-۱-آرایه‌ها ۳۱

۱-۱۰-۱ -مقداردهی اولیه به یک آرایه ۳۲

۱-۱-۱۰-۱-روش اول ۳۲

۲-۱-۱۰-۱-روش دوم ۳۲

۳-۱-۱۰-۱-آرایه‌های چند بعدی ۳۳

۴-۱-۱۰-۱-مقداردهي به آرايه‌هاي چند بعدي ۳۵

۱۱-۱-توابع ۳۶

۱-۱۱-۱ -محدوده تعریف متغیرها در زبان c++ ۳۸

۱-۱-۱۱-۱-متغیرهای محلی یا local ۳۸

۲-۱-۱۱-۱-متغیرسراسری یا global ۳۹

۱۲-۱-ارسال آرایه به یك تابع (به عنوان پارامتر) ۴۰

۱-۱۳- آرگومان پیش فرض ۴۱

۱۴-۱-پیش محاسبه گرها یا ماکروها ۴۲

۱-۱۴-۱ -دستور define ۴۳

۱۵-۱-رشته‌ها ۴۴

۱-۱۵-۱ -تعریف رشته ۴۴

۲-۱۵-۱ -توابع کار روی رشته‌ها ۴۴

۱-۲-۱۵-۱- Strcpy ۴۴

۲-۲-۱۵-۱-Strcmp ۴۵

۳-۲-۱۵-۱-strchr ۴۶

۴-۲-۱۵-۱- strstr ۴۶

۵-۲-۱-۱- strlen ۴۶

۶-۲-۱۵-۱- strcat ۴۷

۷-۲-۱۵-۱- gets ۴۷

۸-۲-۱۵-۱- puts ۴۷

۹-۲-۱۵-۱- دستور strlwr ۴۷

۱۰-۲-۱۵-۱- دستور strupr ۴۸

۱۱-۲-۱۵-۱- دستور tolower ٤٨
۱۲-۲-۱۵-۱- دستور toupper ٤٨
۳-۱۵-۱-توابع تبدیل رشته به عدد و بالعکس *٤٨*
۱-۳-۱۵-۱-تابع atoi ٤٨
۲-۳-۱۵-۱-تابع itoa ٤٩
۱۶-۱-تمرین‌ها ٥٠
۱۶-۱-سئوالات چهارگزینه‌ای ٥١
پاسخ ها *٥٢*

فصل دوم:اشاره‌گرها، تخصیص پویای حافظه و مراجع..... ۵۳

۱-۲-اشاره‌گر ها ٥٣
۲-۲-عملگر های NEW و DELETE ٥٨
۱-۲-۲-مقداردهی اولیه *٦١*
۲-۲-۲-اشغال دینامیکی آرایه *٦١*
۳-۲-۲-تابع مرتبط با عملگر new *٦٣*
۳-۲-مراجع ٦٤
۱-۳-۲-تبدیل دو ریفرنس متفاوت *٦٥*
۴-۲-اشاره‌گر به تابع ٦٦
۵-۲-انواع فراخوانی توابع در C++ ٦٨
۱-۵-۲- فراخوانی با مقدار *٦٩*
۲-۵-۲- فراخوانی با آدرس *٦٩*
۳-۵-۲- فراخوانی با ارجاع *٧٠*
۲-٦-تمرین‌ها ٧١
۲-٦-سئوالات چهارگزینه‌ای ٧٢
پاسخ‌ها: *٧٤*

فصل سوم:تعریف ساختار..... ۷۵

۱-۳-دستور STRUCT ٧٥
۱-۱-۳-تعریف استراکچر به صورت آرایه *٧٨*
۲-۱-۳-تعریف struct به صورت اشاره‌گر *٨٠*

۲-۳-UNION ۸۱
۳-۳-تمرین‌ها ۸۳
۳-۳-سئوالات چهارگزینه‌ای ۸٤
پاسخ‌ها: ۸۵

فصل چهارم:برنامه‌سازی شیءگرا..... ۸۷

۱-۴-تکنیك‌هاي برنامه نويسي..... ۸۷
۲-۴- خصوصیات برنامه‌نویسی شیءگرا ۸۸
۳-۴-انواع زبان‌هاي شيءگرا ازنظر نحوه استفاده از شیءگرایی ۹۳
۴-۴-انواع زبان‌هاي شيءگرا از نظر تعداد کلاس پدر ۹٤
۵-۴-مفهوم فضای نام و فضای نام STD..... ۹٤
۶-۴-دستورات ورودي – خروجي در C++ ۹٤
۷-۴-قواعد تعریف کلاس ۹۵
۱-۷-۴-کنترل نحوه دسترسي به اعضای کلاس ۹٦
۱-۱-۷-۴- تعریف اعضای کلاس به صورت عمومي (public) ۹٦
۲-۱-۷-۴- تعریف اعضای کلاس به صورت خصوصي (private) ۹۷
۳-۱-۷-۴- تعریف اعضای کلاس به صورت محافظت شده (protected)..... ۹۷
۸-۴-توابع عضو کلاس ۹۸
٤-۸-۱-توابع inline ۹۹
۹-۴-محدوده تعریف کلاس..... ۱۰۰
۱۰-۴-تعریف اشیاء ۱۰۰
۱۱-۴-سازنده یک کلاس ۱۰۵
۱-۱۱-۴-نحوه اجرای تابع سازنده در برنامه ۱۰۵
۱۲-۴-تابع تخریب‌گر یک کلاس ۱۰٦
۱۳-۴-مقداردهی اولیه در توابع سازنده ۱۰۹
۱۴-۴-ویژگی‌های توابع سازنده وتخریب‌گر ۱۱۱
۱۵-۴-تعریف چند تابع سازنده برای یک کلاس ۱۱۲
۱۶-۴- توابع عضو ثابت..... ۱۱٤
۱۷-۴- اعضای کلاس با ویژگی STATIC ۱۱٤
۱-۱۷-۴- اعضای داده‌ای static..... ۱۱٤
۲-۱۷-۴-توابع عضو استاتیک ۱۱٦
۱۸-۴-اشاره‌گر هايي به اشياء ۱۲۰
۱۹-۴-اشاره‌گر THIS ۱۲۱
۲۰-۴-تخصیص پویاي اشیا ۱۲۲

۲۱-۴-آرایه‌ای از اشیاء ۱۲٤
۱-۲۱-۴-مقداردهی اولیه به آرایه‌ای از اشیاء با استفاده از فرم ALTERNATIVE ۱۲٦
STUDENT OB[4] [2]; ۱۲۷
۲۲-۴-اشغال دینامیکی آرایه‌ای از اشیاء ۱۲۷
۲۳-۴-عبور اشیاء به توابع ۱۲۸
۱-۲۳-۴- اشیاء به عنوان آرگومان توابع ۱۲۸
٤-۲۳-۲- عبور آدرس شیء به تابع ۱۲۹
٤-۲۳-۳-ارسال اشیاء به صورت مرجع به توابع ۱۳۱
۲۴-۴-اشیاء به عنوان مقدار بازگشتی توابع ۱۳۳
۲۵-۴-دستورالعمل FRIEND ۱۳۷
۱-۲۵-۴-نکاتی در مورد friend ۱۳۸
۲۶-۴-تمرین‌ها ۱٤۳
۲۷-۴- سئوالات چهارگزینه‌ای ۱٤٦

فصل پنجم:وراثت ۱۵۱

۱-۵- تعریف وراثت ۱۵۱
۲-۵-کلاس‌های پایه و مشتق شده ۱۵۱
۳-۵-قواعد ارث‌بري و کنترل دسترسی به اعضای کلاس پایه ۱۵۲
۱-۳-۵-کلاس پایه با پیشوند public ۱۵۲
۲-۳-۵-کلاس پایه با پیشوند protected ۱۵۲
۳-۳-۵-کلاس پایه با پیشوند private ۱۵۳
۴-۵-نحوه نوشتن کلاس‌هاي پایه، مشتق شده و تعریف توابع کلاس پایه ۱۵۳
۵-۵-وراثت چندگانه ۱۵٤
۶-۵-مثال‌هایی از وراثت ۱۵٤
۷-۵-تمرین‌ها ۱٦٦
۸-۵- سئوالات چهارگزینه‌ای ۱٦۷
پاسخ‌ها: ۱٦۸

فصل ششم:تابع مجازی ۱۶۹

۱-۶-تابع مجازی ۱٦۹
۱-۱-۶-اشاره‌گر به نوع پایه و مشتق شده ۱۷۱
۲-۱-۶-کلاس‌های مجرد و توابع مجازی محض ۱۷٤

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

فصل هفتم:سربارگذاری عملگرها..... ۱۸۱

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

فصل هشتم:فایل‌ها..... ۲۰۱

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

۵-۶-۸-دستور ungetc ۲۰۵
۶-۶-۸-دستورات freadو fwrite ۲۰۶
۷-۸-کاراکتر EOF ۲۰۶
۸-۸-دستور FSEEK ۲۰۷
۹-۸-دستور FTELL ۲۰۷
۱۰-۸-دستورREWIND ۲۰۷
۱۱-۸-تابع FEOF ۲۰۸
۱۲-۸-فرمان REMOVE ۲۱۱
۱۳-۸-فرمانRENAME ۲۱۱
۱۴-۸-تمرین‌ها ۲۱۱
۱۵-۸-سئوالات چهارگزینه‌ای ۲۱۲
پاسخ ها ۲۱۲

فصل نهم:استثناها و خطاها..... ۲۱۵

۱-۹-مقدمه ۲۱۵
۲-۹-خطاها ۲۱۵
۳-۹-مدیریت استثناها ۲۱۶
۴-۹-بلاک‌هایTRY-CATCH ۲۱۶
۵-۹-تمرین‌ها ۲۲۲
۶-۹-سئوالات چهارگزینه ای ۲۲۴
پاسخ‌ها ۲۲۵

پیوست‌ها..... ۲۲۵

پیوست ۱: آشنایی با VISUAL STUDIO .NET ۲۲۵
پیوست۲: کتابخانه‌های مهم ۲۳۲
پیوست ۳: جدول کدهای اسکی ۲۳۳

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 گردیدو 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: برنامه های ارائه شده دراین کتاب با آخرین نسخه Visual studio.net سازگار هستند و به راحتی در کنسولVisual C++.NET اجرا میشوند.

۵-رائه نمونه سئوال 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 انشالله 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

تابستان۱۳۹۵

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[1] در آزمایشگاه بل ایجاد گردید. C++ شامل تمامی امکانات زبان C است وعلاوه بر آن دارای امکانات برنامه‌نویسی شیءگرا(OOP)[2] می‌باشد(شکل ۱-۱). بنابراین می‌توان یک برنامه C را با کامپایلر C++ کامپایل کرد .

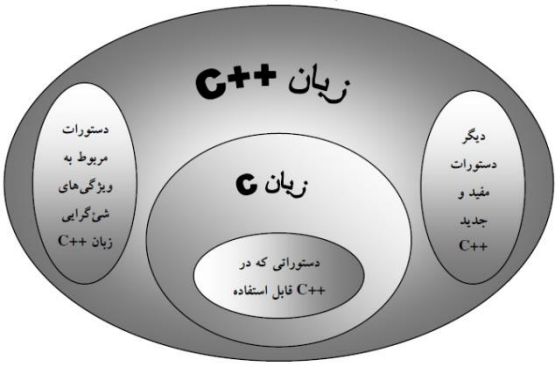
تاکنون کامپایلرهای مختلفی از C++ به بازار عرضه شده است که مهم‌ترین آنها عبارتند از Turbo C++ 3 محصول شرکت Boarland، Visual C++ 6 و Visual C++.Net که محصول شرکت ماکروسافت می‌باشند. Visual C++.Net جزیی از بسته نرم‌افزاری Visual Studio.Net است. در این مجموعه ابزارهای قدرتمند برنامه نویسی زیر وجود دارد:

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

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

---

1 Bjarne Srroustrup
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.

جدول۱- ۱: انواع متغیرها در زبان C++

نوع	بایت	محدوده مقادیر
int	4	2,147,483,647تا–2,147,483,648
unsigned int/ unsigned unsigned long /unsigned long int	4	4,294,967,295تا0
bool	1	trueیاfalse
char	1	127تا–128
unsigned char	1	255تا0
short/short int	2	32,767تا–32,768
unsigned short/ unsigned short int	2	65,535تا0
long/ long int	4	2,147,483,647تا–2,147,483,648
long long	8	تا–9,223,372,036,854,775,808 9,223,372,036,854,775,807 معادل با یک عدد صحیح ۶۴ بیتی است
float	4	3.4E +/- 38 (7رقم)
double / long double	8	1.7E +/- 308 (15رقم)
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. درC++ 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[1] 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-دستورات ورودی- خروجی

۱-۵-۱-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[1] 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 بعدا 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[2] 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.

۱Format Specifier

۲header 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	\ n (new line)	به ابتدای خط بعدی می‌رود .
۲	\ t (tab)	8 ستون به جلو می‌رود.
۳	\"	برا ی چاپ " به کار می‌رود .
۴	\\	برا ی چاپ \ به کار می‌رود.
۵	\b (back space)	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 درکتیشن دوتایی قرار می‌دهیم، اما اگر بخواهیم یک متغیر را چاپ کنیم باید از درصد متناسب با نوع آن متغیر استفاده کنیم.

مثال:

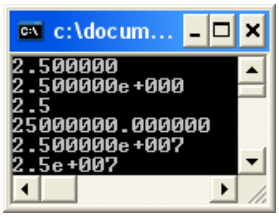
```
int A ;
A=2;
printf (“A” ) ;          // حرف A را چا پ می کند.
printf ( “%d”  , A);     //2 را چا پ می کند.
```

مثال: اگر B=2.5*10000000 و A=2.5 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 معمو لی، هر کدام که ساده تر باشد نوشته می‌شود
p	Pointer to void	آدرس آرگومان را به صورت مبنای ۱۶ چاپ می‌کند.
s	string	رشته‌ها

۲-۵-۱-دستور scanf

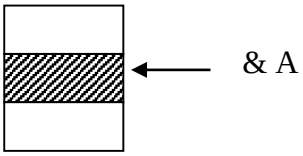
این دستور برای دریافت اطلاعات از ورودی به کار می‌رود. شکل کلی این دستور به صورت زیر است:

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

تابع scanf یک عدد را بر می‌گرداند که این عدد تعداد آیتم‌هایی 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[1] به خانه‌ای از حافظه است که A در آن قراردارد (شکل ۱-۲).



شکل ۱- ۲: اشاره‌گر به آدرس متغیر A

مثال:

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

نکته: وارد کردن داده‌ها دقیقا باید مانند آنچه در دستور scanf_s_s است، انجام شود.

مثال:

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

در اینجا باید داده‌ها را به صورت زیر واردکنیم:

```
5 , 7
```

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

---

[1] pointer

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

۱-۶-۱- عملگرهای محاسباتی

عملگرهای محاسباتی در جدول۱-۴ نمایش داده شده‌اند.

جدول ۱- ۴: عملگرهای محاسباتی

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

۱-۶-۲-عملگرهای رابطه‌ای

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

جدول ۱- ۵: عملگرهای رابطه‌ای

عملگر	رابطه
>	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 ۱-۶ عملگرهای منطقی به ترتیب اولویت را نمایش می‌دهد.

جدول ۱- ۶: 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[1]

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

جدول ۱- ۷: 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"
```

1 Bitwise operators

```
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 ۸-۱ 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 B و A از نوع اعشاری باشد، A/B از نوع اعشاری است.

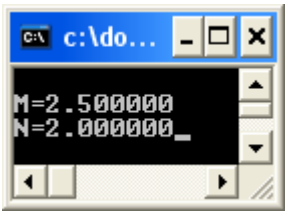
مثال:

```
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++

۱-۹-۱- پاک کردن صفحه کنسول

در Viusal 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 می توان 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 نمایید.

```
#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 استفا ده از for

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

به وسیله دستور break می‌توان از حلقه بینهایت خارج شد. بعدا 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++)
```

۲-۲-۹-۱-حلقه for تو در تو[1]

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 Nested For

```
        {
                printf("\n(%d,%d)",i,j);
         }
    }
```

۳-۹-۱-دستور while

دستور While ابتدا شرط حلقه (condition) را بررسی می‌کند. اگر این شرط صحیح بود، کارهای درون بدنه را انجام می‌دهد و دوباره به سراغ شرط می‌آید و آنرا بررسی می‌کند. تا وقتی که شرط درست باشد، بدنه اجرا خواهد شد.

شکل کلی دستور:

```
while( condition )
{
   body
}
```

نکته: حلقه for یک حلقه معین است، اما while یک حلقه نا معین است و مشخص نیست که چند بار دستوراتش اجرا می‌شود. در دستور whileمشخص نیست که چه وقت شرط جلو این دستور نقض می‌شود.

مثال: برنامه جمع اعداد ۰ تا ۱۰(که قبلا"باز for نوشتیم) را با استفاده از while بنویسید؟

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
int  i=0 , A =0 ;
while(i<=10)
{
A+=i; // A=A+i
i++;
}
printf( "\n sum is %d ",A) ;
_getch();
return 0;
}
```

مثال: برنامه‌ای بنویسید که یک کلید را بخواند و آن کلید را برای ما چاپ کند (بنوسد چه کلیدی فشار داده‌ایم) فقط اگر این کلید q یا Q بود، برنامه را تمام کند. درغیر این صورت از شما بپرسد کلید بعدی چیست.

چون می‌خواهیم یک کاری را به تعداد نا معین انجام دهیم به یک حلقه While احتیاج داریم. در بدنه Whileیک دستور getch_ می‌خواهیم که کلید را بخواند و سپس چاپ کند و برگردد شرط را چک کند.

دستورC='b' به این علت نوشته شده است که مطمئن شویم مقدار اولیه درون C مقدار Q یا q نیست.

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
     char c= 'b'  ;
       while ( c != 'q' &&  c != 'Q' )
     {
     c= _getch ( );
     printf ( "\n % c" , c ) ;
    }
   _getch();
return 0;
}
```

به جای دو دستور موجود در بدنه While می‌توان C= _getche ( ); را قرار داد این دستور خودش کلیدی را که خوانده می‌نویسید، پس احتیاجی به printf نیست. به جای دستورات بالا می‌توان نوشت:

printf ("\n % c" , (c = _getch ( ))) ;

در برنامه فوق اگر کلیدی که زده شده q باشد، ابتدا آنرا چاپ می‌کند و سپس برنامه تمام می‌شود. اگر getch_ را جزء شرط حلقه قرار دهیم ابتدا کلید زده شده را کنترل می‌کند، اگر q نبود آنرا چاپ می‌کند و اگر q بود دیگر آنرا چاپ نمی‌کند و خارج می‌شود.

```
 int main(int argc, _TCHAR* argv[])
{
     char  c ;
      while  ((  c = _getch ( )) != 'q')
      printf ( "\n %c " ,  c ) ;
     _getch();
return 0;
}
```

نکته: گفتیم که اگر یک عبارت منطقی حاصلش یک باشد، آن عبارت درست است. بنابراین حلقه زیر شرطش همیشه درست است و مانند for (; ;) یک حلقه بی نهایت است وتا بی‌شمار ادامه خواهد داشت.

```
while (1)
       {
       }
```

۴-۹-۱-دستور do-while

```
do
{
 دستورات
} while (condition);
```

تفاوت دستور do-while این است که بدنه این دستور برخلاف دستور while حداقل یک بار اجرا می‌شود.

مثال: برنامه زیر اعداد ۱ تا ۱۰ را روی صفحه نمایش می‌دهد.

```
#include "stdafx.h"
#include "conio.h"
int main(int argc, _TCHAR* argv[])
{
 int  i=0;
 do  {
      i=i+1;
      printf("%d",i);
      }while(i<10);
      _getch();
      return 0;
}
```

۵-۹-۱-دستور break

دستورbreak حلقه می‌شکند و کنترل اجرای برنامه را به خط بعد از حلقه منتقل می‌کند.

```
while  یا  for ()
{
  دستورات
  Break;
  دستورات
}
```

۶-۹-۱-دستور continue

به ابتدای حلقه بر می‌گردد.

```
while (      )
{
_______
_______
_______
if( )  continue ;
_______
_______
_______
}
```

تذکر: دستورات break و continue فقط مربوط به حلقه‌ها هستند ونمی‌توان با استفاده از آنها از داخل بلاک‌های دیگر نظیر if خارج شد.

نکته: می‌توانیم شرط خروج از حلقه را با if چک کنیم.

مثال:

```
while  (k !='q')
{
______
_____
if (k= ='Q')  break;
_____
_____
}
```

۷-۹-۱-دستور if

شکل کلی دستور به صورت زیر است:

```
if   (  condition  )
{
Statement  1 ;
}
else  {
 statement  2 ;
}
```

اگر شرط جلو if درست باشد، دستورات موجود در بلاک if اجرا می‌شود و در غیر این صورت دستورات بعد از else اجرا خواهد شد.

نکته: دستور if را بدون else هم می‌توان به کار برد. در این حالت اگر شرط غلط باشد کنترل اجرای برنامه به خط بعد از if منتقل می‌شود.

۸-۹-۱-دستور switch

این دستور برای جلوگیری از تکرار دستور if مفید است.

شکل کلی دستور:

```
switch (expression)
{
case 1:
…
break ;
case 2:
…
break;
case3:
…
break;
case n:
…
break;
default:
…
}
```

اگر دستور break نوشته نشود، بقیه شرط‌ها هم چک می‌شود و زمان بیشتری تلف می‌گردد.

نکته: case 'a' برای چک کردن کاراکتر و case B برای چک کردن متغیر B (int) و case 1 برای چک کردن عدد ۱ به کار می‌رود.

تذکر: متغیرهای از نوع float را نمی‌توان به وسیله دستور Switch - case چک کرد.

مثال: برنامه زیر مقدار متغیر a را بررسی می‌کند و با توجه به مقدار آن، یکی از اعداد ۱ تا ۳ چاپ می‌شود. اگر مقدار a مخالف اعداد ۱ تا ۳ باشد، رشته default چاپ می‌شود.

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