بسم الله الرحمن الرحیم

برنامه‌سازی ۲

رشتهٔ کامپیوتر

گروه تحصیلی کامپیوتر

زمینهٔ خدمات

شاخهٔ آموزش فنی وحرفه‌ای

عنوان و نام پدیدآور	:	برنامه‌سازی ۲ : رشتهٔ کامپیوتر گروه تحصیلی کامپیوتر زمینه خدمات شاخه آموزش فنی و حرفه‌ای.
مشخصات نشر	:	تهران : شرکت چاپ و نشر کتاب‌های درسی ایران، ۱۳۹۴
مشخصات ظاهری	:	۲۱۹ ص. : مصور.
شابک	:	۹۷۸-۹۶۴-۰۵-۲۴۱۳-۸
وضعیت فهرست‌نویسی	:	فیپای مختصر
شناسه افزوده	:	کربلایی، مجید
شناسه افزوده	:	سازمان پژوهش و برنامه‌ریزی آموزشی. دفتر تألیف کتاب‌های درسی فنی وحرفه‌ای و کاردانش
شماره کتاب‌شناسی ملی	:	۳۵۹۳۹۲۱

۱۳۹۴

همکاران محترم و دانش‌آموزان عزیز :

پیشنهادات و نظرات خود را دربارهٔ محتوای این کتاب بـه نشانی تهران ـ صندوق پستی شمارهٔ ۴۸۷۴/۱۵ دفتر تألیف‌کتاب‌های درسی فنی و حرفه‌ای و کاردانش، ارسال فرمایند.

پیام‌نگار (ایمیل) **info@tvoccd.sch.ir**

وب‌گاه (وب‌سایت) **www.tvoccd.sch.ir**

وزارت آموزش و پرورش
سازمان پژوهش و برنامه‌ریزی آموزشی

برنامه‌ریزی محتوا و نظارت بر تألیف : **دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش**

نام کتاب : **برنامه‌سازی ۲ ـ ۴۵۱**

مؤلف : **مجید کربلایی**

اعضای کمیسیون تخصصی : **محمدرضا یمقانی، بتول عطاران، شروین الوندی، مهیار پازوکی، شهناز علیزاده، نیلوفربزرگ‌نیا، سارو آواکیانس، محمدرضا شکرریز، مهناز کارکن، افشین اکبری، سید سعیدرضا سعادت‌یزدی، پردیس پیرایش و زهرا عسگری رکن‌آبادی**

ویراستار علمی : **مهناز کارکن، زهرا عسگری رکن آبادی و حسین صفوی**

ویراستار ادبی : **پردیس پیرایش، صدیقه رسولی قلاتی**

آماده‌سازی و نظارت بر چاپ و توزیع : **ادارهٔ کل نظارت بر نشر و توزیع مواد آموزشی**

تهران : خیابان ایرانشهر شمالی ـ ساختمان شمارهٔ ۴ آموزش و پرورش (شهید موسوی)

تلفن : ۹ـ ۸۸۸۳۱۱۶۱، دورنگار : ۸۸۳۰۹۲۶۶، کد پستی :۱۵۸۴۷۴۷۳۵۹،

وب سایت : www.chap.sch.ir

مدیر امور فنی و چاپ : **لیدا نیک‌روش**

طراح جلد : **مریم کیوان**

صفحه‌آرا : **زهره بهشتی‌شیرازی**

مصحح : **سیما لطفی، نرگس معاونیان**

امور آماده‌سازی خبر : **زینت بهشتی‌شیرازی**

امور فنی رایانه‌ای : **ناهید خیام‌باشی، فاطمه رئیسیان‌فیروزآباد**

ناشر : **شرکت چاپ و نشر کتاب‌های درسی ایران : تهران ـ کیلومتر ۱۷ جادّهٔ مخصوص کرج ـ خیابان ۶۱ (دارو پخش)**

تلفن : ۵ ـ ۴۴۹۸۵۱۶۱، دورنگار : ۴۴۹۸۵۱۶۰، صندوق پستی : ۱۳۹ـ ۳۷۵۱۵

چاپخانه : **شرکت چاپ و نشر کتاب‌های درسی ایران «سهامی خاص»**

سال انتشار و نوبت چاپ : **چاپ اول ۱۳۹۴**

شابک ۸-۲۴۱۳-۰۵-۹۶۴-۹۷۸ ISBN 978-964-05-2413-8

از شماست که مردان و زنان بزرگ تربیت می‌شود. شماها در تحصیل کوشش کنید که برای فضایل اخلاقی، فضایل اعمالی مجهز شوید. شما برای آتیه مملکت ما جوانان نیرومند تربیت کنید. دامان شما یک مدرسه‌ای است که در آن جوانان بزرگ تربیت بشود. شما فضایل تحصیل کنید تا کودکان شما در دامان شما به فضیلت برسند.

امام خمینی «ره»

فهرست

اول یادآوری و تکمیل مطالب

دوم آرایه

سوم داده شمارشی، کلاس و متد

چهارم کار با متدها و کلاس‌های آماده

پنجم ایجاد برنامه‌های ویندوزی با استفاده از ویژوال استودیو

ششم ایجاد برنامه‌های ویندوزی با واکنش نسبت به رویدادها

مقدمه

زبان برنامه‌نویسی C# یکی از زبان‌های برنامه‌نویسی متداول و محبوب در بین برنامه‌نویسان می‌باشد. در کتاب برنامه‌سازی ۱ با برخی از جنبه‌های مقدماتی این زبان آشنا شدید. در این کتاب با جنبه‌های پیشرفته این زبان آشنا می‌شوید. یادگیری مفاهیم این کتاب شما را قادر خواهد ساخت تا برنامه‌های کامل‌تر و پیچیده‌تر نوشته و امکانات این زبان را به نحو مناسب‌تری به‌کار ببرید.

فصل اول کتاب به یادآوری و مرور مطالب کتاب برنامه‌سازی ۱ اختصاص دارد. البته در این فصل علاوه بر مرور مطالب کتاب قبلی، چند مطلب جدید نیز می‌آموزید. فصل ۲ به مفهوم مهم آرایه در زبان C# می‌پردازد. مفهوم آرایه در تمامی زبان‌های برنامه‌سازی کاربرد بسیار مهمی دارد. فصل ۳ به کلاس و متد اختصاص دارد که شروع شیءگرایی می‌باشد و شما را با دنیای جالب برنامه‌نویسی شیءگرا آشنا می‌سازد. البته مفاهیم پیشرفته‌تر شیءگرایی را در کتاب برنامه‌سازی ۳ فرا خواهید گرفت. در فصل ۴ یاد می‌گیرید که چگونه با متدها و کلاس‌های آماده زبان C# کار کنید. در فصل ۵ اصول ساخت یک برنامه ویندوزی با زبان C# را فرا خواهید گرفت و در نهایت در فصل ۶، با نحوه کار کنترل‌های مختلف آشنا خواهید شد.

در همه فصل‌ها سعی شده با آوردن فعالیت‌های کارگاهی، مطلب ارائه شده در قالب پروژه‌های مختلف به صورت کاملاً عملی آموزش داده شود. لذا از هنرجویان انتظار می‌رود با انجام تمرینات کار در کارگاه و خودآزمایی‌ها و فعالیت‌های آورده شده، زمینه یادگیری مطالب را فراهم نمایند.

در پایان از تمام دوستانی که در تألیف این کتاب یاری‌رسان اینجانب بوده‌اند، از جمله سرکار خانم مهناز کارکن، سرکار خانم زهرا عسگری رکن‌آبادی، جناب آقای حسین صفوی تشکر و قدردانی می‌نمایم. ایده‌ها و پیشنهادات شما باعث بهبود کیفیت کتاب خواهد شد. لذا مواردی که به نظر شما می‌رسد را به دیده منت می‌نهیم و ارج می‌گذاریم.

مجید کربلایی

فصل اول

یادآوری و تکمیل مطالب

در کتاب برنامه سازی۱، با مفهوم برنامه، برنامه نویسی و سطوح مختلف زبان‌های برنامه نویسی آشنا شدید، سپس در فصل‌های بعدی کتاب، دستورات مقدماتی و پایه‌ای زبان برنامه نویسی C# را آموختید. همچنین توانستید برنامه‌هایی به زبان C# بنویسید که در محیط کنسول سطر فرمان اجرا شوند. اکنون در این فصل، به یادآوری مطالب آموخته شده قبلی می‌پردازیم و با نوشتن چند برنامه، دستورات پایه‌ای خوانده شده را بررسی و مرور می‌کنیم. در بعضی از برنامه‌ها نیز با ارایه دستورات جدید اطلاعات خود را تکمیل می‌کنید.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ از محیط SharpDevelop برای نوشتن و ترجمه برنامه‌های کنسولی استفاده کند.

۲ــ کاربرد متد TryParse را بیان نماید و از آن استفاده کند.

۳ــ با استفاده از عملگر سه تایی، دستور if را بازنویسی نماید.

۴ــ کلمه کلیدی const را برای تعریف ثابت‌های برنامه به کار بندد.

۵ــ دستورات تکرار متداخل(تودرتو) را در برنامه استفاده نماید.

۶ــ عملکرد دنباله معنی‌دار را بیان نماید و آن را به کار بندد.

۱ــ۱ــ یادآوری

برای مرور آنچه درباره برنامه نویسی با زبان C# آموختیم به یادآوری مباحث مربوط به محیط نگارش و نحوه اجرای برنامه در این محیط می‌پردازیم. یکی از راه‌های ایجاد برنامه به زبان C# نوشتن برنامه در قالب فایل متنی ساده و با ابزاری مانند Notepad است. اگر بخواهیم برنامه ۱ــ۱ را اجرا کنیم، باید آن را به صورت فایل متنی نوشته و با نام Test.cs ذخیره کنیم تا متن برنامه به زبان سی شارپ داشته باشیم.

مثال ۱ــ۱ : برنامه‌ای بنویسید که از لیست زبان‌های نمایش داده شده، زبان سطح میانی را تشخیص دهد و انتخاب نماید.

```
using System;
class Program
{
    static void Main ()
    {
        Console. WriteLine (" which of the following languages are in th middle
languag " );
        Console. WriteLine (" a. C");
        Console. WriteLine (" b. C++");
        Console. WriteLine (" c. C#");
        Console. WriteLine (" d. JAVA");
        Console. Write ("Select Answer (a Or b Or c Or d)? ");
        string choice = Console. ReadLine ( );

        if (choice = = "a")
            Console. WriteLine ("Bravo! Your Answer is correct");
        else if (choice = = "b" | | (choice = = "c" | | (choice = = "d")))
            Console. WriteLine ("Your Answer is incorrect");
        else
        Console. WriteLine ("ERROR");
    }
}
```

برنامه ۱ــ۱ــ تشخیص زبان سطح میانی

اکنون نوبت به ترجمه و تبدیل آن به فایل اجرایی رسیده است. برای ترجمه این برنامه ابتدا وارد

سطر فرمان شده، سپس وارد پوشه‌ای که فایل برنامه ما در آن قرار دارد می‌شویم و فرمان زیر را برای ترجمه برنامه وارد می‌کنیم :

```
CSC  Test.cs
```

در صورتی که برنامه اشکال داشته باشد لیست خطاها ظاهر می‌شود و باید خطاهای احتمالی را برطرف کنیم و در صورت موفقیت فایل اجرایی ساخته می‌شود. برای اجرای برنامه نیز کافی است نام فایل تولید شده را تایپ کرده و کلید Enter را بزنیم.

```
Test.EXE
```

این روش تولید فایل اجرایی برنامه، که از یک ویرایشگر برای نوشتن برنامه و از فایل CSC.EXE برای ترجمه برنامه، استفاده می‌کنند، بسیار وقت گیر است و عیب یابی برنامه‌های بزرگ نیز دشوار است.

۲ــ۱ــ استفاده از محیط توسعه برنامه متمرکز

در برنامه‌های کنسولی ابتدا با استفاده از یک ویرایشگر[۱] متن مانند Notepad، برنامه را نوشته و سپس با پسوند cs. ذخیره کرده و با مترجم C# یعنی برنامه CSC.EXE کامپایل و اجرا می‌کردیم.

روش دیگری که می‌توان برای تولید فایل اجرایی برنامه استفاده کرد، استفاده از یک محیط برنامه‌نویسی متمرکز (IDE[۲]) است که برای نوشتن وعیب‌یابی راحت‌تر برنامه‌ها، طراحی شده است. چندین IDE به وسیلهٔ شرکت‌ها و ارگان‌های مختلف طراحی و به بازار عرضه شده است. بعضی از آنها رایگان و به اصطلاح متن باز[۳] و بعضی دیگر تجاری[۴] هستند. برای انتخاب IDE باید با توجه به سیستم‌عامل کامپیوتر و بودجه مورد نظر اقدام کرد. مثلاً در سیستم‌عامل ویندوز می‌توان از محیط برنامه‌نویسی رایگان و متن باز SharpDevelop یا به اختصار develop#[۵] استفاده نمود (شکل ۱ــ۱).

ویژگی این IDE حجم کمِ آن نسبت به IDE مایکروسافت می‌باشد. برای استفاده از این IDE ابتدا باید نرم افزار Net Framework 4.5 را نصب کنید، سپس نرم افزار متن باز develop# نصب شود.

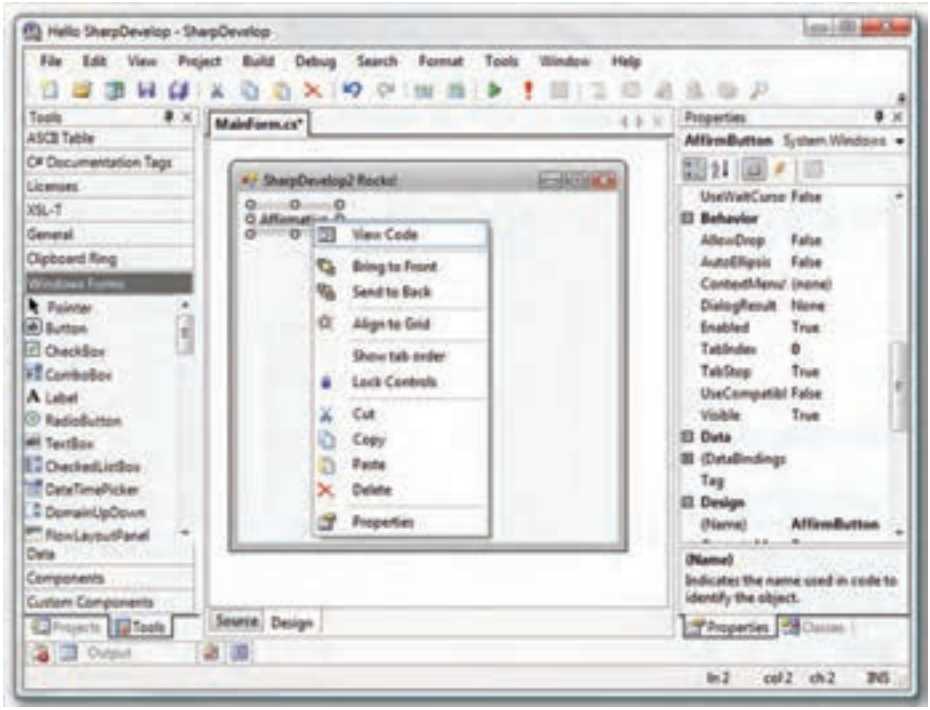
۱ــ Editor

۲ــ Integrated Development Environment

۳ــ Open Source

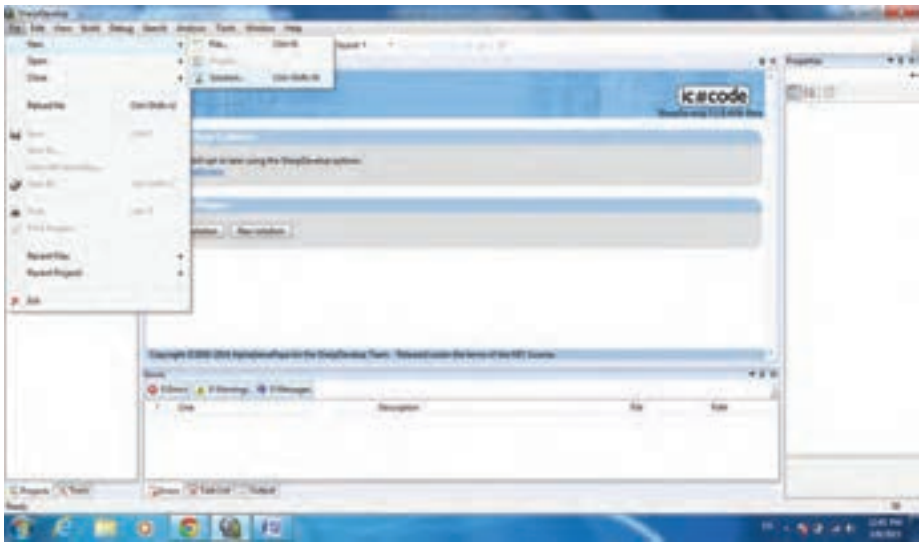
۴ــ Commercial

۵ــ # develop (short for SharpDevelop)



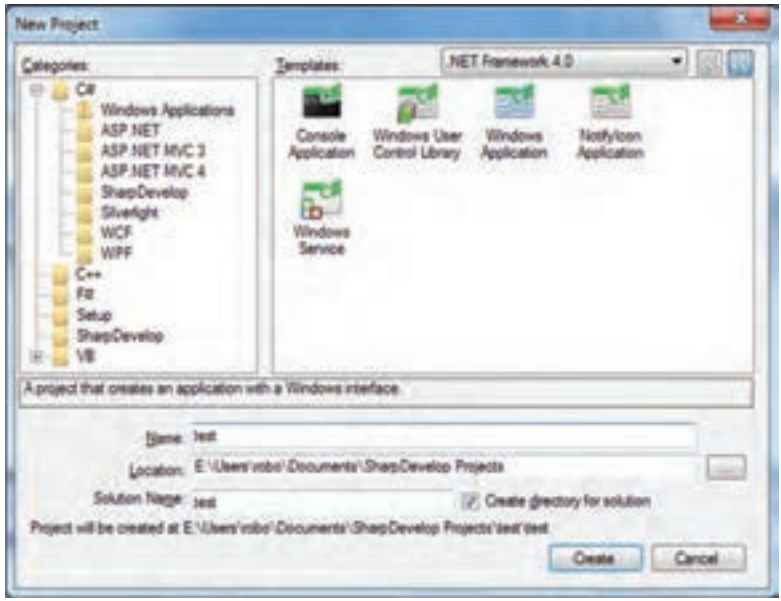
شکل ۱ــ۱ــ محیط برنامه نویسی رایگان و متن باز SharpDevelop

پس از نصب برنامه با کلیک روی آیکون SharpDevelop 5.1 در منوی start برنامه اجرا شده از منوی file گزینه solution را از زیر منوی new انتخاب می‌کنیم.



شکل ۲ــ۱ــ انتخاب گزینه solution از زیرمنوی new

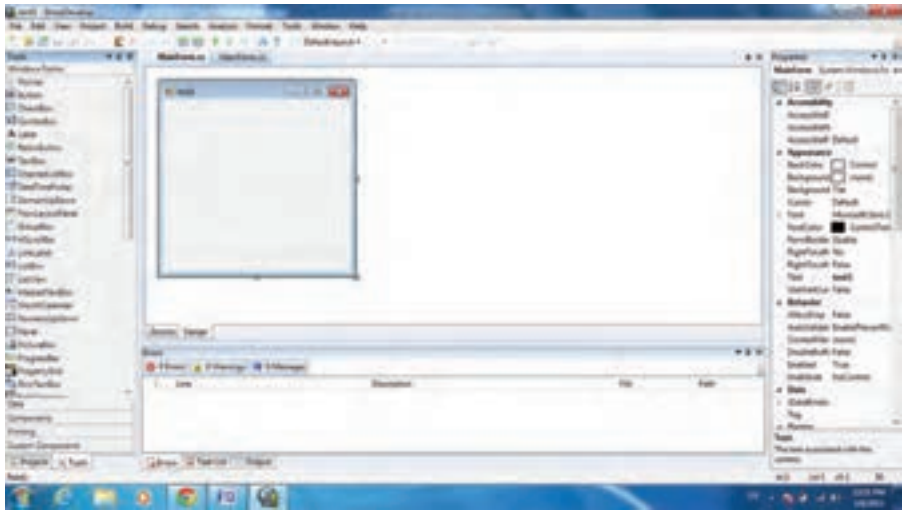
سپس گزینه windows application را انتخاب کرده در قسمت name نامی‌را برای پروژه انتخاب می‌کنیم.



شکل ۳ــ۱ــ تعیین نام پروژه

روی گزینه create کلیک می‌کنیم.

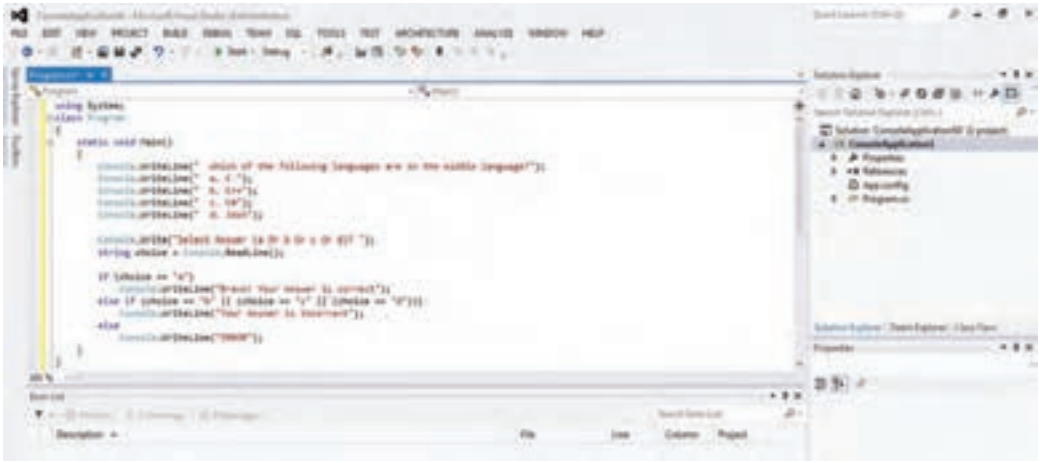
با انتخاب برگه design و برگه tools می‌توانیم به فرم اصلی و ابزارهای برنامه نویسی دسترسی داشته باشیم.



شکل ۴ــ۱ــ فرم اصلی و ابزارهای برنامه نویسی SharpDevelop

در این IDE فرم ابتدایی MainForm نام دارد.

تاکنون از محیط برنامه نویسی ویژوال استودیو اکسپرس[1] نسخه ۲۰۱۲ استفاده می‌کردید که مایکروسافت آن را به طور رایگان از طریق سایت خود عرضه می‌کند، تا برنامه نویسان برای تولید برنامه و ارزیابی این محصول، از آن استفاده کنند و نهایتاً نسخه اصلی ویژوال استودیو را خریداری[2] نمایند.



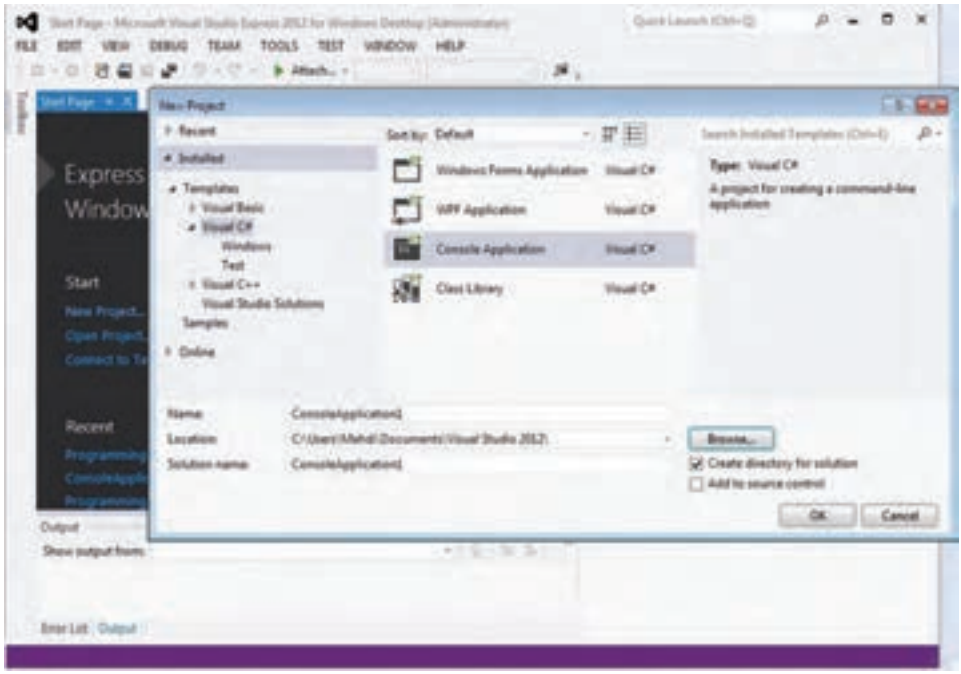
شکل ۵ــ۱ــ محیط Visual Studio

برای نوشتن برنامه‌های این کتاب نیز از وِیژوال استودیو اکسپرس ۲۰۱۲ یا به اختصار VS[3]، استفاده می‌کنیم. در اینجا طریقه نوشتن برنامه و اجرای آن در VS را به اختصار مرور می‌کنیم. برای نوشتن برنامه به زبان C# مانند برنامه ۱ــ۱، ابتدا VS را اجرا کرده، و در پنجره Start page گزینه New Project، گزینه Visual C# و سپس Console Application را انتخاب کنید.

۱ــ Visual Studio Express

۲ــ قیمت ویژوال استودیو ۲۰۱۳ حداقل حدود ۵۰۰ دلار می‌باشد (برای اطلاع از قیمت دقیق و جزئیات به سایت مایکروسافت مراجعه کنید).

۳ــ Visual Studio



شکل ۶ــ۱ــ پنجره پروژه جدید

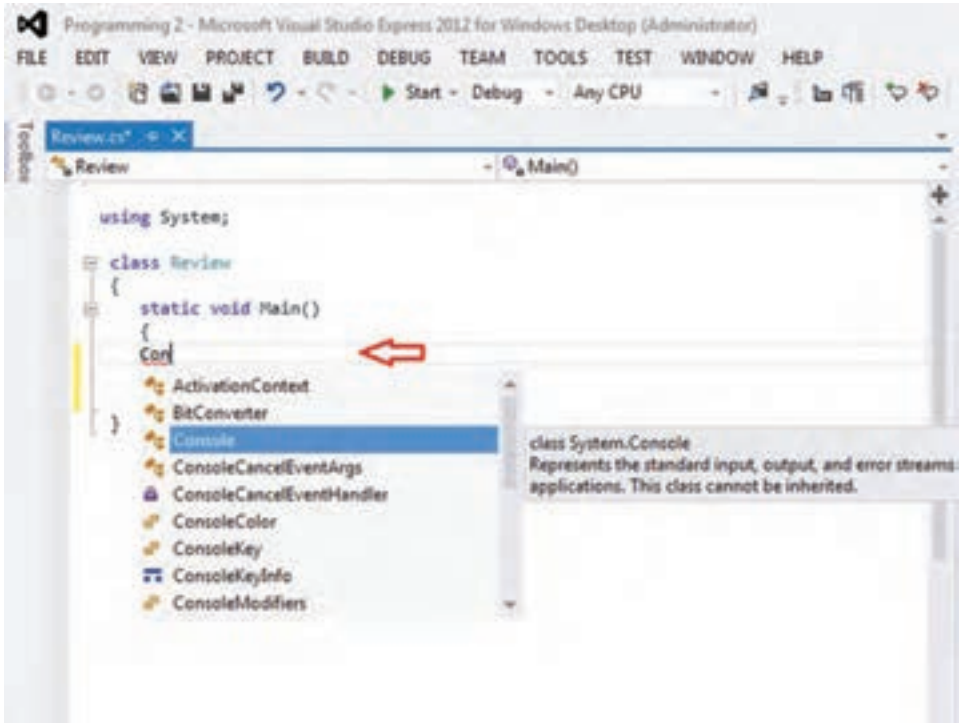
پس از مشخص کردن نام پروژه و مسیری که در آن پروژه ذخیره می‌شود، بر روی OK کلیک می کنیم و در پنجره ویرایشگر برنامه، دستورات را در داخل()Main می‌نویسیم.

محیط VS بسیار قدرتمند و دارای ابزارهای مختلفی برای نوشتن، اجرا و عیب‌یابی برنامه است. یکی از امکانات ارزشمند در هنگام نوشتن برنامه، IntelliSense[1] است که در هنگام تایپ دستورات برنامه، با نوشتن چند حرف از نام دستور (نام کلاس، متد،...)، منو یا لیستی ظاهر می‌شود که در آن نام متدها و دستورات مرتبط نشان داده شده‌اند. در این لیست می‌توانید با کلیدهای فلش بالا و پایین، حرکت کنید و سپس کلمه مورد نظر خود را برگزینید. با زدن کلید Tab یا کلید فاصله[2]، دستور مورد نظر، در برنامه به طور خودکار قرار می‌گیرد و لازم نیست بقیه نام دستور را تایپ کنید. این امکان علاوه‌بر اینکه سبب می‌شود سرعت تایپ برنامه افزایش یابد، باعث می‌شود دستورات با املای صحیح و بدون خطا نوشته شود.

۱ــ Intelligent Sense

۲ــ Space Bar

همان‌طور که در شکل ۷ ــ ۱ مشاهده می‌کنید، برنامه‌نویس قصد تایپ نام کلاس Console را دارد که به محض تایپ حرف C، لیست Intelli Sense به طور خودکار[1] باز می‌شود و نام‌هایی را نشان می‌دهد که با حرف C شروع می‌شوند و اگر برنامه نویس مشغول تایپ دو حرف دیگر شود، در لیست Intellisense به نام کامل Console می‌رسد و می‌تواند بقیه کلمه را تایپ نکند و تنها با زدن کلید Tab یا فاصله، کلمه Console را در برنامه بگنجاند.



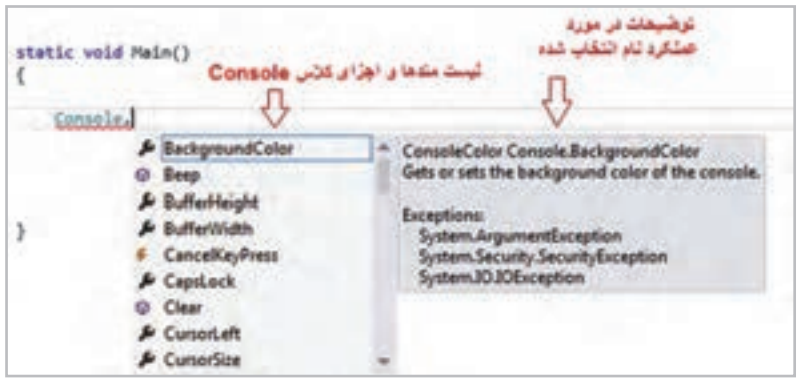
شکل ۷ ــ ۱ــ پنجره کدنویسی

لیست کلماتی که به وسیلهٔ IntelliSense نشان داده می‌شود، مرتبط با کلمه‌ای است که برنامه‌نویس در حال تایپ آن است. مثلاً هنگامی که برنامه نویس علامت نقطه را پس از کلاس Console تایپ می‌کند، لیستی از متدهای این کلاس ظاهر می‌شود.

۱ــ ممکن است این امکان غیرفعال شده باشد که در این‌صورت به توضیح ارائه شده در نکته شماره ۲ مراجعه کنید.

نکته

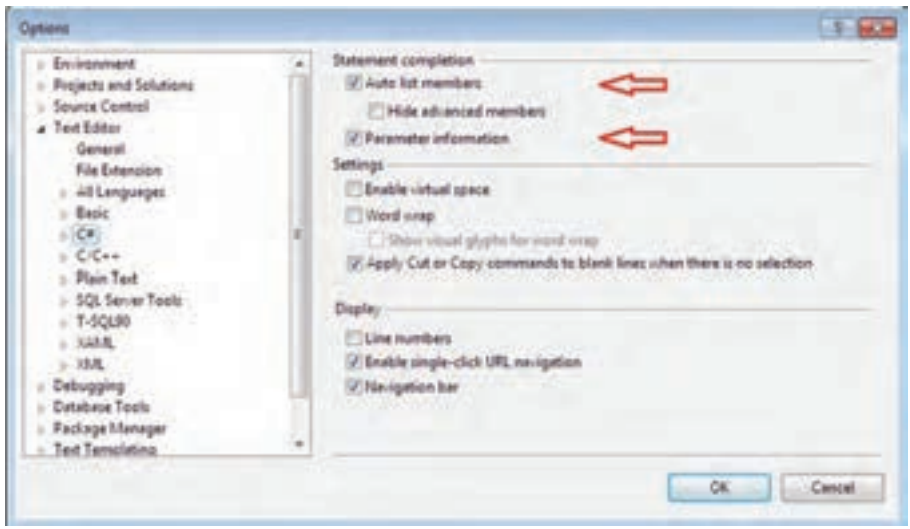
لیست مربوط به IntelliSense را در داخل متن برنامه، می‌توانید با زدن کلید ترکیبی CTRL+SPACE ظاهر کنید.



شکل ۸ــ۱ــ منوی IntelliSense

نکته

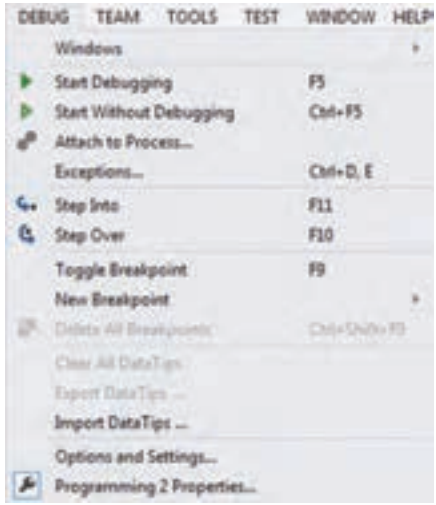
اگر امکان IntelliSense فعال نیست و از آن نمی‌توانید استفاده کنید، باید از طریق منوی Tools گزینه Options وارد قسمت Text Editor شوید و سپس در بخش C#، گزینه Auto list members و Parameter information را فعال کنید.



شکل ۹ــ۱ــ فعال یا غیر فعال کردن امکان IntelliSense

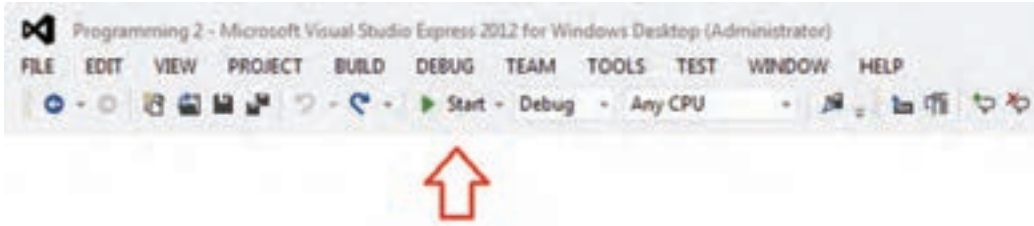
پس از پایان تایپ دستورات، برنامه را ترجمه و اجرا کنید. بدین منظور منوی Debug را باز کرده، و از آن می‌توانید گزینه‌های Start Debugging و یا Start Without Debugging را انتخاب و یا از کلیدهای میانبر آنها CTRL+F5 و یا F5 برای اجرای برنامه استفاده کنید.

سؤال: تفاوت این دو گزینه در اجرای برنامه چیست؟



شکل ۱۰ــ۱ــ منوی Debug برای اجرای برنامه

همچنین می‌توانید در نوار ابزار بالای صفحه، روی کلید Start کلیک کنید.



شکل ۱۱ــ۱ــ گزینه Start در نوار ابزار محیط VS

در دنباله این فصل در قسمت کار درکارگاه عملیات ذکر شده را در محیط VS انجام خواهید داد.

۳ــ۱ــ یادآوری دستورات زبان برنامه نویسی C#

۱ــ۳ــ۱ــ دستورات ورودی و خروجی

همان‌طور که به خاطر دارید برای دریافت داده‌ها از متد ReadLine() و همچنین برای نمایش اطلاعات بر روی صفحه نمایش از متد WriteLine() استفاده می شود.

سؤال: متد دیگری برای نمایش اطلاعات روی صفحه نمایش، متد Write () است این متد با متد WriteLine() چه تفاوتی دارد؟

۲ــ۳ــ۱ــ تبدیل نوع داده : به خاطر دارید که برای نگهداری داده‌ها در برنامه، از متغیرها استفاده می‌شود. هر متغیر مانند ظرفی است که می‌تواند مقداری را در خود نگهداری کند. اندازه و گنجایش هر متغیر توسط نوع داده معین می‌شود. انواع مختلفی از داده‌ها در زبان C# وجود دارند که برای اعداد صحیح، اعداد اعشاری، حروف و کلمات مورد استفاده قرار می‌گیرند. مانند : long ،int ،double ،char ،string.

قبل از اینکه بخواهیم از یک متغیر در برنامه استفاده کنیم، بایستی نام متغیر، گنجایش آن و نوع داده‌ای را که می‌تواند نگهداری کند مشخص کنیم.

کار در کارگاه ۱

مثال ۲ــ۱ : برنامه‌ای بنویسید که سن شما را بر حسب سال نشان دهد.

با توجه به اینکه سن عدد کوچکی است می‌توانیم آن را از نوع byte تعریف کنیم. سپس با پیامی مقدار این متغیر را چاپ می کنیم.

```
using System;
class Review
{
    static void Main ()
    {
        byte age = 16;
        Console. WriteLine ("My age was " + age);
        }
}
```

برنامه ۲ــ۱ــ نمایش سن

در برنامه بالا علاوه بر نمایش یک رشته مقدار یک متغیر یعنی عدد ۱۶ نیز نشان داده می‌شود. حتماً به یاد دارید که علامت + هنگامی که حداقل با یک رشته به کار می‌رود عمل الحاق رشته‌ها را انجام می‌دهد. بنابراین محتوای متغیر به یک رشته تبدیل شده و به رشته "My age Was" الحاق می‌شود.

سؤال: متغیر نوع byte برای نگهداری اعداد صحیح کوچک استفاده می‌شود. آیا محدوده نگهداری آنها را به خاطر دارید؟

برای نمایش سن شما در سال جاری، تغییراتی در برنامه بالا اعمال می‌کنیم که علاوه بر سن سال قبل، سن سال جاری را نیز نمایش دهد. بدین منظور بعد از دو دستور بالا، یک واحد به متغیر age، اضافه می‌کنیم و مجدداً محتوای متغیر را به همراه یک پیام مناسب چاپ می‌کنیم.

```
using System;
class Review
{
    static void Main ()
    {
        byte age = 16;
        Console. WriteLine ("My age was " + age);
        age = age + 1;
        Console.WriteLine ("Now my age is " + age);
    }
}
```

برنامه ۳ــ۱ــ نمایش سن سال جاری

اما با دقت در متن برنامه متوجه می‌شویم که مترجم از دستور افزایش مقدار متغیر به اندازهٔ یک واحد خطا می‌گیرد. اگر علامت ماوس را روی خط قرمز رنگ ببریم توضیح خطا به صورت زیر نوشته می‌شود :

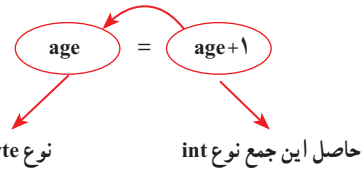
```
age = age + 1;
Cannot implicitly convert type 'int' to 'byte'. An explicit conversion exists (are you missing a cast?)
```

شکل ۱۲ــ۱ــ خطای صادر شده

سؤال: منظور از خطای قبل چیست؟ با استفاده از عملگرهای افزایشی و کاهشی برنامه را بازنویسی کنید.

اکنون می‌خواهیم علت بروز مشکل در دستور; age = age +۱ در برنامه بالا را بررسی کنیم. عملگر + دارای دو عملوند است. یکی از عملوندها، متغیر age از نوع byte و عملوند دیگر عدد ۱ که عددی صحیح است و مترجم به طور پیش فرض اعداد صحیح را از نوع int در نظر می‌گیرد.

ظرفیت داده int، چهار بایت و ظرفیت داده byte یک بایت است و نمی‌توان محتوای int را به‌طور مستقیم درون داده‌های کوچک‌تر از خودش جای داد. بنابراین حاصل عمل جمع، عددی از نوع صحیح int خواهد بود و هنگامی که بخواهید یک مقدار int را در یک متغیر کوچک از نوع byte قرار دهید این مقدار بزرگ‌تر از گنجایش ظرف مورد نظر شما است و بنابراین مترجم خطا اعلام می‌کند.



برای برطرف کردن اشکال، می‌توانید از مترجم بخواهید که حاصل جمع را به نوع byte تبدیل کند تا در متغیر age جای بگیرد. برای این منظور از تبدیلِ نوع[۱] استفاده می‌کنیم (برنامه ۴ــ۱).

شکل کلی تبدیل نوع داده چنین است :

عبارت مبدأ (نوع داده مقصد)

```
using System;
class Review
{
    static void Main ( )
    {
        byte age = 16;
        Console. WriteLine ("My age was " + age);
        age = (byte) (age + 1);
        Console. WriteLine ("Now My age is " + age);
    }
}
```

برنامه ۴ــ۱ــ نوع دیگری از برنامه نمایش سن سال جاری

۱ــ Type Cast

نکته

در تبدیل نوع داده باید به نوع داده‌های مقصد و مبدأ توجه کرد. نوع داده مقصد باید بتواند مقدار دادهٔ مبدأ را در خود جای دهد، زیرا تبدیل نوع داده دارای قوانینی است که در پیوست ۱ جدول مربوط به آن آورده شده است.

مثال ۳ــ۱ : برنامه‌ای بنویسید که با استفاده از تبدیل نوع، عدد را به صورت کاراکتر و همچنین کاراکتر را به صورت عدد تبدیل کرده و نمایش دهد.

الگوریتم یا روش انجام کار : در این برنامه، در متد () Write محتوای متغیر aNumber را که عدد صحیح 67 قرار داده‌ایم، با استفاده از تبدیل نوع (char)، به عنوان کد اسکی یک کاراکتر در نظر گرفته می‌شود که در این صورت، معادل با حرف انگلیسی بزرگ C است.

همچنین محتوای متغیر کاراکتری ch و کاراکتر 'C' با استفاده از تبدیل نوع (int) به یک عدد صحیح که همان کد کاراکتر است تبدیل شده، و با استفاده از متد () WriteLine نمایش داده می‌شوند.

```
using System;
class Review
{

    static void Main ( )
    {
       int aNumber = 67;
       Console. Write ( (char) aNumber);

       char ch = "# ";
       Console. WriteLine (ch);

       Console. WriteLine ("Code of C letter :  " + (int) ' C ');
       Console. WriteLine ("Code of  # :  " + (int) ch);
    }
}
```

برنامه ۵ــ۱ــ نمایش کدهای اسکی کاراکترهای C و #

مثال ۴ــ۱ : برنامه‌ای بنویسید که نام کاربر را سؤال نماید و یک پیام خوشامدگویی به وی اعلام کند.

الگوریتم یا روش انجام کار : در این برنامه برای دریافت نام کاربر، از متد ReadLine() استفاده می‌کنیم و نام کاربر را در یک متغیر ذخیره می‌نماییم. نوع متغیر باید از نوع رشته‌ای String باشد که بتوان یک رشته یا نام کاربر را در آن نگهداری کرد. سپس به وسیلهٔ متد WriteLine () نام کاربر به همراه پیام خوشامدگویی را در خروجی نمایش می‌دهیم. به برنامه زیر دقت کنید.

```
using System;
class Review
{
    static void Main ( )
    {
        Console. Write ("Your Name: ");
        string userName = Console . ReadLine ( );
        Console. WriteLine ("Hi " + userName + " ,Welcom Back to C#! ");
    }
}
```

برنامه ۶ــ۱ــ خوشامد گویی به کاربر

در داخل پرانتز در متد WriteLine () برنامه ۶ــ۱، از علامت + برای اتصال و الحاق رشته‌ها استفاده شده است. عملگر + هنگامی‌که با رشته‌ها و یا متغیرهای رشته‌ای به کار می‌رود، عمل الحاق یا کنار هم قرار دادن مقدار رشته‌ها را انجام می‌دهد.

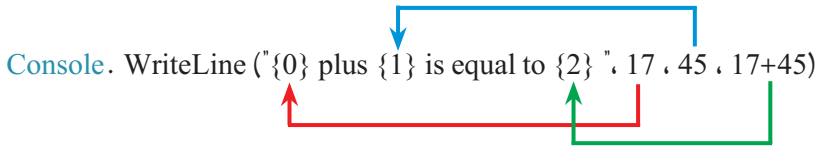
۳ــ۳ــ۱ــ الگوی[۱] جای گذاری در رشته : به جای استفاده از عملگر + می‌توان در متد WriteLine () از الگوی جای گذاری مطابق با روش زیر استفاده کرد.

```
Console. WriteLine ("Hi " {0}, Welcome back to CSharp!" , userName);
```

در این روش، به جای {0}، مقدار متغیر userName قرار می‌گیرد. در واقع {0} جای قرارگیری متغیر را در داخل رشته نشان می‌دهد. اگر بخواهیم مقدار متغیر یا عبارت دیگری را نیز در

۱ــ Formatting

داخل رشته معین کنیم شماره‌های دیگری را بین علامت {} قرار می‌دهیم. مثلاً برای جای عبارت دوم عدد 1 در بین علامت {} استفاده می‌کنیم. مانند دستور زیر که در آن مقدار سه عبارت در رشته با شماره‌های 0 و 1 و 2 مشخص شده است :



علاوه بر محل قرارگیری یک متغیر یا عبارت در یک رشته، می‌توانیم الگو و طریقه نمایش عبارت و همچنین تراز چپ و راست را نیز در صورت نیاز معین کنیم. منظور از عدد تراز تعداد فضای خالی است که در صفحه نمایش، جهت نمایش مقدار متغیر یا عبارت اختصاص داده می‌شود. به طور کلی الگوی جای گذاری چنین است :

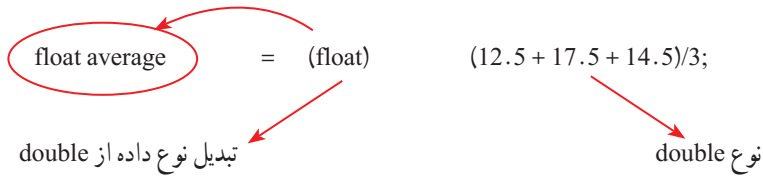
{الگوی نمایش : عدد تراز، شماره}

اگر عدد تراز منفی باشد، مقدار موردنظر در فضای اختصاص یافته چپ چین می‌شود و اگر عدد تراز مثبت باشد، مقدار موردنظر در فضای اختصاص یافته راست چین می‌شود. الگوی نمایش از تعدادی کاراکتر تشکیل می‌شود که هر یک نحوهٔ نمایش داده را مشخص می‌کند. مثلاً برای نمایش اعداد اعشاری با دو رقم اعشار(مثل میانگین نمره) از کاراکتر F استفاده می‌شود. در پیوست۲، لیست کامل کاراکترهای الگوی نمایش برای انواع داده‌ها وجود دارد.

مثال ۵ــ۱ : در برنامه ۷ــ۱ میانگین سه نمره محاسبه شده است و سپس به دو صورت معمولی و همچنین با دو رقم اعشار نشان داده شده است. در هر دو تراز راست با فضای ۱۰ کاراکتر در نظر گرفته شده است.

```
using System;
class Program
{

    static void Main ( )
    {
       float average = (float) (12.5 + 17.5 +14.5) /3;
       Console. Writeline ("Average = {0,10}" , average);
       Console. Writeline ("Average = {0,10: F}" , average);

    }
}
```

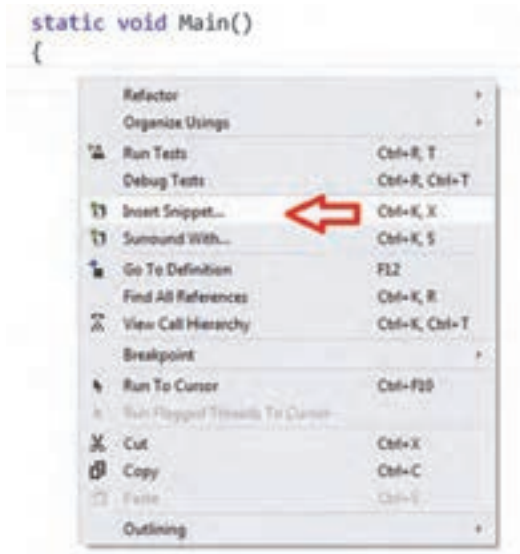
برنامه ۷ــ۱ــ نمایش عدد اعشاری با دو رقم اعشار و با تراز راست



به خاطر دارید پیش فرض برای اعداد اعشاری، نوع double است و در این خط از برنامه به جای استفاده از پسوند F یا f از تبدیلِ نوع استفاده شده است. این عمل با استفاده از کلمه کلیدی (float) در این خط از برنامه انجام شده است.

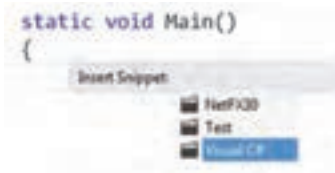
۴ــ۳ــ۱ــ یادآوری دستورات شرطی

۱ــ۴ــ۳ــ۱ــ دستور if–else : از دستور if–else برای تصمیم‌گیری در اجرای دستورات استفاده می‌شود. در جلوی دستور if یک عبارت منطقی ساده و یا مرکب به عنوان شرط نوشته می‌شود، درصورت درست بودن شرط، دستور یا دستورات بعد از این عبارت اجرا می‌شود، در غیر این صورت دستور یا دستورات پس از else اجرا می‌شود. زبان‌های برنامه نویسی، امکاناتی را فراهم کرده‌اند که نیاز به حفظ کردن تمامی جزئیات و تایپ دستورات نیست، به عنوان مثال می‌توانید در پنجره کد نویسی کلیک راست کرده و از منویی که ظاهر می‌شود گزینه Insert Snippet را انتخاب کنید (شکل ۱۳ــ۱).



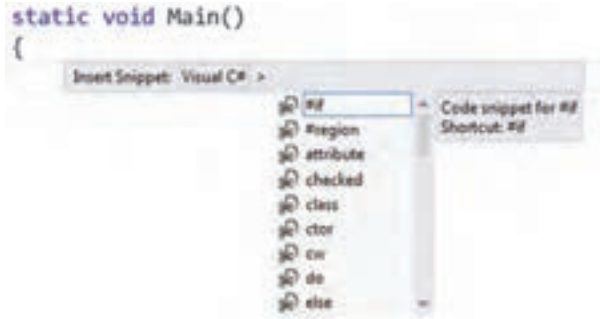
شکل ۱۳ــ۱ــ گزینه Insert Snippet

در این صورت منوی دیگری باز می‌شود و از آن گزینه Visual C# را انتخاب کنید (شکل ۱۴ــ۱).



شکل ۱۴ــ۱ــ انتخاب گزینه Visual C# از Insert Snippet

در این صورت منویی ظاهر می‌شود که دستورات و کلمات رزروشدهٔ زبان سی شارپ را نشان می‌دهد (شکل ۱۵ــ۱).



شکل ۱۵ــ۱ــ انتخاب دستورات برنامه نویسی در گزینه انتخاب شده

به عنوان مثال اگر بخواهیم دستور if را در این لیست پیدا کنیم، حرف i را تایپ می‌کنیم تا به سرعت if در لیست یافت شود. اکنون می‌توانید روی آن کلیک کنید (شکل ۱۵ــ۱). بعد از ظاهر شدن دستور انتخابی، می‌توانیم عبارت داخل پرانتز را به دلخواه تغییر دهیم (شکل ۱۶ــ۱).

```
static void Main()
{
    if (true)
    {

    }
```

شکل ۱۶ــ۱ــ ساختار دستور if

کار در کارگاه ۲

مثال ۶ــ۱ : برنامه‌ای بنویسید که نمرهٔ یک درس را از ورودی دریافت کند و تشخیص دهد آیا این نمره معتبر است یا خیر.

الگوریتم یا روش انجام کار : در این مثال باید عددی از کاربر گرفته شود. متد ReadLine() برای دریافت داده‌ها از ورودی استفاده می‌شود. دادهٔ دریافتی به صورت یک رشته به وسیلهٔ این متد تحویل داده می‌شود. مانند "18.75" بنابراین به یک متغیر رشته‌ای برای ذخیرهٔ آن نیاز داریم. اگرچه کاربر عددی به عنوان نمره وارد می‌کند اما این عدد به صورت رشته‌ای از ارقام، ذخیره می‌شود که قابلیت محاسبات عددی ندارد. بنابراین باید این داده رشته‌ای را به یک عدد تبدیل کنیم. متد Parse() نیز برای تبدیل یک رشته به عدد استفاده می‌شود.

اکنون در شرط دستور if مقدار نمره را بررسی می‌کنیم اگر نمره کمتر از صفر یا بیشتر از ۲۰ باشد نمره غیر مجاز است و باید پیام "نمره غیر مجاز است" بر روی صفحه نشان داده شود. شرط دستور if در این مثال، یک شرط مرکب است که از دو قسمت تشکیل شده است و با کلمه "یا" به یکدیگر مربوط می‌شود. در زبان C# از عملگر || به عنوان عملگر "یا" استفاده می‌شود. در صورتی که شرط برقرار نباشد کنترل برنامه به قسمت else انتقال می‌یابد و دستور مربوط به آن اجرا می‌شود.

```
using System;
class Review
{
    static void Main ( )
    {
        float  number ;
        string input ;
        Console. Write ("Enter a mark: ");
        input = Console. ReadLine ( );
        number = float. parse (in put);
        if ( (number<0) | | (number>20) )
            Console. WriteLine ("Invalid Mark!");
        else
            Console. WriteLine ("Mark is in valid range");
    }
}
```

برنامه ۸ــ۱ــ معتبر بودن نمره

مثال ۷ــ۱ : مثال قبل را توسعه دهید به طوری‌که قبولی یا مردودی را نیز معین کند.

الگوریتم یا روش انجام کار : برای تشخیص قبولی یا مردودی، باید یک دستور if به برنامه قبل اضافه کنیم. اگر نمره کمتر از ۱۰ باشد پیام "شما نمره قبولی نیاوردید. تلاش بیشتری کنید." بر روی صفحه نشان داده شود و در غیر این صورت (در قسمت else آن)، پیام "شما قبول شدید." نمایش داده شود.

قطعه برنامهٔ زیر، دستورِ if جدیدی است که باید به برنامه اضافه شود.

```
if ( (number <10)
    Console. WriteLine ("You failed, try harder next. ");
else
    Console. WriteLine ("You passed. ");
```

سؤال: دستور if جدید را در چه قسمتی از برنامه ۸ ــ۱درج کنیم؟

نکته

اگر کاربر در هنگام ورود داده‌های عددی، کاراکتری غیر از عدد وارد کند، متد ()Parse در تبدیل آن به عدد، دچار خطا شده، و برنامه به طور ناگهانی قطع[1] می‌شود و خطایی ظاهر می‌شود. برای جلوگیری از این مشکل سه راه پیش رو داریم :

۱. پس از دریافت رشته حاوی عدد از کاربر، و قبل از استفاده از متد () Parse، کاراکترهای وارد شده را مورد بررسی قرار دهیم و اگر فقط شامل عدد بود، آن را با استفاده از متد Parse به عدد تبدیل کنیم این روش را در فصل مربوط به رشته‌ها مورد بررسی قرار خواهیم داد.

۲. خطایی که ممکن است متد(Parse) در جریان تبدیل اعداد تولید کند، را کنترل و پیش‌بینی کنیم و برای آن دستورات مناسبی در برنامه قرار دهیم. این روش را در فصل کنترل خطا[2] مورد بررسی قرار می‌دهیم.

۳. به جای استفاده از متد() Parse، از متد () TryParse استفاده شود. این متد اگر تبدیل انجام شود مقدار تبدیل شده را به ورودی دوم خود می‌ریزد و اگر موفق به تبدیل عدد نشود خطایی را نمایش نمی‌دهد و برنامه را قطع نمی‌کند بلکه با تولید یک مقدار منطقی False، مشکل در تبدیل عدد را به اطلاع برنامه‌نویس می‌رساند و برنامه‌نویس با دریافت این مقدار متوجه می‌شود که کاربر عدد را صحیح وارد نکرده است. به عنوان مثال اگر قطعه کد زیر را داشته باشیم :

```
int x;
if (int. Tryparse (str, out x) )
    Console. WriteLine ("your number is: " + x);
else
   Console. WriteLine ("wrong number");
```

اگر متغیر str برابر با متن "8" باشد مقدار x برابر 8 می‌شود و مقدار true به عنوان خروجی TryParse، تولید شده و متن "your number is 8" نمایش داده می‌شود. اما اگر مقدار متغیر str برابر با "a" باشد متد TryParse مقدار false را بر می‌گرداند و در نتیجه متن "wrong number" نمایش داده می‌شود.

۱ــ Crash

۲ــ Handling Exceptions

۲ــ۴ــ۳ــ۱ــ عملگر سه تایی یا علامت سؤال : دستور if را در قطعه برنامه زیر در نظر بگیرید :

```
value ++ ;
if (value > 20)
    max = 20 ;
else
    max = value ;
```

دستورات بالا را می‌توان به صورت خلاصه با استفاده از عملگر علامت سؤال به صورت زیر نوشت :

```
value ++ ;
max = (value > 20) ? 20 : value ;
```

شکل کلی به کار گیری این عملگر به صورت زیر است :

مقدار دوم : مقدار اوّل ? (عبارت منطقی)

کامپیوتر در هنگام برخورد با عملگر سه تایی ابتدا حاصل عبارت منطقی را محاسبه می‌کند، اگر حاصل عبارت برابر درست (True) باشد، نتیجه این عملگر برابر مقدار اول خواهد بود و در صورتی که حاصل عبارت منطقی، برابر نادرست (False) باشد، حاصل این عملگر برابر مقدار دوم است.

سؤال: به نظر شما چرا عملگر ? عملگر سه تایی نامیده می‌شود؟

مثال ۸ــ۱ : در برنامه ۹ــ۱ به جای دستور if از عملگر سه تایی استفاده کنید.

```
using System;
class Review
{
    static void Main ( )
    {
        int number;
        string input , output ;

        Console. Write ("Enter a number: ") ;
        input = Console . ReadLine ( );
        number = int . Parse (input);

        if (number % 2 = = 0)
            output = "Even number";
        else
            output = "Odd number";
        Console. WriteLine ("{0} is an {1}." , number, output) ;
    }
}
```

برنامه ۹ــ۱ــ تشخیص زوج و فرد بودن اعداد

۳ــ۴ــ۳ــ۱ــ دستور switch : برای مواقعی که بخواهیم حالت‌های مختلف یک عبارت را بررسی کرده و بر اساس آن دستورات را اجرا کنیم از دستور switch استفاده می‌کنیم.

کار در کارگاه ۳

مثال ۹ــ۱ : برنامهٔ ۱۰ــ۱ به وسیلهٔ یکی از هنرجویان نوشته شده است. کار این برنامه را بررسی می‌کنیم :

الگوریتم یا روش انجام کار : در این مثال ابتدا یک عدد از کاربر دریافت شده و در یک متغیر ذخیره می‌شود سپس اجرای برنامه وارد دستور switch خواهد شد و عدد دریافتی را با caseهای مختلف از ۱ تا ۳ مقایسه می‌کند اگرعدد وارد شده با هرکدام از caseها برابر باشد دستور مربوط به case اجرا

می‌شود و رنگ زمینه تغییر می‌کند. اما اگر عدد وارد شده با هیچ یک از دستورات case برابر نبود، دستور default اجرا خواهد شد. دستورات مذکور در داخل یک حلقه for قرار دارد که دوبار تکرار می‌شود.

سؤال: آیا می‌توانید برنامه را به صورت دیگری بهتر از آنچه که نوشته شده بنویسید؟

```
using System;
class Program
{
    static void Main ( )
    {
        for (int i = 1; i < = 2; i++)
        {
            int n = int. Parse(Console.ReadLine () );
            switch (n)
            {

                case 1:
                    Console. BackgroundColor = ConsoleColor.Red;
                    Console.Clear();
                    break;
                case 2:
                    Console. BackgroundColor = ConsoleColor.Green;
                    Console.Clear();
                    break;
                case 3:
                    Console. BackgroundColor = ConsoleColor.Blue;
                    Console.Clear();
                    break;
                default:
                    Console. WriteLine ("ERROR");
                    break;
            }
        }
    }
}
```

برنامه ۱۰ــ۱ــ تغییر رنگ صفحه کنسول

۵ــ۳ــ۱ــ یادآوری دستورات تکرار یا حلقه

در مواقعی که می‌خواهیم یک یا چند دستور، بیش از یک بار انجام شوند، به جای اینکه آنها را چندین بار تایپ و یا کپی کنیم، می‌توانیم از دستورات تکرار یا حلقه استفاده نماییم. در کتاب برنامه‌سازی ۱ با انواع دستورات تکرار مانند while، do ــ while و دستور for آشنا شدید. در این قسمت با ذکر چند مثال، انواع دستورات تکرار را یادآوری می‌کنیم.

۱ــ۵ ــ۳ــ۱ــ دستور while : برای مواقعی که یک یا چند دستور باید تا زمانی که حاصل یک عبارت منطقی درست است اجرا شوند، از دستور while استفاده می‌شود.

کار در کارگاه ۴

مثال ۱۰ــ۱ : برنامه‌ای بنویسید که یک عدد را از کاربر دریافت کند و اعداد کمتر از آن تا عدد یک را نشان دهد. مثلاً اگر عدد ۱۰ وارد شد، اعداد ۱۰، ۹، ... تا عدد ۱ به صورت کاهشی یا نزولی نمایش داده شوند.

الگوریتم یا روش انجام کار : در این مثال، ابتدا یک عدد از کاربر دریافت می‌کنیم و آن را در یک متغیر ذخیره می‌نماییم. سپس باید محتوای این متغیر را روی صفحه نشان دهیم و پس از آن، یک واحد از متغیرکم کنیم و مجدداً عملیات نمایش محتوای متغیر و کاهش مقدار آن باید تکرار شود تا زمانی که به عدد یک برسیم.

برای این منظور از دستور تکرار while استفاده می‌کنیم. برنامه زیر را مشاهده کنید :

```
using System;
class Review
{
    static void Main ( )
    {
        int  number;
        string input;
        Console. WriteLine (" Counting down to one" ) ;
        Console. Write (" Enter a number (1–100): " ) ;
        input = Console. ReadLine ( );
        number = int. Parse (input);
        while (number > 0)
```

```
        {
            Console. WriteLine (number);
            number – –;
        }
    }
}
```

برنامه ۱۱ــ۱ــ نمایش اعداد متوالی به صورت کاهشی تا صفر

سؤال: اگر علامت‌های آکولاد باز و بسته در حلقهٔ بالا حذف شوند پس از اجرای برنامه، چه اتفاقی می‌افتد؟

در قطعه برنامه زیر به دستورات تکرار شونده توجه کنید :

```
while (number > 0)
{
    Console. WriteLine (number);
    number – –;
}
```

اگر مایل باشید، می‌توانید دستورات داخل حلقه while را در یک دستور خلاصه کنید. یعنی :

```
while (number > 0)
        Console. WriteLine (number – –);
```

در داخل پرانتز متغیر number به همراه عملگر کاهشی نوشته شده است. چون عملگر – –، بعد از نام متغیر ذکر شده است، ابتدا مقدار متغیر چاپ می‌شود و سپس مقدار آن یک واحد کاهش می‌یابد.

مثال ۱۱ــ۱ : برنامه‌ای بنویسید که یک عدد را از کاربر دریافت کند و تعداد ارقام آن را شمارش و نشان دهد.

الگوریتم یا روش انجام کار : ابتدا باید داده‌ای را از کاربر دریافت کنیم و با توجه به اینکه متد() ReadLine داده دریافتی را به صورت یک رشته تحویل می‌دهد، برای شمارش تعداد ارقام

عدد، به دو روش زیر می توانیم عمل کنیم :

۱ــ تعداد کاراکتر رشته دریافتی را شمارش کنیم که همان ارقام عدد هستند[۱]. در فصل سوم با این روش تعداد ارقام عدد را حساب می‌کنیم.

۲ــ مانند مثال‌های قبلی، رشته دریافتی را به عدد تبدیل کرده، و سپس با استفاده از تقسیم‌های متوالی به ۱۰ تعداد دفعات تقسیم را می‌شماریم که همان تعداد ارقام عدد است. در برنامه ۱۲ــ۱ از این روش استفاده شده است.

```
using System;
class Review
{
    static void Main ( )
    {
        int  number , remainder, digits=0;
        string input;
        Console. Write (" Enter a number: " ) ;
        input = Console. ReadLine ( );
        number = int. Parse (input);
        while (number > 0)
        {
            remainder = number % 10 ;
            digits ++ ;
            number = number / 10 ;
        }
        Console. WriteLine ("The number has {0} digits" , digits);
    }
}
```

برنامه ۱۲ــ۱ــ نمایش شمارش ارقام

سؤال: آیا این برنامه برای اعداد منفی نیز کار می‌کند؟ اگر پاسخ خیر است با کمترین تغییر در شرط دستور while کاری کنید که برای اعداد منفی نیز برنامه به درستی کار کند.

۱ــ البته علامت عدد و یا فاصله را باید در نظر داشت و به‌عنوان تعداد ارقام آنها را محسوب نکرد.

۲ــ۵ ــ۳ــ۱ــ دستور for : برای اجرای دستور یا دستورات به تعداد دفعات معین، از دستور for استفاده می‌کنیم.

مثال ۱۲ــ۱ : در برنامه مثال قبل (چاپ اعداد به صورت کاهشی)، به جای دستور while از دستور for استفاده کنید.

مثال ۱۳ــ۱ : برنامه‌ای بنویسید که نمرات درس برنامه سازی ۲ مربوط به یک کلاس ۱۰ نفری را دریافت کند و مجموع و میانگین نمرات را حساب کند.

الگوریتم یا روش انجام کار : در این مثال چون عمل دریافت نمرات و محاسبه مجموع باید ده بار تکرار گردد از یک حلقه استفاده می‌کنیم که در داخل حلقه عملیات مربوطه را قرار می‌دهیم. اگرچه از هر نوع حلقه می‌توانیم برای این منظور استفاده کنیم اما در این مثال حلقه for را به کار می‌گیریم.

```
using System;
class Review
{
    static void Main ( )
    {
        float number, total =0, average;
        string input;
        for (int i = 0; i < 10; i++)
        {
            Console. Write (" Enter a mark: " ) ;
            input = Console. ReadLine ( );
            number = float. Parse (input);
            total = total + number;
        }
        average = total / 10 ;
        Console. WriteLine ("Total : " + total);
        Console. WriteLine ("Average : " + average);
    }
}
```

برنامه ۱۳ــ۱ــ محاسبه مجموع و معدل ۱۰نمره

۴ــ۱ــ استفاده از مقدار ثابت[۱] در برنامه

مثال ۱۴ــ۱ : برنامه مثال قبل را برای یک کلاس با ۱۵ نفر دانش‌آموز تغییر(توسعه) دهید.

الگوریتم یا روش انجام کار : در این حالت لازم است به جای دریافت ۱۰ عدد، پانزده عدد دریافت کنیم، بنابراین حلقه تکرار for باید به تعداد ۱۵ بار اجرا شود. برای این منظور کافی است عدد ۱۰ را در حلقهٔ for را به ۱۵ تغییر می‌دهیم. همچنین باید مجموع نمرات را به تعداد نمرات دریافتی یعنی عدد ۱۵ تقسیم کنیم. به این ترتیب باید در انتهای برنامه مجموع نمرات (total) را به عدد ۱۵ تقسیم کنیم.

اما فرض کنید که برنامه نویس فراموش کند عدد ۱۰ دوم را تغییر دهد در این صورت با اجرای برنامه پانزده نمره دریافت می‌شود و مجموع نمرات به درستی حساب می‌شود اما میانگین نمرات به اشتباه حساب می شود. برای جلوگیری از چنین اشتباهاتی و همچنین توسعه راحت برنامه می‌توان در ابتدای کلاس یک شناسه به عنوان یک ثابت تعریف کرد و عدد ۱۰ را به آن نسبت داد سپس در سرتاسر برنامه هر جا که به عدد ۱۰ نیاز بود از آن شناسه استفاده کرد.

برای تعریف شناسه یا نام ثابت از کلمه کلیدی const استفاده می‌شود. معمولاً نامی‌که برای اعداد ثابت تعریف می‌شود با حروف بزرگ نوشته می‌شود تا در برنامه مشخص باشد که این شناسه، یک مقدار ثابت است. نحوهٔ تعریف ثابت‌ها، مانند تعریف متغیرها می‌باشد با این تفاوت که در ابتدای تعریف آنها کلمه const قرار دارد.

```
const int SIZE = 15;
```

برنامه ۱۴ــ۱ را در نظر بگیرید :

```
using System;
class Review
{
    const int SIZE = 15;  ⇦
    static void Main ( )
    {
        float number, total =0, average;
        string input;
        for (int i = 0; i < SIZE; i++)  ⇦
        {
```

۱ــ Constant

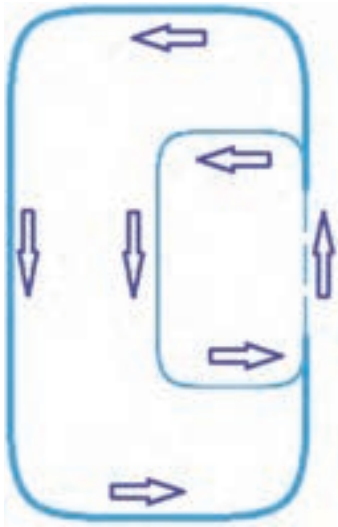
```
            Console. Write (" Enter a mark: ") ;
            input = Console. ReadLine ( );
            number = float. Parse (input);
            total = total + number;
        }
        average = total / SIZE ;
        Console. WriteLine ("Total : " + total);
        Console. WriteLine ("Average : " + average);
    }
}
```

برنامه ۱۴ــ۱ــ امکان تغییر راحت برنامه برای دریافت ۱۵ نمره

سؤال: اگر بخواهید برنامهٔ ۱۴ــ۱ برای محاسبه میانگین نمرات پنج دانش‌آموز به کار رود، چه تغییری در آن اعمال می‌کنید؟

۵ــ۱ــ حلقه‌های متداخل (تو در تو)

در بعضی از مواقع، یک دستور تکرار را در داخل دستور تکرار دیگری به کار می‌بریم. به عبارت دیگر هنگامی که در داخل یک حلقه، حلقه دیگری قرار داشته باشد، حلقه‌های تودرتو یا متداخل[۱] نامیده می‌شوند.



شکل ۱۷ــ۱ــ پیست دومیدانی مشابه حلقه‌های تودرتو

حلقه‌های تودرتو را مانند پیست دومیدانی در نظر بگیرید، حال دونده‌ای را در نظر بگیرید که طبق دستور مربی، باید به ازای هر بار دویدن در پیست بزرگ، مجبور باشد که در داخل پیست کوچک نیز پنج بار طناب بزند. اگر مربی وی از او بخواهد به تعداد ۱۰ بار در پیست بزرگ با همان شرط ذکر شده بدود، در مجموع، چند بار در پیست کوچک طناب زده است؟

۱ــ Nested Loops

مثال ۱۵ــ۱ : در برنامه ۱۵ــ۱، از دو دستور for استفاده شده است که یکی در داخل دیگری قرار دارد و بنابراین حلقه تودرتو را تشکیل می‌دهند.

الگوریتم یا روش انجام کار : در این برنامه، دستور for با متغیر شمارنده i، حلقه بیرونی[۱] و دستور for با متغیر شمارنده j، حلقه داخلی[۲] را تشکیل می‌دهد. کامپیوتر به ازای هر بار اجرای حلقه بیرونی، حلقه داخلی را به طور کامل (پنج بار) انجام می‌دهد.

```
using System;
class Review
{
    static void Main ( )
    {
        for (int i = 1; i <= 10; i++)
        {
            for (int j = 1; j <= 5 ; j++)
                Console. Write (" *" ) ;
            Console. WriteLine ();
        }
    }
}
```

حلقه بیرونی

حلقه داخلی

برنامه ۱۵ــ۱ــ حلقه تو در تو

در حلقه داخلی دستور نمایش رشته "*" قرار دارد، بنابراین با اجرای حلقه داخلی، پنج بار رشته "*" در کنارهم در روی یک خط در صفحه نمایش به صورت زیر چاپ شود :

*****

از آنجا که حلقه بیرونی، ده بار تکرار می‌شود بنابراین طرح زیر، روی صفحه نمایش نقش می‌بندد :

*****
*****
*****
*****

۱ــ Outer Loop

۲ــ Inner Loop

```
*****
*****
*****
*****
*****
*****
```

سؤال: نقش دستور ;() Console . WriteLine در این برنامه چیست؟

سؤال: برای اینکه در ابتدای هر خط در طرح فوق، شماره سطر را چاپ کنیم، چه باید کرد؟

کار در کارگاه ۵

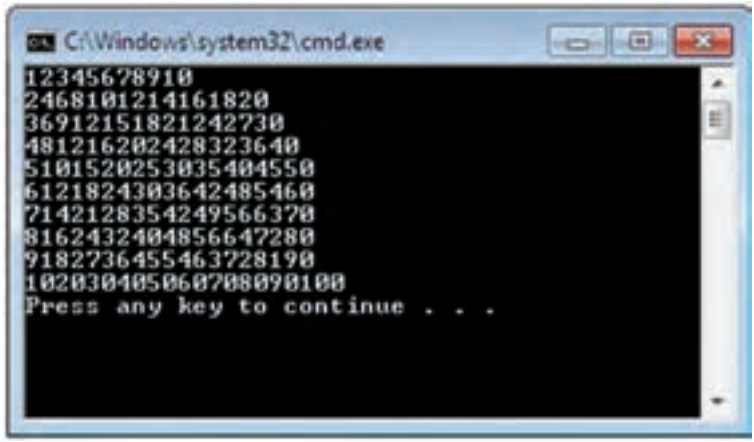
مثال ۱۶ــ۱ : برنامه‌ای بنویسید که جدول ضرب اعداد ۱ تا ۱۰، را روی صفحه نشان دهد.

الگوریتم یا روش انجام کار : برای نمایش جدول ضرب اعداد ۱ تا ۱۰، کافی است در برنامه قبل به جای نمایش رشته "*" حاصل عبارت i*j را محاسبه و روی صفحه چاپ کنیم.

```
using System;
class Review
{
    static void Main ( )
    {
        for (int i = 1; i < = 10; i++)
        {
            for (int j = 1; j <= 10; j++)
                Console. Write ("{0}", i* j) ;
            Console. WriteLine ();
        }
    }
}
```

برنامه ۱۶ــ۱ــ جدول ضرب اعداد ۱ تا ۱۰

نتیجه اجرا یا خروجی برنامه فوق چنین خواهد بود :

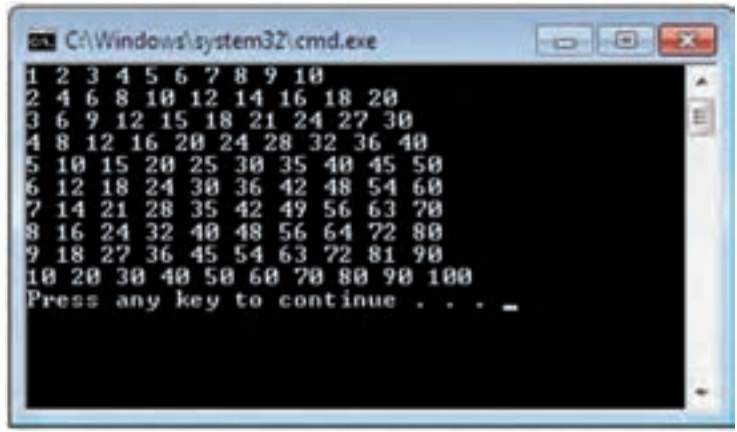


شکل ۱۸ــ۱ــ خروجی برنامه جدول ضرب

همان‌طور که مشاهده می‌کنید هیچ فاصله‌ای بین اعداد وجود ندارد و اعداد قابل تشخیص نیستند. برای رفع این مشکل باید بین هر عدد فاصله چاپ کنیم. یک روش ابتدایی اضافه‌کردن یک یا دو فاصله برای هر عدد است که در متد ( ) Write اعمال کنیم :

```
Console. Write ("{0} ", i*j) ;
```

در این صورت، نتیجه اجرا یا خروجی برنامه چنین خواهد شد :



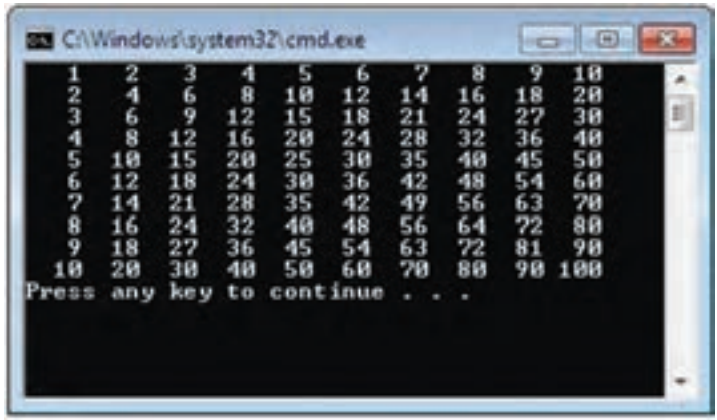
شکل ۱۹ــ۱ــ جدا کردن اعداد جدول ضرب

در این حالت خواندن اعداد بهتر شد و می‌توان اعداد را از یکدیگر تمیز داد. اما شکل کلی خروجی چندان جالب نیست و اعداد در زیر یکدیگر قرار ندارند.

خوشبختانه در زبان C#، راه حل چنین مشکلی در نظر گرفته شده است و امکان شکل یا الگوی[1] نمایش از قبل پیش بینی شده است. اعداد جدول ضرب دو رقمی‌هستند به جز حالت خاص ضرب ۱۰ در ۱۰ است که حاصل برابر ۱۰۰ و یک عدد ۳ رقمی‌است بنابراین اگر برای چاپ هر عدد ۳ مکان و یک مکان نیز به عنوان فاصله در نظر بگیریم، در مجموع فضایی به اندازه ۴ کاراکتر، حداقل فضای لازم برای جداسازی اعداد است. در علامت{0} بعد از عدد صفر، تعداد فضای اختصاص داده شده را به صورت زیر مشخص می‌کنیم :

```
Console. Write ("{0,4}", i*j) ;
```

در این صورت خروجی برنامه چنین خواهد بود :



شکل ۲۰ــ۱ــ خروجی نهایی برنامه جدول ضرب

۶ــ۱ــ کاراکتر \[2]

در زبان‌های برنامه نویسی، از جمله زبان C#، کاراکتر \، یک کاراکتر خاص و معنی‌دار است. هر گاه این کاراکتر، در رشته‌ای دیده شود، کاراکتر بعدی آن اثر و عملکرد خاصی، جدای از شکل ظاهری آن دارد، به عبارت دیگر کاراکتر \، سبب تغییر عملکرد کاراکتر بعدی می‌شود. به همین دلیل

۱ــ Format String

۲ــ Back Slash Character

به این مجموعه کاراکترها که کاراکتر اول \ و دومی کاراکتر دیگری است، دنباله فرار[1] گفته می‌شود.

یکی از کاربردهای این دنباله، نمایش علامت" در یک رشته است. اگر بخواهیم جمله‌ای را همراه با علامت " روی صفحه نمایش چاپ کنیم به مشکل برخورد می‌کنیم. چون معنای معمول این کاراکتر، نشانه شروع و پایان یک رشته است و در وسط یک رشته نمی‌تواند قرار گیرد. لذا باید معنی معمول آن را تغییر دهیم. بنابراین از دنباله "\ برای بیان این کاراکتر استفاده می‌کنیم. :

```
Console. WriteLine ("C# is pronounced \"see Sharp\" " ) ;
                                  ⇧دنباله معنی دار⇧
```

سؤال: خروجی دستور بالا چیست؟

در جدول زیر تعدادی از دنباله‌های معنی‌دار نشان داده شده‌اند.

جدول ۱ــ۱ــ دنباله معنی‌دار

عملکرد	دنباله
ایجاد یک بوق هشدار[2]	\a
حذف یک کاراکتر(Backspace)	\b
ایجاد یک خط خالی(New Line)	\n
ایجاد یک فاصله افقی زیاد tab	\t
ایجاد یک تک کوتیشن (')	\'
ایجاد یک دابل کوتیشن (")	\"
ایجاد یک Back slash (\)	\\

نکته

زبان C# روی حروف کوچک و بزرگ حساس است و این شامل دنباله‌های جدول بالا نیز می‌شود.

۱ــ Escape Sequence ۲ــ Alarm

مثال ۱۷ــ۱ : در برنامه ۱۷ــ۱ دنباله‌های معنی‌دار استفاده شده‌اند. خروجی هر دستور به‌صورت توضیح در جلوی آن نشان داده شده است.

```
using System;
class Escape Sequence
{
static void Main ( )
{
Console. WriteLine ("Escape Sequence") ;              // Enscape Sequence
Console. WriteLine ("\t Escape Sequence") ;           //          Enscape Sequence

Console. WriteLine ("\n Escape Sequence") ;           //
                                                      // Enscape Sequence
Console. WriteLine ("\" Escape Sequence") ;           // " Enscape Sequence
Console. WriteLine ("\' Escape Sequence") ;           // ' Enscape Sequence
Console. WriteLine (" Escape\ b Sequence") ;          // ' Enscape Sequence
Console. WriteLine ("\ \  Escape Sequence") ;         // \ Enscape Sequence
Console. WriteLine ("\ ~ Escape \b Sequence") ;       // Error
```

برنامه ۱۷ــ۱ــ نمایش دنباله معنی دار

خودآزمایی فصل اول

الف) درستی یا نادرستی هر عبارت را تعیین کنید.

۱ــ مقدار یک شناسه ثابت را در طول برنامه می‌توان تغییر داد.

۲ــ برای بررسی حالات مختلف یک عبارت و اجرای دستورات بر اساس آن از دستور switch استفاده می‌کنیم.

۳ــ برای نمایش کاراکترهای خاص مثل " وسط یک رشته از دنباله معنی دار استفاده می‌شود.

ب) جاهای خالی را با عبارت مناسب پر کنید.

۴ــ لیستی از نام متدها و دستورات که هنگام تایپ نمایش داده می‌شود ..... نام دارد.

۵ــ برای نمایش منوی دستورات و کلمات رزرو شده سی شارپ از فرمان ..... استفاده می‌شود.

۶ــ برای تعریف شناسه یا نام ثابت از کلمه کلیدی ..... استفاده می‌شود.

۷ــ IDE ساده و کم حجم، برای نوشتن و ترجمه برنامه‌های سی شارپ..... نام دارد.

ج) به سؤالات زیر پاسخ دهید.

۸ــ تفاوت متد Parse و Try parse را بنویسید.

۹ــ قطعه برنامه زیر را Trace کنید و در ماتریس زیر جای‌گذاری کنید.

```
for (int i=1; i<=4; i++)
 {
    for (int j=4; j>=1; j--)
     Console.WriteLine( (i=j) ? 1: 0);
   Console.WriteLine();
 }
```


۱۰ــ در اجرای قطعه کد زیر پیام خطا داده می‌شود. به نظر شما مشکل چیست؟ با تغییر کد مشکل را برطرف کنید.

```
int x = 14.35;
```

۱۱ــ عملکرد هر یک از سه دستور زیر را بررسی کنید. آیا خروجی این سه دستور یکسان است؟

الف)

```
Console. WriteLine ("Hi" + userName +" , welcome back to CSharp!");
```

ب)

```
Console. WriteLine ("Hi" {0}, welcome back to CSharp!" , userName);
```

ج)

```
Console. WriteLine ("Hi" {0}, {1}", userName, "welcome back to CSharp!" );
```

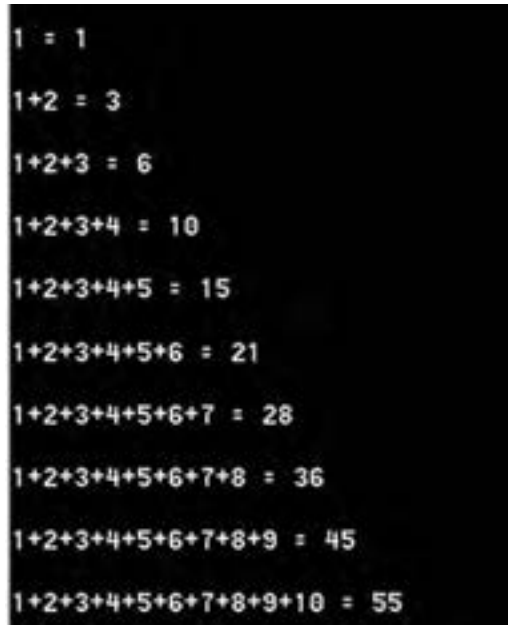
۱ــ برنامه‌ای بنویسید که شکل روبرو را روی صفحه نمایش دهد.

۲ــ برنامه زیر سه عدد را دریافت می‌کند و عدد بزرگ‌تر را نمایش می‌دهد. از عملگر ۳ تایی یا عملگر علامت سؤال به جای دستورات if استفاده کنید.

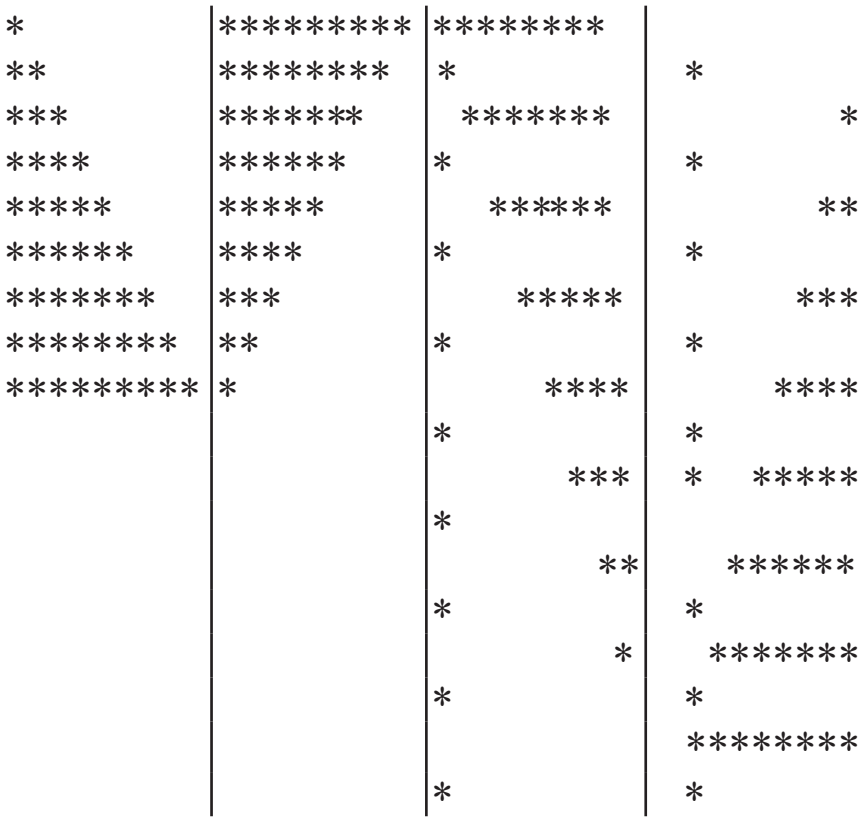
```
using System;
class Review
{
    static void Main ( )
    {
        float a, b, c, max;
        string input;
        Console. WriteLine ("This program finds the maximum number.");
        Console. Write ("Enter first number: ");
        input = Console. ReadLine ( );
        a = float. Parse (input);
        Console. Write ("Enter second number: ");
        input = Console. ReadLine ();
        b = float. Parse (input);

        Console. Write ("Enter third number: ");
        input = Console. ReadLine ();
        c = float. Parse (input);

        max =a;
        if (max < b)
```

```
        max = b;
    if (max < c)
        max = c;
    Console. WriteLine ("Maximum number is " + max);

    }
}
```

۳ــ برنامه‌ای بنویسید که با استفاده از حلقه‌های تو در تو، مجموع اعداد طبیعی را طبق شکل روبرو محاسبه و نمایش دهد.



۴ــ برنامه‌هایی بنویسید که هر یک بتواند یکی از طرح‌های زیر را نمایش دهد.



۵ــ خروجی اجرای دستورات زیر چیست؟

```
string columns = "Column 1\tColumn 2\tColumn 3" ;
string rows = "Row 1\r\nRow 2\r\nRow 3";
Console. WriteLine (columns);
Console. WriteLine (rows);
```

۶ــ تمرین زیر به زبان انگلیسی است، معنی و مفهوم آن را درک کرده، سپس آن را حل کنید.

Lucky Numbers:

a) Write a program to find and print all four_digit numbers of the type abcd, where: a+b = c+d

Example: 2745 is a lucky number, where 2+7 = 4+5 → 9 = 9

(Hint: Use four nested for_loops _ one for each digit as seen below)

```
using System;
class Program
{
    static void Main (string [ ] args)
    {
        Console. WriteLine ("Four Digits Lucky numbers: ");
        for (int a = 1; a<= 9; a++)
            for (int b = 0;  b <= 9; b++)
                for (int c = 0; c <= 9; c++)
                    for (int d = 0; d < = 9; d++)
                        if  (                )
                            Console. WriteLine ("{0}{1} {2} {3}", a, b, c, d);
    }
}
```

b) How many lucky numbers are printed? Add a statement to your program to calculate how many lucky numbers are there.

فعالیت

متن زیر از مستندات MSDN با موضوع حلقه‌های تکرار، برداشت شده است. آن را با کمک هم کلاسی خود ترجمه کنید و به کلاس ارایه نمایید.

A loop is a statement, or set of statements, that are repeated for a specified number of times or until some condition is met. The type of loop you use depends on your programming task and your personal coding preference. One main difference between C# and other languages, such as C++, is the foreach loop, designed to simplify iterating through arrays or collections.

واژگان و اصطلاحات انگلیسی فصل اول

ردیف	واژه انگلیسی	معنی به فارسی
۱	Alarm	
۲	Asterisk	
۳	Back Slash Character	
۴	Command Prompt	
۵	Comment	
۶	Commercial	
۷	Constant	
۸	Counter	
۹	Crash	
۱۰	Disk Operating System	
۱۱	Escape Sequence	
۱۲	Exception	
۱۳	Format String	
۱۴	Handling Exceptions	
۱۵	Inner Loop	
۱۶	Integrated Development Environment	
۱۷	Intelligent Sense	
۱۸	Nested Loops	
۱۹	Open Source	
۲۰	Outer Loop	
۲۱	Source	
۲۲	Space Bar	

فصل دوم

آرایه

بخاطر دارید که متغیر مکانی از حافظه است که مانند یک ظرف مقداری را در خود نگه می‌دارد و با ورود مقدار جدید، مقدار قبلی آن از بین می‌رود. برخی مواقع در برنامه‌ها نیاز به نگه‌داری داده‌ها داریم و گاهی مقدار این داده‌ها خیلی زیادند و نمی‌توانیم ده‌ها و یا حتی صدها متغیر تعریف کنیم. بنابراین لیستی از داده‌های همنام تعریف می‌کنیم. در این فصل می‌خواهیم از مفهوم آرایه[1] برای تعریف چنین متغیرهایی استفاده کنیم و اهمیت و کاربرد آنها را با چند مثال مورد توجه قرار دهیم.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ آرایه و کاربرد آن را توضیح دهد.

۲ــ متغیری از نوع آرایه تعریف کند و آن را در برنامه به کار بندد.

۳ــ روش مرتب سازی حبابی را شرح داده و برنامه مرتب سازی عناصر آرایه را بنویسد.

۴ــ با استفاده از دستورforeach به عناصر آرایه و رشته دسترسی پیدا کند.

۵ــ روش‌های جست‌وجو را توضیح داده و تفاوت آنها را بیان نماید.

۶ــ کلید جستجو را با استفاده از برنامه جستجوی خطی و دودویی پیدا کند.

۱ــ Array

۱ــ۲ــ تعریف آرایه

به مثال زیر توجه کنید.

مثال ۱ــ۲ : فرض کنید می‌خواهیم برنامه‌ای بنویسیم که در یک کلاس ۱۶ نفری محاسبه کند چند درصد نمره‌ها، بالاتر از میانگین کلاس در درس برنامه سازی۲ هستند.

الگوریتم یا روش انجام کار : برای نوشتن این برنامه، پس از دریافت نمرات کل کلاس لازم است ابتدا مجموع و میانگین نمرات را محاسبه کنیم. سپس هر نمره را با میانگین، مقایسه کرده و محاسبه کنیم چه تعدادی از نمرات بیشتر از میانگین کلاس بوده‌اند. برای شمردن این نمرات از شمارنده استفاده می‌کنیم و سپس درصد می‌گیریم.

برای نوشتن این برنامه، ناگزیریم از نمراتِ کلاس دوبار استفاده کنیم :

ــ بار اول در خواندن ورودی‌ها.

ــ بار دوم مقایسهٔ ورودی‌ها با میانگین محاسبه شده.

طبق روال کتاب برنامه سازی۱ متغیری را از نوع اعشاری تعریف می‌کنیم و در یک حلقه for، ۱۶ بار دریافت کرده و جمع و میانگین را محاسبه می‌کنیم.

سؤال: آیا می‌توانیم برای بار دوم از داده‌های ورودی استفاده کنیم و عمل مقایسه را انجام دهیم؟

این نیازِ ما به داده‌های ورودی سبب می‌شود به جای تعریف یک متغیر اعشاری برای نمره و ۱۶ بار تکرار دریافت داده، از ۱۶ متغیر اعشاری که یک لیست پشت سر هم را تشکیل می‌دهند استفاده کنیم، تا در صورت نیاز به دفعات لازم از آنها بهره ببریم.

در مبحث حلقه‌ها یاد گرفتیم که برای دریافت ۱۶ نمره به صورت زیر عمل کنیم :

```
string input;
float mark, total =0;
for (int i = 0; i <= 15; i++)
{
    Console. Write ("Enter a mark: ");
    input = Console. ReadLine ();
    mark = float. Parse (input);
    total = total + mark;
}
```

سؤال: متغیر mark در این روش پس از پایان حلقه، چه مقداری را در خود دارد؟

در روش بالا ما به همه ۱۶ داده‌ای که درون یک متغیر ریخته شده است دسترسی نداریم. چرا؟ بنابراین چگونه می‌توانیم برای بار دوم به داده‌ها دسترسی پیدا کنیم؟ پاسخِ این سؤال در تعریف متغیر قرار دارد. ما نیازمند خانه‌های بیشتری از حافظه هستیم تا داده‌ها را به صورت تک تک در آنها قرار دهیم. این خانه‌های پشتِ سرِهم و همنام آرایه نام دارند.

تعریف آرایه : مکان‌های متوالی در حافظه کامپیوتر که در آن داده‌هایی از یک نوع نگهداری می‌شوند آرایه نام دارند. مثلاً ۱۶ مکان متوالی را در نظر بگیرید که در هر یک از آنها نمره درسی دانش‌آموزان یک کلاس قرار دارد.

17.5	15	19	16.5	.	.	.	20	14	18

هر یک از مکان‌ها، یک عنصرِ[۱] آرایه نامیده می‌شود. برای تفکیک و دسترسی به‌این مکان‌ها، از یک عدد صحیح به نام اندیس[۲] استفاده می‌شود. (مانند پلاک خانه‌های یک کوچه) اولین عنصر آرایه با اندیس صفر مشخص می‌شود و عنصر بعد از آن شمارهٔ یک و به ترتیب جلو می‌رود تا آخرین عنصر که یک واحد کمتر از تعداد عناصر آرایه است. بنابراین آرایه‌ای با ۵ عنصر دارای اندیس‌های 5، 4، 3، 2، 1، 0 می‌باشد. تعداد عناصر آرایه را اندازه یا طول[۳] آرایه می‌نامند.

سؤال: در مثال بالا اندیس آخرین عنصر آرایه چه عددی است؟

۲ــ۲ــ نحوه ایجاد یا تعریف آرایه در زبان C#

به خاطر دارید که برای تعریف یک متغیر از نوع float به شکل زیر عمل می‌کردیم :

```
float mark;
```

برای تعریف یک آرایه از نوع float مشابه تعریف یک متغیر به شکل زیر عمل می‌شود :

```
float[] mark;
```

۱ــ Element

۲ــ Index

۳ــ Length Or Size

اما تعریف بالا هنوز کامل نیست و فقط مرحلهٔ اولِ تعریف است. در زبان سی شارپ برای تعریف یک آرایه باید دو مرحله‌ای عمل کنیم :

شکل کلی مرحله اول تعریف آرایه :

نام متغیر آرایه [] نوع داده ;

در تعریف متغیر گفته شد مکانی از حافظه برای نگهداری داده‌هاست اما با تعریف بالا هنوز مکانی از حافظه برای آرایه تخصیص نیافته است. با استفاده از عملگر new و مشخص کردن اندازه آرایه در مرحله دوم، تعریف کامل می‌شود.

شکل کلی مرحله دوم تعریف آرایه :

نام متغیر آرایه = new نوع داده [اندازه آرایه] ;

بنابراین برای تعریف متغیرmark با ۲۰ عنصر به صورت زیر عمل می‌کنیم :

```
float[] mark;          ← مرحله اول
new float [20]         ← مرحله دوم
```

به جای دو دستور بالا می‌توانیم دستور زیر را جایگزین نماییم :

```
float[ ] list = new float [20];
```

شکل کلی تعریف آرایه :

نام آرایه [ ] = new نوع داده [اندازه آرایه] ;

۳ــ۲ــ دسترسی، مقداردهی و نمایش عناصر آرایه

۱ــ۳ــ۲ــ دسترسی به عناصر آرایه : برای دسترسی به عناصر آرایه، از نام آرایه با

ذکراندیس به صورت زیر استفاده می‌کنیم :

نام متغیر آرایه [اندیس] ;

برای مثال list[0] بیانگر اولین عنصر آرایه و list[19] آخرین عنصر، که بیستمین عضوِ آرایه list است.

۲ــ۳ــ۲ــ مقداردهی عناصر آرایه : برای مقداردهی عناصر آرایه به چند روش می‌توان عمل کرد. یک روش مقداردهی، با استفاده از دستور انتساب است. برای مثال ذخیره کردن نمره 17.5 در اولین عنصر آرایه را با استفاده از دستور انتساب مشاهده می‌کنید.

```
list[0]=17.5f;
```

سؤال: به یاد دارید که چرا در انتهای عدد 17.5 حرف f قرار دارد؟

برای مقداردهی هر عنصر آرایه در طول اجرای برنامه، می‌توانید داده‌ای را از کاربر دریافت کرده و آن را در عنصری از آرایه نگهداری کنید. مثلاً نمره‌ای را از کاربر دریافت و در سومین عنصر ذخیره می‌کنیم :

```
string input;
float [] list = new float [20];
Console. Write ("Enter your mark: ");
input = Console. ReadLine();
list[2] = float. Parse (input);
```

عنصر سوم ←

و یا برای دریافت تمام نمرات و قراردادن آنها در آرایه list به صورت زیر عمل می‌کنیم :

```
for (int i = 0; i < 20; i++)
{
    Console. Write ("Enter a mark: ");
    input = Console. ReadLine ();
    list [i] = float. Parse (input);
}
```

روش دیگر مقداردهی عناصر آرایه در هنگام تعریف و ایجاد آرایه است که در انتهای تعریف آرایه در بین علامت‌های { } مقدار هر عنصر از آرایه را به ترتیب معین می‌کنیم :

نام آرایه [] نوع داده = new نوع داده [] {مقدار، ...، مقدار، مقدار} ;

همان‌طور که مشاهده می‌کنید عددی بین علامت‌های [ ] وجود ندارد، بنابراین اندازه آرایه را ذکر نمی‌کنیم بلکه اندازه چنین آرایه‌هایی با تعداد مقادیر نوشته شده بین علامت‌های { } تعیین می‌شود. مثلاً برای ایجاد آرایه‌ای برای سکه‌های رایج بر حسب ریال، به صورت زیر تعریف می‌کنیم.

```
int [] coin = new int [] {500,1000, 2000, 5000};
```

دستور بالا آرایه‌ای به نام coin شامل چهار عنصر ایجاد می‌کند که هر عنصر آن مقدار یک سکه را مشخص می‌کند که از سکه ۵۰۰ ریالی شروع و به سکه ۵۰۰۰ ریالی ختم می‌شود.
دستور بالا را می‌توان با دستورات زیر جایگزین نمود :

```
int[] coin = new int [4];

coin [0] = 500;
coin [1] = 1000;
coin [2] = 2000;
coin [3] = 5000;
```

همچنین با توجه به اینکه آرایه coin دارای مقادیر اولیه مشخص است می‌توان بدون استفاده از عملگر new آن را ایجاد کرد. بنابراین دستورات بالا را می‌توان با دستور زیر جایگزین نمود :

```
int[] coin = {500, 1000, 2000, 5000};
```

نکته

بعد از ایجاد آرایه، نمی‌توانید اندازه آن را تغییر دهید یعنی نمی‌توانید عنصری به آن اضافه و یا کم کنید.

بنابراین می‌توان در تعریف آرایه‌هایی که از قبل مقادیر اولیه‌شان مشخص است بدون استفاده از عملگر new به شکل زیر عمل کرد :

نام آرایه [] نوع داده = {مقدار، مقدار، ...، مقدار} ;

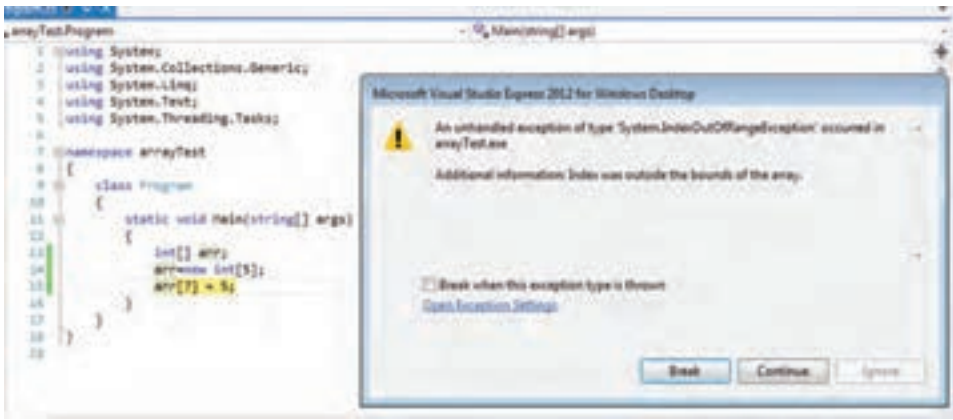
۳ــ۳ــ۲ــ نمایش عناصر آرایه : برای نمایش محتوای عناصر آرایه می‌توان از متد Write() یا WriteLine() استفاده نماییم. مثلاً برای نمایش محتوای آخرین عنصر آرایه نمرات، خواهیم داشت :

```
System. Console. WriteLine ("Last mark: " + list [19]);
```

سؤال: عناصر اندیس‌های زوج این آرایه را با حلقه for نمایش دهید؟

نکته

در زبان C# محدوده اندیس آرایه کنترل می‌شود و نباید از عدد صفر کمتر و همچنین از اندازه آرایه بیشتر یا مساوی باشد. اگر برنامه‌نویس اشتباه کند و اندیس بالاتری را استفاده کند در هنگام ترجمه برنامه با خطا روبه‌رو می‌شود (شکل ۱ــ۲).



شکل ۱ــ۲ــ خطای سرریزی محدوده اندیس آرایه

کار در کارگاه ۱

در آغاز فصل، مثال ۱ــ۲ را با الگوریتم ذکر کردیم. اکنون می‌خواهیم مرور دوباره داشته باشیم و برنامه را در محیط VS بنویسیم. می‌خواستیم برنامه‌ای بنویسیم که در یک کلاس ۱۶ نفری محاسبه کند چند درصد نمره‌ها، بالاتر از میانگین کلاس در درس برنامه‌سازی۲ هستند.

الگوریتم یا روش انجام کار : برای نوشتن این برنامه، پس از دریافت نمرات به وسیلهٔ حلقه در آرایه و محاسبه میانگین، در حلقهٔ دوم، عمل مقایسه صورت می‌گیرد. هر نمره با میانگین مقایسه شده و درصورتی که نمره بیشتر از میانگین کلاس باشد، به وسیلهٔ یک متغیر به عنوان شمارنده، تعداد را محاسبه می‌کنیم و سپس درصد می‌گیریم.

```
using System;

class Program
{
  static void Main(string[] args)
  {
    float[] mark;
    mark = new float[15];
    float avg, percen, sum = 0;
    byte counter = 0;

    for (int i = 0; i <= 15; i++)
    {
      Console.Write("enter the {0}th mark: ", i + 1);
      mark[i] = float.Parse(Console.ReadLine());
      sum = sum + mark[i];
    }
    avg = sum / 16;
    for (int i = 0; i <= 15; i++)
      if (mark[i] > avg)
        counter++;
    Console.WriteLine(" {0} person(s) > AVG ", counter);
    Console.WriteLine("-------------------");
```

```
            Console.WriteLine();
            percen = (counter / 16f) * 100
            Console.WriteLine("     {0} % > AVG ", percen);
            Console.ReadKey();

        }
    }
```

برنامه ۱ــ۲ــ محاسبه درصد نمرات بالاتر از میانگین کلاس

۴ــ۲ــ حلقه foreach

برای مواقعی که بخواهیم محتوای تمام عناصر آرایه را مورد استفاده قرار دهیم یا آنها را روی صفحه نمایش ببینیم، می‌توان به جای حلقه for ازحلقه foreach استفاده کنیم. دستور foreach روی هر عنصر از داده قابل شمارش تکرار می‌شود. مثلاً هر دو نوع array و string قابل شمارش هستند.

شکل کلی این دستور به صورت زیر است :

foreach (نوع داده نام متغیر in نام آرایه)

دستور ؛

دستور زیر عناصر آرایه score را روی صفحه نمایش می‌دهد.

```
foreach (int  aValue  in  score)
        Console.WriteLine(aValue) ;
```

همان‌طور که مشاهده می‌کنید در هنگام دسترسی به عناصر آرایه از اندیس کمک نمی‌گیریم بلکه با هر بار تکرار حلقه، متغیر aValue نقش یک عنصر از آرایه را به عهده می‌گیرد که با کلمه کلیدی in به آرایه score متصل می‌شود.

مثال ۲ــ۲ : برنامه‌ای بنویسید که نمرات دانش‌آموزان را دریافت کند و تعداد مردودی‌ها را نمایش دهد.

الگوریتم یا روش انجام کار : پس از تعریف آرایه و دریافت نمرات، توسط دستور foreach روی عناصر آرایه حرکت می‌کنیم و شرط مردودی که نمره کمتر از ۱۲ است را بررسی کرده و از متغیر شمارنده کمک می‌گیریم (برنامه ۲ــ۲).

```
using System;

  class MarkOfSudents
  {
    static void Main(string[] args)
    {
      byte counter = 0;
      float[] mark = new float[15];
      for (int i = 0; i < 15; i++)
      {
        Console.Write("Enter the mark of {0}th student :  ", i + 1);
        mark[i] = float.Parse(Console.ReadLine());
      }
      foreach (float m in mark)
        if (m < 12) counter++;
      Console.WriteLine(" - - - - - - - - - - - - - - - - - - - - - - ");
      Console.WriteLine(" the number of failed student(s) is : " + counter);
      Console.ReadKey();
    }
  }
```

برنامه ۲ــ۲ــ دریافت نمرات و شمارش مردودی‌ها

مثال ۳ــ۲ : رشته‌ای را از ورودی دریافت کنید و تک تک کاراکترهای آن را در خطوط جداگانه چاپ کنید.

الگوریتم یا روش انجام کار : با توجه به‌اینکه داده رشته‌ای نیز مانند آرایه از عناصر قابل شمارشی تشکیل شده است، می‌توان با استفاده از دستورforeach به تک تک عناصر آن دسترسی پیدا

کرد. بنابراین رشته‌ای را دریافت می‌کنیم و با استفاده از متغیر نوع کاراکتر به تک تک کاراکترهای رشته دست می‌یابیم (برنامه ۳ــ۲).

```
using System;

  class Testforeach
  {
    static void Main()
    {
      string name;
      Console.Write("Enter a Name:");
      name = Console.ReadLine();
      foreach (char ch in name)
        Console.WriteLine(ch);
    }
  }
```

برنامه ۳ــ۲ــ نمایش کاراکترهای رشته در خطوط جداگانه

ویژگی *Length*

آرایه names را در نظر بگیرید.

```
string[] names = new string[10];
```

طول آرایه names برابر ۱۰ است. در زبان C# می‌توان توسط ویژگی[۱] Length از اندازه یک آرایه یا تعداد عناصر آن مطلع شد.

نام آرایه · Length

مثال ۴ــ۲ : در قسمت آزمون چندگزینه‌ای یک برنامه آموزشی، پاسخ صحیح سؤالات در آرایه‌ای به نام answer قرار دارد، می‌خواهیم پاسخ صحیح سؤالات یا محتوای عناصر این آرایه را نمایش دهیم.

الگوریتم یا روش انجام کار : برای نمایش عناصر آرایه answer از حلقه for استفاده

۱ــ Property

می‌کنیم. شمارنده حلقه باید از صفر شروع و تا آخرین عنصر آرایه که یک واحد کمتر از تعداد عناصر آرایه است پیش رود. بنابراین به تعداد عناصر آرایه نیاز داریم که از ویژگی Length استفاده می‌کنیم :

```
for (int i=0; i<answer. Length ; i++)
    Console. Write (answer [i]);
```

قطعه برنامه ۴ــ۲ــ نمایش پاسخ‌های صحیح آرایه answer

سؤال: به نظر شما این ویژگی در چه برنامه‌هایی ممکن است مورد استفاده قرار بگیرد؟

۵ــ۲ــ تعیین اندازه آرایه در هنگام اجرای برنامه

در مثال‌هایی که تاکنون انجام دادیم اندازه یا تعداد داده‌ها از قبل مشخص بود و اندازه آرایه در متن برنامه مشخص شده بود، اما حالتی را در نظر بگیرید که تعداد داده‌هایی که قرار است کاربر وارد کند قبل از اجرای برنامه نامشخص باشد در این حالت چه باید کرد؟ مثلاً می‌خواهیم برنامه‌ای بنویسیم که میانگین نمرات دانش‌آموزان کلاس را محاسبه کند و برای هر کلاس، با تعداد دانش‌آموزان مختلف، قابل استفاده باشد.

روش اول : می‌توانیم یک آرایه با تعداد عناصر زیاد مثلاً ۵۰ عنصر را پیش‌بینی کنیم و داده‌ها را از کاربر دریافت و در آرایه قرار دهیم و پس از ورود آخرین نمره، کاربر یک عدد خاص مثلاً عدد ۱- را برای پایان دادن به ورودی، دریافت کند و در این صورت حلقهٔ دریافت نمرات پایان یابد. در این روش با وارد کردن هر نمره نیز یک شمارنده را افزایش می‌دهیم تا تعداد نمرات را در اختیار داشته باشیم و در این صورت، ویژگی Length دیگر در حلقه کاربرد ندارد. چرا؟ در این روش همواره تعدادی از عناصر آرایه خالی و بدون استفاده هستند.

روش دوم : ابتدا تعداد نمرات را از کاربر سؤال کنیم و سپس آرایه‌ای به همان میزان ایجاد کرده و با استفاده از حلقه for مانند مثال‌های قبلی اقدام به دریافت نمرات کرده و از ویژگی Length نیز به عنوان شرط انتهای حلقه استفاده کنیم.

مثال ۵ــ۲ : می‌خواهیم برنامه‌ای بنویسیم که میانگین نمرات دانش‌آموزان یک کلاس با هر تعداد دانش‌آموز را محاسبه کند.

الگوریتم یا روش انجام کار : برای نوشتن چنین برنامه‌ای از روش دوم که توضیح داده شد استفاده می‌کنیم. در برنامه زیر به نحوه تعریف آرایه list توجه کنید.

```
using System;

class StudentList
{
    static void Main()
    {
      string input;
      int listSize;
      Console. Write ("Enter number of the students: ");
      input = Console. ReadLine ();
      listSize = int.Parse (input);

      float [] list = new float [listsize];  ⇦
      float total = 0;

      for (int i = 0; i< list. Length ; i++)
      {
          Console. Write ("Enter a mark: ");
          input = Console. ReadLine();
          list[i] = float. Parse (input);
          total + = list [i];
      }
      Console. WriteLine ("Average of the marks: {0}", total / list.
Length);
    }
}
```

برنامه ۵ــ۲ــ محاسبه میانگین نمرات با تعداد نامشخص

۶ ــ ۲ــ مرتب کردن داده‌های یک لیست

الگوریتم‌های مختلفی برای مرتب کردن عناصر لیست وجود دارد بعضی از آنها خیلی ساده و ابتدایی هستند مانند روش مرتب‌سازی حبابی[۱] که زمان زیادی طول می‌کشد تا لیست مرتب شود و برخی از روش‌ها وجود دارند مانند روش مرتب‌سازی سریع[۲] که سرعت بسیار بالایی دارد.

مثال ۶ــ۲ : نمرات دانش‌آموزان یک کلاس را دریافت نموده و به صورت صعودی مرتب کنید. سپس رتبه‌های اول تا سوم را اعلام کنید.

الگوریتم یا روش انجام کار : فرض کنید می‌خواهیم لیستی از نمرات دانش‌آموزان را که به صورت زیر است مرتب کنیم :

۱۷, ۱۴, ۸/۵, ۱۳, ۲۰

بعد از مرتب کردن، لیست به صورت زیر خواهد شد :

۸/۵, ۱۳, ۱۴, ۱۷, ۲۰

در این روش از ابتدای لیست شروع کرده و دو عنصر اول و دوم را با یکدیگر مقایسه می‌کنیم. اگر ترتیب آنها درست نبود، آنها را جابه‌جا می‌کنیم. سپس به سراغ عنصر دوم و سوم می‌رویم و عمل مقایسه و در صورت لزوم جابه‌جایی را انجام می‌دهیم. و تا آخر لیست به همین ترتیب جلو می‌رویم. در الگوریتم مرتب سازی همواره دو عمل «مقایسه»[۳] و «جابه جایی»[۴] انجام می‌شود.

سؤال: وقتی که به انتهای لیست می‌رسیم چه اتفاقی می‌افتد؟

اکنون اعداد مورد نظر را به روش حبابی و به‌ترتیب صعودی یعنی از کوچک به بزرگ مرتب می‌کنیم :

۱ــ Bubble Sort Method

۲ــ Quick Sort Method

۳ــ Compare

۴ــ Swap

شکل ۲ــ۲ــ مراحل مرتب سازی حبابی، فاز اول

توجه کنید که پس از انجام این تعداد مقایسه و جابه‌جایی، بزرگ‌ترین عددی که در لیست وجود دارد خود را به جای اصلی خود، یعنی در انتهای لیست می‌رساند.

سؤال: چرا به‌این روشِ مرتب‌سازی، حبابی گفته می‌شود؟

هنوز لیست به طور کامل مرتب نشده و باید مجدداً از ابتدای لیست شروع کرده و تا انتهای لیست عمل مقایسه و جا به‌جایی را انجام دهیم تا عدد بزرگ‌تر بعدی نیز، خود را به مکان صحیح برساند. در فاز دوم، عددِ بزرگِ دوم، به مکانِ یکی مانده به انتهای لیست منتقل می‌شود.

بنابراین باید، عملیات مقایسه و جابه‌جایی را بارها تکرار کرد تا در نهایت، لیست به طور کامل مرتب شود.

سؤال: در مثال ذکرشده چند فاز باید طی شود تا اعداد در جای خود قرار گیرند؟ مراحل بعدی را مانند نمونه بنویسید.

توجه داشته باشید که عمل مرتب سازی یک عمل وقت‌گیر است به‌خصوص روش حبابی که یک روش کند است، چون تعداد عمل مقایسه و جا به جایی در آن بسیار زیاد است.

سؤال: در مرتب‌سازی حبابی، وقت‌گیرترین و بدترین حالت آرایشِ اعدادِ ورودی کدام است؟

اکنون سعی می‌کنیم الگوریتم را پیاده‌سازی کنیم. همان‌طور که در قطعه برنامه زیر مشاهده می‌کنید در فاز اول ۴ مرحله مقایسه و جابه جایی داریم. چرا؟ برای این که مراحل بالا را پیاده سازی کنیم به حلقه‌ای نیاز داریم که در هر بار اجرای حلقه، عدد زم را با عدد ۱+زم مقایسه کند و در صورتی که بزرگ‌تر بود، جابه‌جایی انجام دهد. دستورات جابه‌جایی با استفاده از متغیر کمکیِ سوم را در سال گذشته تمرین کرده‌اید که یکی از کاربردهای آن در مرتب سازی است. اکنون دستور مربوط به‌این بخش را می‌نویسیم :

```
for (int j = 0;      ; j++)
    if (arr[j] > arr[j + 1])
    {
        temp = arr [j];
        arr[j] = arr[j + 1];
        arr[j + 1] = temp;
    }
```

قسمت شرط پایانی حلقه را فعلاً ننوشته‌ایم تا به توافق برسیم. اگر در پاسخ به سؤال بالا، مراحل مقایسه و جابه جایی را در فاز دوم و سوم و ... نوشته باشید، متوجه شده‌اید که حلقهٔ بالایی باید به تعداد فازهای مرتب‌سازی تکرار شود. بنابراین باید حلقهٔ بالا را درون حلقهٔ دیگری قرار دهیم که مشخص‌کنندهٔ فازهای تکرار است.

سؤال: چرا تعداد تکرار حلقهٔ بیرونی که معرف فازهای مرتب سازی است، یکی کمتر از تعدادِ اعداد است؟

```
for (int i = 4; i > 0 ; i--)
    for (int j = 0;      ; j ++)
        if (arr [j] > arr[j + 1])
        {
            temp = arr [j];
            arr[j] = arr[j + 1];
            arr[j + 1] = temp;
        }
```

اکنون نوشتن حلقه‌های مرتب‌سازی حبابی تمام شده است اما هنوز نمی‌دانیم حلقهٔ داخلی که نشان دهندهٔ مراحل در هر فاز است، چه شرطی برای پایان دارد. اگر به واژهٔ حبابی توجه کنیم در می‌یابیم که در پایان هر فاز، یک عنصر در مکان صحیح خود در لیست جای می‌گیرد. پس دیگر نیازی به مقایسهٔ این عنصر نداریم زیرا به درستی در مکان خود قرار گرفته و مرتب شده است. بنابراین در فاز بعدی یک مرحله کمتر از فاز قبل، مقایسه و جابه جایی داریم که با شرط j<i می‌نویسیم. برنامه ۶ــ۲ را ببینید و با هم کلاس خود برای اعداد قبلی Trace کنید. (برنامه ۶ــ۲)

سؤال: دستوراتی به برنامه اضافه نمایید تا رتبه‌های اول تا سوم را چاپ نماید.

```
using System;

class BubbleSort
{
   static void Main(string[] args)
   {
      Console.Write("enter the array size: ");
      int arrsize = int.Parse(Console.ReadLine());

      int[] arr;
      arr = new int[arrsize];

      for (int i = 0; i < arrsize; i++)
      {
         Console.Write("enter the {0}th number: ", i+1);
         arr[i] = int.Parse(Console.ReadLine());
      }

      //Bubble Sort.....................................

      int temp = 0;
      for (int i = arrsize - 1; i > 0; i--)
      {
         for (int j = 0; j < i; j++)
            if (arr[j] > arr[j + 1])
            {
               temp = arr[j];
               arr[j] = arr[j + 1];
               arr[j + 1] = temp;
            }
      }

      // show  the sorted array
      Console.WriteLine();
      Console.WriteLine("  the sorted array is:    ");
      Console.WriteLine("-------------------------");
      for (int i = 0; i < arrsize; i++)
         Console.WriteLine("arr[{0}] = {1}", i+1, arr[i]);
   }
}
```

مقایسه

و

جابجایی

برنامه ۶ــ۲ــ مرتب‌سازی حبابی

۷ــ ۲ــ عمل جستجو در لیست

جستجو پرکاربردترین عملیاتی است که روی لیست‌ها انجام می‌شود و به منظور یافتن عنصری در آرایه صورت می‌گیرد که به آن کلید جستجو گفته می‌شود. در این کتاب ما به توضیح و برنامه نویسی دو نوع جستجو می‌پردازیم.

۱ــ۷ــ۲ــ جستجوی خطی(ترتیبی)[۱] :

ساده‌ترین روش جستجو، به‌صورت ترتیبی می‌باشد که در آرایه‌های کوچک و یا مرتب نشده مورد استفاده قرار می‌گیرد. در این روش جستجو، هر عنصر از آرایه با کلید جستجو مقایسه می‌شود. و برای تعیین اینکه کلید مورد نظر در آرایه وجود دارد یا نه، برنامه باید کلید را با تمام عناصر آرایه مقایسه کند. بنابراین در صورتی که کلید یافت شود نیازی به ادامهٔ جستجو نیست و می‌توان از برنامه خارج شد.

مثال ۷ــ۲ : برنامه‌ای بنویسید که نمره ۲۰ را از میان لیست نمرات کلاس پیدا کرده و اعلام کند مربوط به نفر چندم لیست است؟

الگوریتم یا روش انجام کار : ابتدا آرایه‌ای از نمرات دانش‌آموزان تعریف و به‌وسیلهٔ حلقه for نمرات را دریافت می‌کنیم. سپس برای جستجوی ترتیبی، با استفاده از حلقهٔ for تک تک عناصر را با کلید جستجو مقایسه کرده و در صورت یافتن کلید به کار حلقه پایان می‌دهیم. برای یافتن کلید از متغیر بولی found کمک می‌گیریم. این متغیر مقدار اولیه false دارد تا زمانی که کلید پیدا شود، در صورتی که کلید یافت نشود، هم چنان مقدار false خواهد داشت.[۲]

۱ــ Liner (Sequential) Search

۲ــ به چنین متغیرهایی که وضعیت دوحالتی خاموش یا روشن را نشان می‌دهند Flag گفته می‌شود. استفاده از تکنیک Flag در برنامه‌های زیادی کاربرد دارد.

```
using System;
  class Search
  {
    static void Main(string[] args)
    {
      Console.Write("enter the array size: ");
      int arrsize = int.Parse(Console.ReadLine());
      Console.WriteLine("-----------------");
      int[] arr;
      arr = new int[arrsize];
      //.....Initializing array.....
      for (int i = 0; i < arrsize; i++)
      {
        Console.Write("enter the {0}th number: ", i+1);
        arr[i] = int.Parse(Console.ReadLine());
      }
      //.....Linear Search.....
      bool found = false;
      int pos=-1;
      Console.WriteLine("-----------------");
      Console.Write("enter the search key: ");
      int key = int.Parse(Console.ReadLine());
      Console.WriteLine("= = = = = = = = =");
      for (int i = 0; i < arrsize; i++)
        if (arr[i] == key)
        {
          found = true;
          pos = i+1;
          break;
        }
      if (found)
        Console.WriteLine("the key found in  position number {0}  ", pos);
      else
        Console.WriteLine("the key not found");
      Console.ReadKey();
    }
  }
```

۲ــ۷ــ۲ــ جستجوی دودویی(باینری)[۱] :

روش دیگر جستجوی کلید در بین عناصر آرایه، روش جستجوی دودویی است. فرض کنید می‌خواهید شماره پلاک یک خانه را در یک کوچه بیابید، آنچه باعث می‌شود سریع‌تر خانهٔ مورد نظر را پیدا کنید، مرتب بودن شماره پلاک‌ها در کوچهٔ مورد نظر است.

سؤال: اگر در یک کوچهٔ طولانی دنبال یک خانه بودید و این شماره‌ها مرتب نبود، چه می‌کردید؟

روش جستجوی دودویی، روی آرایه‌های مرتب قابل اجراست. برای استفاده از روش جستجوی دودویی، در صورتی که آرایه مورد نظر نامرتب باشد، ابتدا باید آن را مرتب کنید. در مثال‌های قبل با روش مرتب سازی حبابی آشنا شدید. یکی از کاربردهای مرتب سازی در جستجوی دودویی است. این روش جستجو برای آرایه‌های بزرگ مناسب است. چرا؟

سؤال: به طور کلی چرا از این نوع جستجو استفاده می‌کنیم؟

مثال ۸ــ۲ : برنامه‌ای بنویسید که با دریافت کلید جستجو به روش دودویی آن را جستجو کنید.

الگوریتم یا روش انجام کار : اگر بازی حدس عدد را در کتاب برنامه سازی۱ به خاطر بیاورید، برای حدسِ عددی که بازیکن اول در ذهن سپرده بود، از روش حرکت کردن در لیست اعداد استفاده می‌کردیم. در ذهن فرد حدس زننده این بود که اعداد مرتب هستند. بنابراین به محض حدس اولین عدد منتظر نتیجه از بازیکن اول می‌ماند تا بگوید عدد حدس زده شده از کلید جستجو بزرگ‌تر است یا کوچک‌تر.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
8.5	9	10	11	12.75	14	14.5	15	16	16.5	17	17.5	18	19.5	19.75	20

مقایسه داده مورد نظر با عنصر وسط

شکل ۳ ــ۲ــ لیست اعداد مرتب به همراه شماره عنصر

به‌این ترتیب در بازی حدس عدد از تکنیک مشابه جستجوی دودویی استفاده می‌شود. در این

۱ــ Binary

جستجو برای یافتن کلید، ابتدا عنصر وسط لیست با کلید مقایسه می‌شود. این مقایسه ۳ نتیجه در بردارد :

۱ــ کلید با عنصر وسط لیست یکسان است. بنابراین کلید پیدا شده و در این‌صورت شماره یا اندیس عنصر وسط آرایه را یادداشت می‌کنیم.

۲ــ عنصر وسط لیست، کوچک‌تر از کلید است، بنابراین نتیجه می‌گیریم که داده در نیمه سمت چپ نمی‌باشد. در نتیجه باید منطقه مورد جستجو را به نیمه سمت راست آرایه محدود کنیم.

۳ــ عنصر وسط لیست، بزرگ‌تر از کلید است، بنابراین نتیجه می‌گیریم که داده در نیمه سمت راست لیست نمی‌باشد. در نتیجه باید منطقه مورد جستجو را به نیمه سمت چپ آرایه محدود کنیم.

پس از بررسی مشخص کردن منطقه جدید، مجدداً عمل جستجو را تکرار می‌کنیم یعنی عنصر وسط لیست جدید را مورد بررسی قرار می‌دهیم و یکی از ۳ حالت بالا رخ خواهد داد که بر آن اساس یا داده یافت می‌شود و یا مجدداً عمل جستجو در لیست جدیدی که نصفِ اندازهٔ قبلی است، تکرار می‌شود.

در روش جستجوی دودویی با یک عمل مقایسه، منطقه مورد جستجو به‌اندازه نصف، کوچک می‌شود. بنابراین با چند عمل مقایسه یک لیست بزرگ از داده‌ها به منطقه کوچکی ختم می‌شود بنابراین سرعت عملیات جستجو در این روش بسیار بالا است.

برای مثال اگر لیستی دارای ۱۰ هزار عنصر باشد در اولین مرحله ۵۰۰۰ عنصر کنار می‌روند. در مرحله دوم ۲۵۰۰ عنصر و در مرحله سوم ۱۲۵۰ عنصر کنار می‌روند. بنابراین خواهیم داشت :

مرحله جستجو	تعداد عناصر لیست	مرحله جستجو	تعداد عناصر لیست
۱	۱۰۰۰۰	۲	۵۰۰۰
۳	۲۵۰۰	۴	۱۲۵۰
۵	۶۲۵	۶	۳۱۳
۷	۱۵۷	۸	۷۹
۹	۴۰	۱۰	۲۰
۱۱	۱۰	۱۲	۵
۱۳	۳		

در این جستجو، با حداکثر ۱۳ بار مقایسه، کلید را پیدا می‌کنیم و یا مطمئن می‌شویم که در لیست، چنین داده‌ای وجود ندارد. جستجوی دودویی در مقایسه با روش جستجوی ترتیبی دارای سرعت فوق العاده بالایی است. چرا؟

سؤال: کلید جستجو، در لیست ۱۰۰۰۰عنصری مثال قبل در روش ترتیبی در چند مرحله جستجو می‌شود؟

نکته

توجه داشته باشید از روش جستجوی دودویی تنها زمانی می‌توانید استفاده کنید که عناصر لیست مرتب باشند. بنابراین اگر لیست مرتب نباشد و بخواهید در لیست بزرگی جستجو کنید، بهتر است ابتدا لیست را مرتب کنید و سپس به روش دودویی جستجو نمایید.

برای اینکه بتوانیم هر بار لیست را نصف کنیم، اندیس‌های آرایه را به سمت منطقه مورد نظر هدایت می‌کنیم. برای این منظور باید وسط لیست را پیدا کنیم و عملیات جابه‌جایی اندیس داشته باشیم.

```
middle = (first + last) / 2;   محاسبهٔ اندیس وسط لیست
last = middle −1;   عملیات جابه جایی اندیس زمانی که منطقه جستجو سمت چپ باشد
first = middle +1;   عملیات جابه جایی اندیس، زمانی که منطقه جستجو سمت راست باشد
```

برنامه جستجو را مشاهده کنید. این برنامه برای داده‌های مرتب شده کفایت می‌کند، اما همیشه برای احتیاط بهتر است قبل از جستجوی دودویی داده‌های وارد شده را مرتب کنید. چرا؟

```
using System;
  class BinSearch
  {
    static void Main(string[] args)
    {
      Console.Write("enter the array size: ");
      int arrsize = int.Parse(Console.ReadLine());
      Console.WriteLine("-----------------");
      int[] arr;
      arr = new int[arrsize];
      //.....Initializing arr ay.....
      for (int i = 0; i < arrsize; i++)
      {
        Console.Write("enter the {0}th number: ", i+1);
        arr[i] = int.Parse(Console.ReadLine());
      }
      //.....Binary Search.....
      int last = arrsize - 1;
      int first = 0;
      int middle = (first + last) / 2;
      bool found = false;
      Console.WriteLine("-----------------");
      Console.Write("enter the search key: ");
      int searchkey = int.Parse(Console.ReadLine());
      Console.WriteLine("= = = = = = = = =");
      while (last >= first)
      {
        middle = (first + last) / 2;
        if (arr[middle] == searchkey)
        {
          found = true;
          break;
        }
        else if (arr[middle] > searchkey)
          last = middle - 1;
        else
          first = middle + 1;
      }
      if (found)
        Console.WriteLine("the key found in  position number= {0} ", middle);
else
        Console.WriteLine("the search key not found");
      Console.ReadKey();
    }
    }
```

برنامه ۸ــ۲ــ درصورتی قابل استفاده است که داده‌ها، مرتب شده وارد شوند. در ورود داده‌ها دقت کنید.

خودآزمایی فصل دوم

الف) درستی یا نادرستی عبارات زیر را تعیین کنید.

۱ــ جستجوی دودویی فقط بر روی لیست‌های مرتب امکان پذیر است.

۲ــ جستجوی خطی برای آرایه‌های بزرگ مناسب است.

۳ــ دستور foreach فقط بر روی آرایه‌ها قابل استفاده است.

۴ــ اندیس اولین عنصر آرایه، ۱ است.

۵ــ از آرایه زمانی استفاده می‌کنیم که مجبوریم چندین بار از داده‌های ورودی در برنامه استفاده کنیم.

ب) جاهای خالی را با عبارات مناسب پرکنید.

۶ــ در عمل جستجو در آرایه به مقداری که جستجو می‌شود..... می‌گویند.

۷ــ در هر الگوریتم مرتب‌سازی همیشه دو عمل..... و ..... انجام می‌شود.

۸ــ تعداد عناصر آرایه را ..... آرایه می‌نامند.

۹ــ به هر مکان آرایه، یک ..... آرایه می‌گویند.

ج) به سوالات زیر پاسخ دهید.

۱۰ــ آرایه‌ای برای نام روزهای هفته تعریف کنید.

۱۱ــ آرایه‌ای برای نام ماه‌های سال تعریف کنید.

۱۲ــ برنامه زیر را Trace کنید و در یک خط بنویسید چه عملی انجام می‌دهد.

```
class Program
{
static void Main(string[] args)
{
int[] b = new int[5];
int[] a = new int[5];
int[] c = new int[5];
for (int i = 0; i <= 4; i++)
```

```
{
    a[i] = int.Parse(Console.ReadLine());
    b[i] = int.Parse(Console.ReadLine());
    if (a[i] > b[i])
        c[i] = a[i];
else
        c[i] = b[i];
    Console.WriteLine(c[i]);
}
  Console.ReadKey();
{
```

۱۳ــ آرایه زیر را در نظر بگیرید و معادل دستورات زیر را با روش دیگری از تعریف آرایه بازنویسی کنید.

```
string[] week = new string[7];
week[0] = "Saturday";
week[1] = "Sunday";
week[2] = "Monday";
week[3] = "Tuesday";
week[4] = "Wednesday";
week[5] = "Thursday";
week[6] = "Friday";
```

۱۴ــ دستوری بنویسید که تعداد عناصر آرایه week را نمایش دهد.

۱۵ــ برنامه زیر را Trace کنید و خروجی آن را بنویسید.

```
namespace arrayTest
{
  class Program
  {
```

```
        static void Main(string[] args)
        {

            int[] arr = { 10, 13, 15, 6, 8, 20 };
            arr[1] = 3;
            for (int i = 0; i < 6; i++)
            {
                arr[i] += 2;
                Console.WriteLine(arr[i]);
            }
            Console.ReadKey();

        }
    }
}
```

۱۶ــ با توجه به اعلان آرایه arr به سؤالات زیر پاسخ دهید.

```
int[] arr = { 10, 13, 15, 6, 8, 20 };
```

دستوری بنویسید که مقدار سومین عنصر آرایه را صفر کند.

آرایه arr چقدر حافظه اشغال می‌کند؟

نتیجه اجرای قطعه کد زیر چیست؟

```
int temp = arr[0];
arr[0] = arr[5];
arr[5] = temp;
```

۱۷ــ کدام یک از روش‌های جستجو سریع‌تر است؟ چرا؟

۱۸ــ در یک آرایه مرتب با ۱۰ عنصر حداکثر با چند مقایسه نتیجه جستجو مشخص می‌شود؟

تمرینات برنامه‌نویسی فصل دوم

۱ــ با استفاده از حلقه foreach نام روزهای هفته را چاپ کنید.

۲ــ برنامه‌ای بنویسید که نمرات درس برنامه‌سازی یک کلاس ۱۵ نفره را دریافت کند و بیشترین و کمترین نمره کلاس را پیدا کرده و نمایش دهد.

۳ــ برنامه قبل را توسعه دهید تا علاوه بر بیشترین و کمترین نمره کلاس، اعلام کند چندمین دانش‌آموز بیشترین یا کمترین نمره را دارد.

۴ــ شرکت مخابرات می‌خواهد به مشتریانی که در طول یک سال، حداقل ۶ ماه، قبض تلفن همراه خود را قبل از بیستم ماه پرداخت می‌کنند به عنوان جایزه یک بسته مکالمه رایگان ۶۰۰ دقیقه‌ای هدیه دهد. برنامه‌ای بنویسد که برای یک مشتری روز پرداخت قبض در هر ماه را دریافت و اعلام کند که بسته مکالمه رایگان به مشتری تعلق می‌گیرد یا نه؟

۵ــ صاحب یک کارگاه آهنگری می‌خواهد به کارگرانی که بیش از ۳ فرزند دارند، مبلغ ۳۰۰۰۰۰ تومان مساعده پرداخت کند. برنامه‌ای بنویسید که تعداد فرزندان ۱۸ کارگر این کارگاه را دریافت کرده و تعداد کارگرانی که بیش از ۳ فرزند دارند را محاسبه کند و نمایش دهد. سپس کل مبلغ مساعده را نمایش دهد.

۶ــ برنامه قبل را توسعه دهید تا برای هر کارگاه با هر تعداد کارگر و هر مبلغ مساعده قابل استفاده باشد.

۷ــ برنامه سؤال ۵ را با foreach بازنویسی کنید.

۸ــ برنامه ۲ را توسعه دهید تا نام و نمره درس برنامه‌سازی یک کلاس ۱۵ نفره را دریافت کند. سپس نام دانش‌آموزانی که کمترین و بیشترین نمره را دارند به همراه نمره آنها نمایش دهد.

۹ــ یک شرکت خصوصی ۳درصد حقوق کارمندان خود را به عنوان مالیات از آنها کسر می‌کند. برنامه‌ای بنویسید که میزان حقوق ۳۵ کارمند این شرکت را دریافت کند. سپس میزان خالص دریافتی هر کارمند را بعد از کسر مالیات محاسبه و مجدداً در آرایه ذخیره کند.

فعالیت

متن زیر از MSDN با موضوع آرایه برداشت شده است. آن را با کمک هم کلاسی خود ترجمه کنید و به کلاس ارائه نمایید.

Arrays in General

C# arrays are zero indexed; that is, the array indexes start at zero. Arrays in C# work similarly to how arrays work in most other popular languages There are, however, a few differences that you should be aware of.

When declaring an array, the square brackets ([]) must come after the type, not the identifier. Placing the brackets after the identifier is not legal syntax in C#.

```
int[] table; // not int table[];
```

Declaring Arrays

C# supports single-dimensional arrays, multidimensional arrays (rectangular arrays) The following example shows how to declare single-dimensional array:

Single-dimensional arrays:

```
int[] numbers;
```

Declaring them (as shown above) does not actually create the arrays. In C#, arrays are objects and must be instantiated. The following examples show how to create arrays:

Single – dimensional arrays:

```
int[] numbers = new int[5];
```

واژگان و اصطلاحات انگلیسی فصل دوم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Array	
۲	Binary	
۳	Bubble Sort Method	
۴	Compare	
۵	Element	
۶	Index	
۷	Length	
۸	Liner Search	
۹	Sequeutial Search	
۱۰	Property	
۱۱	Quick Sort Method	
۱۲	Size	
۱۳	Swap	

فصل سوم

داده شمارشی، کلاس و متد

در این فصل علاوه بر آشنایی با نوع داده شمارشی، با مفاهیم متد و کلاس که از مفاهیم پایه برنامه‌سازی شئ‌گرا هستند به صورت مقدماتی آشنا خواهید شد. هدف از این فصل، نوشتن متد یا ایجاد یک کلاس نیست. شناخت اجزای یک کلاس و متد کمک می‌کند در کاربرد آنها بیشتر به تعاریف شان دقت کنید و آنها را درست به کار گیرید. بدیهی است ایجاد کلاس در کتاب برنامه سازی۳ مورد توجه خاص قرار می‌گیرد.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ نوع داده شمارشی را توضیح دهد.
۲ــ از داده شمارشی در برنامه‌های خود استفاده کند.
۳ــ متد را تعریف کند و کاربرد آن را بیان نماید.
۴ــ از برخی متدهای آماده در برنامه‌های خود استفاده کند.
۵ــ کلاس، شئ و کاربرد آنها را توضیح دهد.
۶ــ شکل کلی یک کلاس و اجزای یک کلاس را نام برده و هر یک را توضیح دهد.
۷ــ کلاس‌های آماده سی‌شارپ را در برنامه‌های خود به کار بندد.

۱ــ۳ــ نوع داده شمارشی[۱]

تاکنون با داده‌های ساده کار کرده‌اید. داده شمارشی از داده‌های مرکب است که خود شامل یک سری داده است. در شکل ۱ــ۳ قسمتی از دو برنامه نشان داده شده است که در هر یک از آنها دستور switch استفاده شده است. اگر فرض کنیم هر کدام از برنامه‌ها صحیح کار می‌کنند و کارکرد یکسانی دارند شما کدام‌یک را ترجیح می‌دهید؟

```
switch (player)
{
    case 1:
        ⋮
    case 2:
        ⋮
    case 1:
        ⋮
    case 1:
        ⋮
}
```
قطعه برنامه الف

```
switch (player)
{
    case playerRank. Leader:
        ⋮
    case PlayerRank. Co–Leade:
        ⋮
    case PlayerRank. Elder:
        ⋮
    case PlayerRank. Member:
        ⋮
}
```
قطعه برنامه ب

شکل ۱ــ۳ ــ ساختار switch

همان طور که در شکل ۱ــ۳ مشاهده می‌کنید با نگاه به قطعه برنامه (الف) متوجه می‌شویم که مقدار متغیر player با اعداد ۱، ۲، ۳ و۴ مقایسه می‌شود. اما این سؤال پیش می‌آید که این اعداد، مربوط به چه موضوعی هستند؟ آیا شماره بازیکن است ؟ آیا فرصت بازی بازیکن است؟ و . . .

اما وقتی به قطعه برنامه (ب) نگاه می‌کنیم برداشت دقیق‌تری نسبت به برنامه داریم. مقدار متغیر

۱ــ Enumerated Type

player با مقادیر Leader ، Co_Leader ، Elder ، Member که چهار سطح و مقام در بازی[1] است، مقایسه می‌شود.

خوانایی برنامه (ب) بالاتر از برنامه (الف) است، به‌دلیل اینکه به جای اعداد ثابت و بی معنی، از کلمات و نام‌های با معنی استفاده شده است. در فصل اول کتاب با مفهوم و کاربرد ثابت‌ها در برنامه آشنا شدید و مشاهده کردید که چگونه خوانایی برنامه را بالا می‌برند. در این بخش نیز با نوع داده شمارشی آشنا می‌شوید که سبب بالابردن خوانایی برنامه می‌شوند.

نوع داده شمارشی مجموعه‌ای از چند نام دلخواه می‌باشد که حالت‌ها و مقادیر مختلف یک موضوع را نشان می‌دهد. مثلاً برای روزهای هفته به جای اعداد ۱ تا ۷ (یا ۰ تا ۶) از نام‌ها و کلمات معنی دار استفاده می‌کنیم. علاوه بر روزهای هفته، برای نام ماه‌های سال، مقام یا درجه یک بازیکن و مدرک تحصیلی اشخاص نیز می‌توان از نوع داده شمارشی استفاده کرد.

برای تعریف یک نوع داده شمارشی از کلمه کلیدی enum به صورت زیر استفاده می‌شود. نوع دسترسی معمولاً public است و روش نوشتن نام نوع داده نیز مطابق با روش پاسکال است.

محل قرارگیری تعریف نوع داده شمارشی، معمولاً خارج از کلاس و در ابتدای برنامه است.

```
نام دلخواه enum نوع دسترسی
{
        لیستی از نام‌ها و کلمات
}
```

نکته

در لیست نام‌ها و کلمات در نوع داده شمارشی، هر نام با علامت کاما از نام دیگر جدا می‌شود. نقطه ویرگول در این تعریف استفاده نمی‌شود.

در تعریف صفحه بعد نوع داده شمارشی برای نام روزهای هفته میلادی با دسترسی وسیع را مشاهده می‌کنید.

۱ــ بازی CoC

```
public enum DayOfWeek
{
    Sunday,
    Monday,
    Tuesday,
    Wednesday,
    Thursday,
    Friday,
    Saturday
}
```

هر یک از اعضای نوع داده شمارشی معادل با یک عدد ثابت است، این اعداد به طور پیش فرض از عدد صفر شروع می‌شوند و به ترتیب، یک واحد اضافه می‌شوند. مثلاً در تعریف قبل نام Sunday معادل با عدد صفر و نام Monday برابر یک و Saturday برابر ۶ است. اگر مایل باشید می‌توانید عدد دیگری را برای نام‌ها اختصاص دهید. در تعریف زیر نوع داده شمارشی برای ماه‌های یک سال میلادی را مشاهده می‌کنید. در این تعریف January معادل با عدد یک و February برابر دو است.

```
public enum MonthOfYear
{
    January = 1,
    February,
    March,
    April,
    May,
    June,
    July,
    August,
    September,
    October,
    November,
    December
}
```

سؤال: در قطعه برنامه بالا نام December معادل با چه عددی است؟

همان‌طور که با نوع داده‌های ساده و ابتدایی مانند int یا float کار می‌کنیم و می‌توانیم اعداد ثابت و یا متغیری از آن نوع را در برنامه مورد استفاده قرار دهیم دقیقاً همان عملیات را می‌توانیم با نوع داده شمارشی البته با کمی‌محدودیت انجام دهیم. در این قسمت با ذکر مثال با آنها آشنا می‌شویم.

۱ــ۱ــ۳ــ دسترسی به اعضای یک نوع داده شمارشی :

با نوشتن نام نوع داده شمارشی و سپس نام عضو که این دو با علامت نقطه از هم جدا می‌شوند، می‌توانیم به اعضا یا مقادیر یک نوع داده شمارشی دسترسی پیدا کنیم.

```
نام عضو . نوع داده شمارشی
   ↓          ↓
DayOfWeek . Saturday
```

شکل ۲ ــ۳ــ دسترسی به یک عضو نوع داده شمارشی

۲ــ۱ــ۳ــ تعریف یک متغیر از نوع داده شمارشی :

مانند متغیرهای ساده، متغیر نوع شمارشی قادر است فقط یکی از مقادیر نوع شمارشی را در خود جای دهد و به صورت شکل ۳ــ۳ تعریف می‌شود. اصول نام گذاری متغیرها را رعایت کرده و طبق روش کوهان شتری نام متغیر را می‌نویسیم.

```
; نام متغیر    نوع داده شمارشی
                  ↓          ↓
              DayOfWeek    holiday;
```

شکل ۳ــ۳ــ تعریف یک متغیر نوع داده شمارشی

۳ــ۱ــ۳ــ مقداردهی متغیرهای شمارشی :

مقداردهی متغیرهای نوع شمارشی معمولاً از طریق دستور انتساب به صورت شکل ۴ــ۳ انجام می‌شود.

```
; مقدار  =  نام متغیر
holiday = DayOfWeek . Sunday;
```

شکل ۴ــ۳ــ مقداردهی به یک متغیر نوع شمارشی

نکته

توجه داشته باشید مقداری که در متغیر نوع شمارشی قرار می‌گیرد باید با نوع آن مطابقت داشته باشد. مثلاً ذخیره عدد ۳ در متغیر holiday به صورت مستقیم امکان پذیر نیست و حتماً باید از تبدیل نوع استفاده کنید تا عضوی مطابق عدد ۳ در آن ذخیره گردد.

۴ــ۱ــ۳ــ نمایش مقدار یک عضو یا محتوای یک متغیر نوع شمارشی :

مانند متغیرهای ساده، از متد WriteLine() برای نمایش محتوای متغیری از نوع داده شمارشی و یا یک عضو از آن می‌توان استفاده کرد. شکل ۵ــ۳ را مشاهده کنید.

```
Console. writeLine ("My holiday is " + holiday);
Console. writeLine ("We should go to school on {0}", Day Of Week. Saturday);
```

نام عضو . نوع داده شمارشی

شکل ۵ــ۳ــ نمایش مقدار یک نوع داده شمارشی

کار در کارگاه ۱

مثال ۱ــ۳ : برنامه‌ای بنویسید که گروه سنی کاربر را با توجه به سن او طبق جدول ۱ــ۳ مشخص کند.

جدول ۱ــ۳ــ گروه سنی

محدوده سنی	نام
بالای ۱۸ سال	بزرگسالان (adult)
بالای ۱۵ سال	نیمه دوم نوجوانی(late teens)
بالای ۱۲ سال	ابتدای دوران نوجوانی (early teens)
بالای ۷ سال	کودکی(Children)
بالای ۳ سال	خردسالی(Infancy)

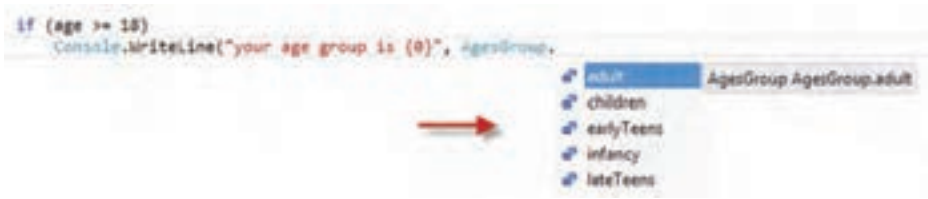
الگوریتم یا روش انجام کار : در این برنامه باید ابتدا سن کاربر سؤال شود و سپس با استفاده از دستورات if تودرتو، شرط‌های جدول یک به یک بررسی شود. اگر یک شرط برقرار شد گروه سنی نمایش داده شود. اگر کمی فکر کنیم می‌بینیم خواسته این برنامه، موضوع جدیدی نیست زیرا شبیه‌این کار را برای نمره یک دانش‌آموز قبلاً انجام داده‌اید. اما در این برنامه از نوع داده شمارشی استفاده می‌کنیم تا با مفهوم این نوع داده و روش استفاده از آن در یک برنامه آشنا شویم.

```
using System;
public enum AgesGroup
{
    adult, earlyTeens, late Teens, children, infancy
}
        class userAges Group
        {
            static void Main (string[] args)
            {
                int age = int. Parse (console. Readline());
                if (age > = 18)
                    Console. WriteLine ("your age group is {0}", AgesGroup. adult);
                else if (age > = 15)
                    Console. WriteLine ("your age group is {0}", AgesGroup. lateTeens);
                else if (age > = 12)
                    Console. WriteLine ("your age group is {0}", AgesGroup. earlyTeens);
                else if (age > = 7)
                    Console. WriteLine ("your age group is {0}", AgesGroup. children);
                else if (age > = 3)
                    Console. WriteLine ("your age group is {0}", AgesGroup. infancy);
            }
}
```

برنامه ۱ــ۳ــ استفاده از نوع داده شمارشی برای تشخیص گروه سنی

۱ــ در هنگام نوشتن برنامه، از منوی IntelliSense برای نوشتن سریع نوع داده شمارشی مانند شکل ۶ــ۳ کمک بگیرید. در برنامه ۱ــ۳، نوع داده AgesGroup با پنج عضو تعریف شده است، هر

یک از اعضا، معادل یک عدد ثابت است. adult معادل صفر و earlyTeens معادل عدد یک و الی آخر. . .



شکل ۶ــ۳ــ مقداردهی از نوع داده شمارشی

۲ــ در هنگام اجرای برنامه ۱ــ۳، اعداد مختلفی را به عنوان سن کاربر وارد کنید تا خروجی برنامه را مشاهده کنید.

۳ــ دستوراتی به ابتدای برنامه اضافه کنید تا نام کاربر سؤال شود و در پایان برنامه همراه با گروه سنی وی چاپ گردد.

سؤال: در مثال ۱ــ۳ به‌جای سن کاربر سال تولد او را دریافت کرده و گروه سنی او را مشخص کنید.

مثال ۲ــ۳ : برنامه‌ای بنویسید که مقام یک بازیکن را برحسب تعداد سکه‌ای که در اختیار دارد طبق جدول ۲ــ۳ مشخص کند.

جدول ۲ــ۳ــ درجه یک بازیکن

تعداد سکه	مقام بازیکن
تا ۵۰۰۰	معمولی
تا ۱۰۰۰۰	ارشد
تا ۱۵۰۰۰	معاون
از ۱۵۰۰۰ به بالا	رئیس

الگوریتم و روش انجام کار : در این برنامه باید ابتدا تعداد سکه بازیکن سؤال شود و سپس با استفاده از دستورات if تودرتو، شرط‌های جدول یک به یک بررسی شود. اگر یک شرط برقرار شد درجه بازیکن نمایش داده شود.

در برنامه ۲ــ۳، نوع داده PlayerRank با چهار عضو تعریف شده است، هر یک از اعضا، معادل یک عدد ثابت است. Leader معادل صفر و Co_Leader معادل عدد یک و الی آخر. . . .

```
using System;
public enum PlayerRank
{
    Leader, Co-Leader, Elder, Member
}
class playerRankDemo
{
    static void Main()
    {
        PlayerRank player;          ← تعریف متغیر از نوع داده شمارشی
        Console. Write ("Enter Player coins: ");
        string input = Console. ReadLine();
        long coins = long. Parse (input);

        if (coins <10000)
            player = PlayerRank. Member;
        else
            if (coins <15000)
                player = PlayerRank. Elder;
            else
                if (coins <20000)
                    player = PlayerRank. Co-Leader;
                else
                    player = PlayerRank. Leader;
        Console. WriteLine ("The player Rank: {0}", player);
    }
}
```

برنامه ۲ــ۳ــ استفاده از نوع داده شمارشی برای تشخیص مقام یک بازیکن

۱ــ در هنگام نوشتن برنامه، از منوی IntelliSense برای نوشتن سریع نوع داده شمارشی مانند شکل ۷ــ۳ کمک بگیرید.

شکل ۷ــ۳ــ استفاده از نوع داده شمارشی

۲ــ در هنگام اجرای برنامه ۲ــ۳، اعداد مختلفی را به عنوان تعداد سکه وارد کنید تا خروجی برنامه را مشاهده کنید.

۳ــ دستوراتی به ابتدای برنامه اضافه کنید تا نام بازیکن سؤال شود و در پایان برنامه همراه با مقام وی چاپ گردد.

۴ــ دستوراتی به انتهای برنامه ۲ــ ۳ اضافه کرده تا پس از نمایش مقام بازیکن، تعداد الماس[۱] وی را سؤال کند. (قبل از دریافت تعداد الماس، پیام مناسب چاپ کند.)

۵ ــ دستوراتی به انتهای برنامه اضافه کنید که اگر تعداد الماس وی بیش از ۵۰۰۰ بود به بازیکن پیشنهاد داده شود که آیا مایل هستید با خرج کردن (از دست دادن) تعداد ۵۰۰۰ الماس وضعیت و مقام خود را یک پله افزایش دهید؟ اگر کاربر مایل بود و پاسخ Yes داد در این صورت مقام وی یک پله بالا رود و در غیر این صورت در همان وضعیت باقی بماند. در انتهای برنامه مقام جدید، تعداد سکه و الماس باقیمانده وی با رنگ‌های مناسب نشان داده شود.

۲ــ۳ــ استفاده از نوع داده‌های شمارشی آماده در کتابخانه .NET

انواع مختلفی از نوع داده‌های شمارشی آماده در کتابخانه .NET وجود دارد، از جمله نوع شمارشی ConsoleColor که بارها برای تعیین رنگ زمینه و قلم صفحه کنسول از آن استفاده کردید. در واقع به جای حفظ کردن تعدادی عدد، از نام رنگ‌ها که قبلاً در ذهن ما نقش بسته است، استفاده می‌کنیم. مثلاً برای اینکه پیامی به رنگ زرد بر روی زمینه آبی نوشته شود، از دستورات زیر

۱ ــ Gems

استفاده می‌کنید :

```
Console. Background Color = ConsoleColor. DarkBlue;
Console. Foreground Color = ConsoleColor. Yellow;
Console. WriteLine ("Using colors in console mode.");
```

۳ــ ۳ــ شئ[۱]

در زبان محاوره‌ای، با شنیدن کلمه شئ، به یاد یک جسم بی‌جان می‌افتیم. به عنوان مثال، در محیط مدرسه، روزانه با اشیایی نظیر کتاب، دفتر و توپ فوتبال، سروکار داریم. تصور از کلمه شئ می‌تواند کمی توسعه یابد به‌طوری که موضوعات دیگر نظیر کارنامه دانش‌آموز و یا فاکتور خرید یک کالا را نیز به عنوان یک شئ در نظر بگیریم.

هر شئ دارای ویژگی‌هایی است که آن را از اشیای دیگر متمایز می‌سازد. مثلاً یک توپ فوتبال دارای ویژگی‌هایی مانند رنگ، طرح و اندازه است که آن را از توپ فوتبال دیگر متمایز می‌سازد و یا در وضعیت ساکن و یا در حال حرکت است.

بر روی هر شئ نیز عملیاتی می‌توان انجام داد. مثلاً در مورد توپ فوتبال می‌توان عمل شوت زدن، پرت کردن و یا گرفتن توپ را انجام داد.

دانش‌آموز را نیز می‌توانیم یک موجود در نظر بگیریم که دارای ویژگی‌هایی مانند نام و نام‌خانوادگی، نام پدر، شماره ملی است که وی را از بقیه دانش‌آموزان متمایز می‌سازد. همچنین دارای رفتارهای مختلف است. وقتی از وی نامش را سؤال می‌کنند پاسخ می‌دهد، وقتی دوستان و آشنایان را می‌بینند سلام می‌کند. در امتحان شرکت می‌کند، به سؤالات پاسخ می‌دهد و نمره می‌گیرد، خوشحال می‌شود، می‌خندد.

در زندگی روزمره اشیاء و موجودات مختلف با یکدیگر در ارتباط و تعامل هستند. توپ فوتبال توسط دانش‌آموزی شوت می‌شود، توپ به حرکت درآمده و سرعت می‌گیرد و وضعیت آن تغییر می‌کند. دانش‌آموز دیگری برای گرفتن توپ تلاش می‌کند و در اثر باخت بازی

---

۱ ــ Object

ناراحت شده و حتی گریه می‌کند.

دانشمندان رشته ریاضی و کامپیوتر، سعی کرده‌اند که زبان‌های برنامه نویسی طراحی کنند که در آنها، برنامه‌نویس بتواند ویژگی‌ها و رفتارهای اشیاء و موجودات زندگی واقعی را مدل سازی کند تا بتواند برنامه‌ای برای حل مسائل در ارتباط با آنها بنویسد.

زبان‌های برنامه‌نویسی که در آنها امکان تعریف ویژگی‌ها و رفتارهای اشیاء و موجودات فراهم شده است، همچنین اجازه می‌دهند اشیایی برمبنای ویژگی‌های تعریف شده، ایجاد شوند و با یکدیگر ارتباط و نسبت به هم واکنش داشته باشند، زبان‌های شئ گرا[1] نامیده می‌شود.

زبان‌های C++، جاوا و C# از جمله زبان‌های شئ گرا هستند، برنامه‌ای که به این زبان‌ها نوشته می‌شود و اجرا می‌گردد، در واقع از تعدادی شئ تشکیل شده است که با یکدیگر در ارتباط و تعامل هستند.

کلمه شئ در این زبان‌ها، نه تنها اشیای بی‌جان، بلکه موجودات جان‌دار نظیر دانش‌آموز، کارمند را نیز در بر می‌گیرد (شاید بهتر باشد به جای کلمه شئ، کلمه موضوع را به کار ببریم که می‌تواند به هر موجود یا اشیایی اشاره کند).

با توجه به مطالب گفته شده، یک شئ شامل تعدادی ویژگی، وضعیت، رفتار و عملیات است که آن را از اشیای دیگر متمایز می‌سازد.

مثال ۳ــ۳ : کارنامه یک دانش‌آموز را به عنوان یک شئ در نظر می‌گیریم، ویژگی‌های متمایز کننده و عملیات و رفتارهای انجام شده بر روی آن می‌تواند چنین باشد :

مقادیری نظیر نام و نام خانوادگی، نام دروس و نمرات هر یک از آنها، ازجمله ویژگی‌هایی است که یک کارنامه را از کارنامه دیگری جدا و قابل تشخیص می‌کند و برای یک دانش‌آموز و مدرسه اهمیت دارد. وضعیت یک کارنامه می‌تواند قبولی، ردی یا مشروطی باشد. عملیات متداولی که بر روی یک کارنامه صورت می‌گیرد، وارد کردن نمرات در کارنامه، رتبه بندی، چاپ کارنامه و استخراج معدل است که توسط دفتردار مدرسه انجام می‌شود. برای حل یک مسئله به روش شئ گرا، تشخیص اشیاء و ارتباط آنها با یکدیگر بسیار اهمیت دارد.

ویژگی‌ها و وضعیت یک شئ به وسیله تعدادی متغیر که فیلد نامیده می‌شوند، مشخص می‌شود و رفتارهای اشیاء در قالب متدها تعریف می‌گردند. بنابراین محل و مکان تعریف فیلدها و متدهای یک شئ در داخل یک کلاس است (شکل ۸ــ۳).

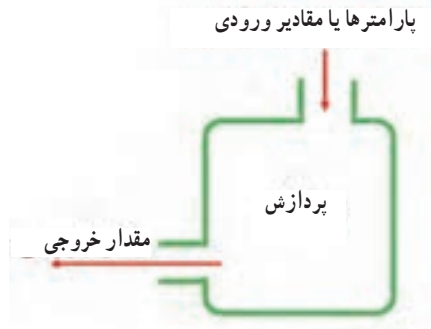
۱ــ Object Oriented Language

شکل ۸ــ۳

با توجه به شکل، می‌توان گفت که **کلاس** تعریف مشخصات، وضعیت و رفتارهای یک شئ را در بردارد و نوع شئ را مشخص می‌کند. بنابراین بهتر است نام کلاس، مطابق با نام شئ باشد که در حال تعریف آن هستیم.

۱ــ۳ــ۳ــ متد :

همان‌طور که می‌دانید متد، مجموعه‌ای از دستورات است که برای انجام یک عمل خاص و حل یک مسئله کوچک به کار می رود. به‌طورکلی متد را می‌توان مانند یک دستگاه در نظر گرفت، که از یک طرف موادی وارد آن می‌شود و از طرف دیگر موادی تغییر یافته، از آن خارج می‌شود. این مواد معمولاً داده‌ها هستند که بر روی آنها یک مرحله پردازش صورت می‌گیرد.



شکل ۹ــ۳ــ پردازش پارامترها

نکته

محل تعریف هر متد در داخل یک کلاس است.

متد جزیی از یک کلاس است. برنامه‌هایی که تاکنون نوشته‌ایم دارای یک کلاس و متد Main بوده‌اند. شکل و ساختار تعریف متد ()Main را در شکل ۱۰ــ۳ مشاهده می‌کنید :

```
static void Main (String[] args)  ← عنوان
{
                                  ← بدنه متد
}
```

شکل ۱۰ــ۳ــ عنوان و بدنه متد Main

تعریف متد ()Main و یا هر متد دیگر، از دو قسمت تشکیل می‌شود :

۱ــ عنوان[۱] متد

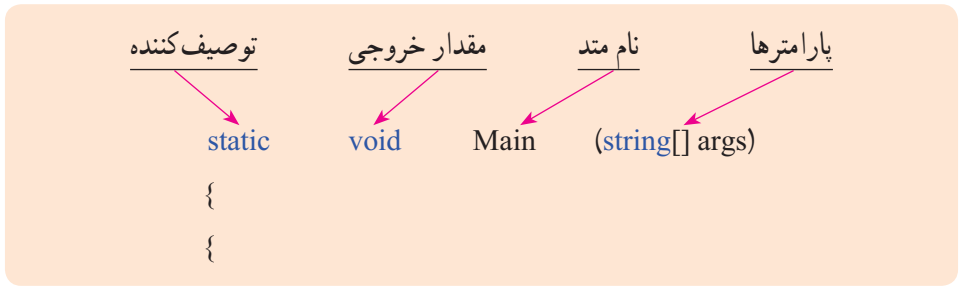
۲ــ بدنه متد

همان‌طور که با مشاهده شناسنامه یک فرد، اطلاعات مهمی‌در مورد مشخصات فردی وی بدست می آوریم، از روی خط عنوان یک متد نیز، اطلاعات اساسی درمورد متد و طریقه استفاده از آن را متوجه می‌شویم. به خط اول تعریف متد، عنوان متد می گویند. در قسمت عنوان متد، مشخصاتی مانند، نام متد، نوع داده خروجی، لیست داده‌های ورودی (پارامترها[۲]) و همچنین روش دسترسی[۳] به متد و نحوه استفاده از آن معین و تعریف می شود. بنابراین خط عنوان یک متد، بسیار بسیار مهم است و حاوی اطلاعات اساسی در مورد روش استفاده از متد است. در شکل ۱۰ــ۳ خط عنوان متد ()Main که در ویژوال استودیو مشاهده می‌کنید، نشان داده شده است. حال به شرح هر یک از قسمت‌های آن می‌پردازیم.

۱ــ Heading

۲ــ Parameters

۳ــ Access



شکل ۱۱ــ۳ــ معرفی خط عنوان متد Main()

بدنه متد همان دستوراتی است که در داخل متد نوشته می‌شود و در برنامه‌های قبلی بارها در متد Main نوشته‌اید و با این دستورات آشنا هستید.

۱ــ۱ــ۳ــ۳ــ توصیف کننده[۱] : خط عنوان یک متد با کلمه‌ای شروع می‌شود که روش ایجاد و محدوده دسترسی به متد را مشخص می‌کند. به عنوان مثال توصیف کننده static، روش ایجاد متد را مشخص می‌کند و نشان می‌دهد که به محض اجرای برنامه، چنین متدی ساخته شده و قابل استفاده است.

متد Main() نیز همواره باید از نوع static باشد اگر توصیف‌کننده static را ننویسید، مترجم در ترجمه برنامه خطا می‌دهد. تمام متدهای آماده‌ای که تاکنون استفاده کرده‌اید نظیر ReadLine() نیز از نوع استاتیک هستند که در کلاس Console از قبل تعریف شده‌اند. استفاده از متدهای static ساده است.

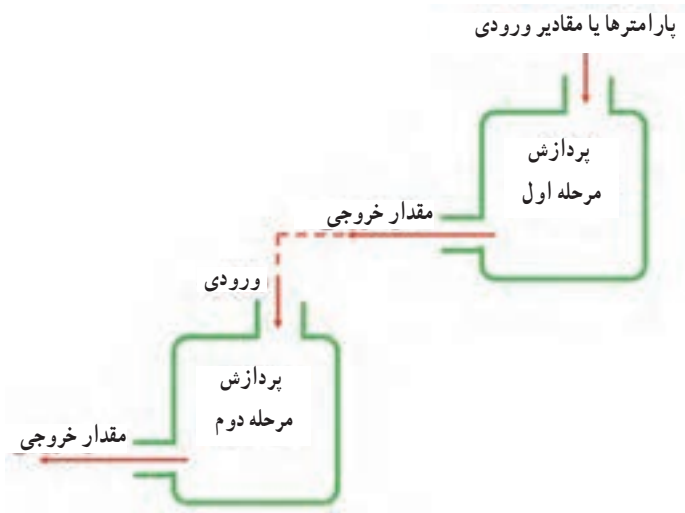
توصیف کننده‌های دیگری نظیر private و protected وجود دارند که به وسیلهٔ آنها می‌توان محدوده دسترسی به متد را به یک کلاس محدود کرد و یا با استفاده از توصیف کننده public محدوده دسترسی به یک متد را بسیار گسترده و در واقع، بدون محدودیت معین کرد. متدهای ReadLine() و WriteLine() با توصیف‌کننده public تعریف شده‌اند و به همین دلیل است که می‌توانید در هر برنامه‌ای (هر کلاسی) از آنها استفاده کنید.

نکته

اگر هیچ توصیف کننده‌ای برای تعیین محدوده دسترسی یک متد ذکر نشود، نوع دسترسی private به‌طور پیش فرض برای متد در نظر گرفته می‌شود.

۱ــ Modifier

۲ــ۱ــ۳ــ۳ــ مقدار خروجی یا نوع داده برگشتی[۱] : در خط عنوان متد، پس از توصیف‌کننده‌ها، مقدار خروجی یا نوع داده برگشتی متد را باید مشخص کنید. **منظور از مقدار خروجی، آنچه که روی صفحه نمایش چاپ می شود نیست** بلکه مقداری است که از طریق اجرای متد قابل دریافت است و می‌توان آن را در یک متغیر ذخیره کرد و یا به متد دیگری برای ادامه پردازش ارسال کرد. شکل ۱۲ــ۳ را ملاحظه کنید.



شکل ۱۲ــ۳ــ استفاده از خروجی یک متد به عنوان ورودی متد دیگر

به عنوان مثال، می‌خواهیم شعاع دایره را از ورودی گرفته و مساحت آن را محاسبه کنیم. بدین‌منظور باید ابتدا از متد ReadLine استفاده کرده و عدد را دریافت کنیم، عدد دریافت شده به صورت string است، پس باید با استفاده از متد Parse() عدد را به نوع داده اعشاری تبدیل کنیم.

```
string numStr=Console.ReadLine();
float num = float.Parse(numStr);
```

ممکن است متدی مقدار برگشتی یا خروجی نداشته باشد مانند متد WriteLine()، در این صورت نوع برگشتی آن void یا بدون مقدار خروجی است. معمولاً متد Main() نیز، بدون مقدار

۱ــ Return Type

برگشتی (void) است[1].

در مقابل متد WriteLine()، متد ReadLine() را درنظر بگیرید که رشته‌ای را از کاربر دریافت می‌کند و آن را به برنامه تحویل می‌دهد. مقدار برگشتی این متد، یک رشته است و معمولاً آن را در یک متغیر از نوع رشته‌ای ذخیره می‌کنیم.

نکته

ممکن است یک متد، مقدار خروجی و یا مقدار ورودی نداشته باشد.

۳ــ۱ــ۳ــ۳ــ نام متد : در خط عنوان متد، پس از تعیین نوع برگشتی، باید نام متد را تعیین کنیم. در نام‌گذاری متد باید مانند نام متغیرها، اصول نام گذاری شناسه‌ها را رعایت کنیم. معمولاً نام متد به صورت یک فعل امری نام‌گذاری می‌شود. این نام بهتر است بیان کننده نوع کاری باشد که متد، قصد انجام آن را دارد. برای مثال متد Clear برای پاک کردن صفحه نمایش و متد Write برای نوشتن روی صفحه نمایش است.

۴ــ۱ــ۳ــ۳ــ لیست پارامترهای ارسالی : در خط عنوان متد، پس از تعیین نام، باید پارامترها یا داده‌هایی که از طرف برنامه به متد داده می‌شود (اصطلاحاً به متد ارسال می‌شود) را مشخص کنیم. مثلاً برای متد Write پارامتر ورودی می‌تواند یک رشته باشد که باید روی صفحه کنسول نمایش داده شود. بعضی از متدها نیز پارامتر ورودی ندارند مانند متد ReadLine().

به عنوان مثال، خط عنوان متد WriteLine() را بررسی می‌کنیم :

```
public static void WriteLine (string value)
```

خروجی — نوع داده ورودی

با توجه به خط عنوان، نام متد WriteLine است و ورودی آن یک رشته مانند value است. این متد خروجی ندارد و با دسترسی وسیع و از نوع استاتیک است.

سؤال: با توجه به خط عنوان متد WriteLine()، توصیف کننده‌های آن کدام است؟

۱ــ البته خروجی متد Main() می‌تواند عدد صحیح int نیز باشد.

۵ــ۱ــ۳ــ۳ــ روش استفاده یا فراخوانی[۱] متد : باید بتوانیم از متد در برنامه استفاده کنیم و یا به اصطلاح آن را فراخوانی کنیم. چنانچه متدی از نوع استاتیک باشد فراخوانی آن بسیار ساده است به شرط اینکه خط عنوان آن را بدانیم. برای اطلاع از خط عنوان متدهای آماده Net. می‌توانید به راهنمای آنها مراجعه کنید. پس از اطلاع از عنوان متد، کافی است فضای نامی و نام کلاس را قبل از نام متد ذکر کنید. این سه قسمت با علامت نقطه از یکدیگر جدا می‌شوند. اگر متد دارای پارامتر است باید مقدار پارامترها را نیز در داخل پرانتز معین کنید و در انتها علامت نقطه ویرگول را بنویسید.

فضای نامی . نام کلاس . نام متد ; مقدار پارامترها

فرض کنید می‌خواهیم صدایی با فرکانس ۲۰۰۰ هرتز به مدت ۱ ثانیه ایجاد کنیم. در کتاب برنامه‌سازی۱ از متد ()Beep برای ایجاد صدا استفاده کردید. این متد صدا را با فرکانس معین در مدت زمان مشخص تولید می‌کند. خط عنوان این متد به صورت زیر است :

```
public static void Beep(int frequency, int duration)
```

این متد به صورت استاتیک و در کلاس Console تعریف شده است و خروجی ندارد. این کلاس در فضای نامی System قرار دارد، بنابراین فراخوانی متد ()Beep چنین است :

```
System.Console.Beep(2000,1000)
```

سؤال: در چه صورت می‌توانیم دستور بالا، را به صورت زیر خلاصه کنیم؟

```
Console.Beep(2000,1000)
```

فرض کنید می‌خواهیم نام کاربر را از ورودی دریافت کنیم. بارها از متد RedLine برای دریافت از ورودی استفاده کرده‌اید. خط عنوان این متد به صورت زیر است :

```
public static string ReadLine( )
```

این متد ورودی ندارد و خروجی آن از نوع string است و به صورت استاتیک و در کلاس

۱ــ Method call

Console تعریف شده است و در فضای نامی System قرار دارد، بنابراین فراخوانی متد ReadLine() چنین است :

```
System.Console.ReadLine( )
```

سؤال: در چه صورت می‌توانیم دستور بالا، را به صورت زیر خلاصه کنیم؟

```
Console.ReadLine()
```

با اجرای هر یک از دستورات بالا، نام کاربر از ورودی دریافت می‌شود ولی برای استفاده در جایی ذخیره نمی‌شود. برای استفاده از نام کاربر، باید خروجی متد را در یک متغیر ذخیره کنیم و یا به متد WriteLine() برای چاپ بر روی صفحه نمایش ارسال کنیم. برای ذخیره نام کاربر در یک متغیر می‌توانیم چنین دستوری بنویسیم :

```
string userName = Console.ReadLine( );
```

و همچنین برای نشان دادن نام کاربر بر روی صفحه نمایش، چنین دستوری می‌نویسیم :

```
Console.WriteLine(Console.ReadLine( ));
```

سؤال: چرا متغیر userName از نوع string تعریف شده است؟

خودآزمایی فصل سوم

الف) در سؤالاتِ چند گزینه‌ای زیر پاسخ صحیح را انتخاب نمایید.

۱ــ چه چیز یک شئ را از اشیا دیگر متمایز نمی‌کند؟

الف) مشخصه ب)رفتار ج)عملیات د)توصیف کننده

۲ــ کدام گزینه در مورد متد صحیح نیست؟

الف) محل تعریف متدهای یک شئ داخل کلاس است.

ب) نحوه دسترسی به متد در خط عنوان متد تعیین می‌شود.

ج) یک متد می‌تواند پارامتر ورودی نداشته باشد.

د) اگر محدوده دسترسی به متد تعیین نشود به طور پیش فرض public در نظر گرفته می‌شود.

۳ــ کدام گزینه در مورد نوع داده شمارشی صحیح است؟

الف) محل قرارگیری تعریف نوع داده شمارشی در متد Main است.

ب) شماره معادل با اولین عضو نوع داده شمارشی، عدد یک است.

ج) برای جدا کردن نام‌ها در نوع داده شمارشی، علامت , به کار می‌رود.

د) نوع دسترسی به یک نوع داده شمارشی معمولاً private به کار می‌رود

ب) جاهای خالی را با عبارات مناسب پرکنید.

۴ــ مجموعه‌ای از چند نام دلخواه که حالت‌ها و مقادیر مختلف یک موضوع را نشان می‌دهند ..... نام دارد.

۵ــ اگر یک متد مقداری برنمی‌گرداند، نوع خروجی آن را ..... تعیین می‌کنیم.

۶ــ کلمه‌ای که خط عنوان متد با آن شروع شده و محدوده دسترسی به متد را تعیین می‌کند ..... نام دارد.

ج) به سؤالات زیر پاسخ دهید.

۷ــ ویژگی‌ها و رفتار و عملیات قابل انجام بر روی هر یک از اشیای زیر را تعیین کنید.

کتاب، تلویزیون، دستگاه بازی

۸ــ با یک مثال نشان دهید که خروجی یک متد می‌تواند ورودی یک متد دیگر باشد.

۹ــ روش ایجاد اکثر متدهای آماده کتابخانهNET. چیست؟

۱۰ــ چرا روش ایجاد متد ()Main باید static باشد؟

۱۱ــ اشکال برنامه زیر چیست؟ با یک تغییر ساده در برنامه اشکال را برطرف کنید.

```
using System;
namespace checkError
{
  class Program
  {
    static void Main(string[] args)
    {
       enum Sesean
    { spring, summer, fall, winter  }

      Console.ReadKey();
    }
  }
}
```

فعالیت

متن زیر از MSDN با موضوع داده‌های شمارشی برداشت شده است.

enum

The enum keyword is used to declare an enumeration, a distinct type that consists of a set of named constants called the enumerator list.

Usually it is best to define an enum directly within a namespace so that all classes in the namespace can access it with equal convenience. However, an enum can also be nested within a class or struct.

By default, the first enumerator has the value 0, and the value of each successive enumerator is increased by 1. For example, in the following enumeration, Sat is 0, Sun is 1, Mon is 2, and so forth.

```
enum Days {Sat, Sun, Mon, Tue, Wed, Thu, Fri};
```

Enumerators can use initializers to override the default values, as shown in the following example.

```
enum Days {Sat=1, Sun, Mon, Tue, Wed, Thu, Fri};
```

In this enumeration, the sequence of elements is forced to start from 1 instead of 0. However, including a constant that has the value of 0 is recommended. For more information, see Enumeration Types (C# Programming Guide).

Every enumeration type has an underlying type, which can be any integral type except char. The default underlying type of enumeration elements is int. To declare an enum of another integral type, such as byte, use a colon after the identifier followed by the type, as shown in the following example.

```
enum Days: byte {Sat=1, Sun, Mon, Tue, Wed, Thu, Fri};
```

The approved types for an enum are byte, sbyte, short, ushort, int, uint, long, or ulong.

واژگان و اصطلاحات انگلیسی فصل سوم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Access	
۲	Behavior	
۳	Enumerated Type	
۴	Gem	
۵	Heading	
۶	Member	
۷	Method Call	
۸	Modifier	
۹	Object Oriented Language	
۱۰	Parameter	
۱۱	Property	
۱۲	Return Type	

فصل چهارم

کار با متدها و کلاس‌های آماده

تاکنون با تعدادی از متدها و کلاس‌های سی شارپ کار کرده‌اید. این متدها و کلاس‌ها، از قبل آماده و درون ساختار سی شارپ تعبیه شده است که می‌توانیم از آنها در برنامه‌های خود استفاده کنیم و به این طریق برنامه خود را به راحتی و با قدرت بیشتر بنویسیم. کتابخانهNET Framework. به فراوانی دارای این کلاس‌ها و متدهاست. در این فصل برای نمونه به سه کلاس اشاره می‌کنیم که کاربردهای زیادی در برنامه‌های شما خواهد داشت.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ اجزای خط عنوان یک متد را توضیح دهد.

۲ــ ورودی‌ها و خروجی‌های یک متد را نام ببرد.

۳ــ با کلاس math و متدهای پرکاربرد آن برنامه بنویسد.

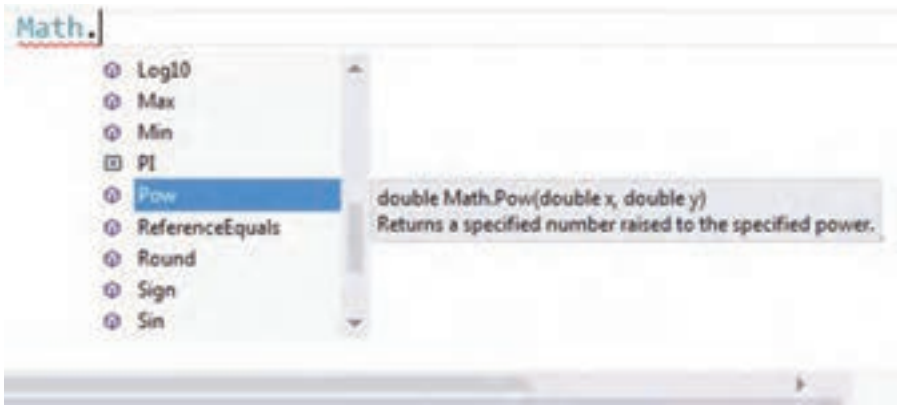
۴ــ از کلاس array و متدهای sort و search این کلاس در برنامه‌های خود استفاده کند.

۵ــ کاربرد کلاس string را بیان نماید و با استفاده از متدهای این کلاس برنامه بنویسد.

۱ــ۴ــ کلاس Math

برای انجام محاسبات ریاضی علاوه بر چهار عمل اصلی، نیاز به عملگرهای دیگری نظیر توان و یا عمل جذر و غیره می‌باشد. زبان C# از طریق کتابخانه Net.، انواع توابع ریاضی، مثلثاتی و لگاریتمی را برای برنامه‌نویس فراهم نموده است. این توابع به‌صورت متدهای استاتیک در کلاس Math تعریف شده‌اند. در این قسمت با تعدادی از آنها که ساده می‌باشند، آشنا می‌شویم. البته قرار نیست اینجا ریاضیات کار کنیم، اما خواهید دید به راحتی می‌توانید کار محاسبه را به برنامه خود واگذارید.

به محض نوشتن نام کلاس Math و علامت نقطه، لیستی از متدهای کلاس Math به وسیلهٔ IntelliSense نشان داده می‌شود.



شکل ۱ــ۴ــ لیست متدهای کلاس Math

با حرکت در لیست مربوطه، در مورد هر یک از متدها توضیحی ظاهر می‌شود. این توضیح شامل خط عنوان متد و عملکرد آن است. در واقع شناسنامه متد نشان داده می‌شود. در ادامه به‌توضیح برخی متدهای کلاس Math می‌پردازیم.

۱ــ۱ــ۴ــ متد ( ) Pow

ورودی‌های این متد، دو عدد x و y از نوع اعشاری با دقت بالا است و متد نیز یک عدد اعشاری دقت بالا که حاصل x به توان y است را تولید نموده و به برنامه اصلی باز می‌گرداند.

مثلاً برای محاسبه عبارت ۵ به توان ۲ کافی است اعداد ۵ و ۲ را به این متد ارسال کنیم و این متد نیز پس از اجرا عدد ۲۵ را بر می‌گرداند. از مقدار برگشتی می‌توان استفاده کرد. به عنوان مثال آن را داخل یک متغیر بریزیم و در یک عملیات دیگر مانند چاپ از آن بهره بگیریم :

```
double number = Math. Pow (5, 2);
Console. WriteLine (number);
```

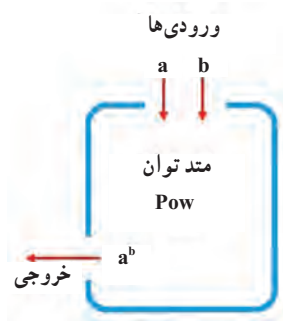
البته می‌توانیم به صورت مستقیم نیز مقدار برگشتی را چاپ کنیم :

```
Console. WriteLine (Math.Pow (5, 2);
```

و یا مقدار خروجی را استفاده نکرد و دور ریخت!

```
Math.Pow (5, 2);
```

عملکرد متد Pow را می‌توان به صورت شکل زیر نشان داد :



شکل ۲ــ۴ــ عملکرد متد Pow

مثال ۱ــ۴ : با اجرای برنامه زیر، عدد ۲ به توان اعداد مختلف (از توان صفر تا ۱۶) روی صفحه نمایش نشان داده می‌شوند.

```
class mathDemo
{
    static void Main ( )
    {
        for (int i = 0; i <= 16;  i++)
        {
            double number = Math. Pow (2, i);
            Console. WriteLine (number);
        }
    }
}
```

برنامه ۱ــ۴ــ نمایش توان‌های مختلف عدد ۲

نکته

کلاس Math در فضای نامی System تعریف شده است و برای استفاده از متدهای آن باید نام System قبل از نام کلاس نیز ذکر گردد و یا در ابتدای برنامه با دستور using، حوزه System به برنامه معرفی شود.

در جدول زیر لیستی از متدهای ریاضی نشان داده شده است. این متدها از کلاس Math هستند.

جدول ۱ــ۴ــ برخی متدهای ریاضی از کلاس Math

نام متد	کاربرد	شکل کلی	مثال
Cos	کسینوس یک زاویه برحسب رادیان را حساب می‌کند.	Math.Cos (عدد)	Math.Cos(10)
Log	لگاریتم عدد را در مبنای تعیین شده حساب می‌کند.	Math. Log (مبنا، عدد)	Math. Log(16,2)
Max	عدد بزرگ‌تر از بین دو عدد را پیدا می‌کند.	Math.Max (عدد، عدد)	Math.Max(10,20)
Min	عدد کوچک‌تر از بین دو عدد پیدا می‌کند.	Math.Min (عدد،عدد)	Math.Min(10,20)
Pow	حاصل عدد به توان تعیین شده را حساب می‌کند.	Math.Pow (توان، عدد)	Math.Pow(10,3)
Round	عدد را گرد می‌کند.	Math.Round (عدد)	Math.Round(10.7)
Sin	سینوس یک زاویه برحسب رادیان را حساب می‌کند.	Math.Sin (عدد)	Math.Sin(20)
Sqrt	جذر عدد را حساب می‌کند. همان رادیکال خودمان.	Math.Sqrt (عدد)	Math.Sqrt(100)
Tan	تانژانت یک زاویه برحسب رادیان را حساب می‌کند.	Math.Tan (عدد)	Math.Tan(20)
Truncate	بخش اعشار عدد را حذف می‌کند.	Math.Truncate (عدد)	Math.Truncate (10.5)

۲ــ۱ــ۴ــ عدد π : برای محاسبه مساحت دایره نیاز به عدد π است. این عدد گنگ از حاصل تقسیم محیط دایره بر اندازه قطر به‌دست می‌آید و برای تمامی‌دایره‌ها ثابت و در حدود ۳/۱۴ است[۱]. اگر نیاز به دقت بیشتری در محاسبات داشتید می‌توانید از ثابت PI که در کلاس Math تعریف شده است استفاده کنید.

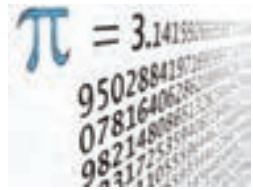
```
Math. PI
```

توجه داشته باشید که PI متد نیست بلکه یک ثابت است. می‌توانید مقدار آن را در یک متغیر اعشاری ذخیره کنید و یا به متد WriteLine() برای چاپ ارسال کنید :

```
Console. WriteLine (Math. PI);
```

فرض کنید : می‌خواهیم اندازه مساحت دایره‌ای را محاسبه و نمایش دهیم که کاربر اندازه شعاع آن را وارد می‌کند.

الگوریتم و روش انجام کار : برای محاسبه مساحت[۲] دایره از فرمول πr^2 استفاده می‌کنیم که منظور از r شعاع[۳] دایره است.




```
// Area of a Circle = πr2
double area = Math. PI * Math. Pow (radius, 2);
```

کنجکاوی

بررسی کنید متد Abs چه عملی انجام می‌دهد؟

۱ــ Pi= 3/14159265358979323846

۲ــ Area

۳ــ Radius

کار در کارگاه ۱

۱ــ برنامه زیر را در محیط VS تایپ کرده و نتیجه اجرای آن را مشاهده کنید :

```
using System;
class Program
{
    static void Main ( )
    {
        Console. WriteLine (Math.Max(100, 200) );   //200
        Console. WriteLine (Math.Max(−100, −200) );

        Console. WriteLine (Math.Min(100, 200) );   //100
        Console. WriteLine (Math.Min(−100, −200) );

        Console. WriteLine (Math.Pow (2, 3) );   //8
        Console. WriteLine (Math.Pow(−2, 3) );

        Console. WriteLine (Math.Round(18.5) );  //19
        Console. WriteLine (Math.Round(18.25) );

        Console. WriteLine (Math.Sqrt (225) );   //15
        Console. WriteLine (Math.Sqrt (200) );

        Console. WriteLine (Math.Truncate (1.23) );   //1
        Console. WriteLine (Math.Truncate (1.54) );
    }
}
```

برنامه ۲ــ۴ــ آزمایش متدهای کلاس Math

۲ــ خروجی هر خط برنامه را بنویسید. نتیجه خروجی بعضی از خطوط برنامه نوشته شده است.

مثال ۲ــ۴ : برنامه‌ای بنویسید که ۱۰ عدد را بگیرد و کمترین آنها را نمایش دهد.

الگوریتم و روش انجام کار : برنامهٔ محاسبهٔ کمترین عدد را سال گذشته نوشته‌اید. با توجه به اینکه در فصل ۲ آرایه را یاد گرفته‌اید، این برنامه را با آرایه می‌نویسیم. برای این منظور پس از تعریف متغیر آرایه‌ای، برای دریافت اعداد از حلقه for کمک می‌گیریم. سپس اولین عدد را به عنوان کمترین عدد لیست در متغیر min می‌ریزیم و از عدد دوم تا انتهای لیست، عمل مقایسه و ریختن در متغیر min را در صورت نیاز انجام می‌دهیم. متد Min کلاس math به ما کمک می‌کند تا عمل مقایسهٔ زیر را با دستور انتساب جایگزین کنیم (برنامه ۳ــ۴).

```
if (min < arr [i])
    min = arr[i];
```

سؤال: با توجه به برنامه زیر توضیح دهید کدام دستور جایگزین شرط بالاست.

```
using System;
namespace MinAndMax
{
    class Program
    {
        static void Main(string[ ] args)
        {
            int[] arr;
            arr = new int[10];
            for (int i = 0; i <=9; i++)
            {
                Console.Write("enter the {0}th number: ", i+1);
                arr[i] = int.Parse(Console.ReadLine());
            }
            int min = arr[0];
            for (int i = 1; i <= 9; i++)
                min = Math.Min(min, arr[i]);
```

```
            Console.WriteLine("minimum is {0}", min);
            Console.ReadKey();
        }
    }
}
```

برنامه ۳ــ۴ــ محاسبه کوچک‌ترین عدد

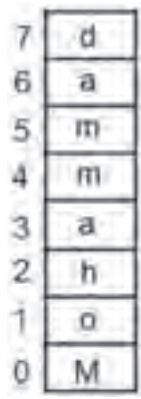
سؤال: اگر بخواهیم بیشترین عدد را هم در این برنامه نمایش دهیم چه تغییراتی باید انجام دهیم؟

۲ــ۴ــ کلاس string

همان‌طور که در برنامه می‌توان متغیری برای نگهداری داده‌های عددی تعریف کرد، برای داده‌های رشته‌ای نیز می‌توان متغیری از نوع string تعریف کرد. این نوع داده در واقع یک کلاس است و متغیری که از این نوع تعریف می‌شود اشاره به ردیفی از کاراکترها می‌کند.

مثال ۳ــ۴ : برای نگهداری کلمه Mohammad از یک متغیر رشته‌ای به صورت زیر استفاده می‌کنیم :

```
string name = "Mohammad" ;
```



شکل ۴ــ۴ــ ساختمان یک رشته

برای آنکه به حروف یک رشته دسترسی داشته باشیم می‌توانیم مانند آرایه‌ها، از اندیس حروف استفاده کنیم.

[شماره کاراکتر] نام رشته متنی

توجه داشته باشید اندیس از صفر شروع می‌شود. بنابراین name[5] کاراکتر ششم رشته name را در خود جای داده است.

نکته

تغییر ناپذیری یک رشته : پس از اینکه یک رشته در حافظه ایجاد شد، محتویات آن قابل تغییر نیست. به عبارت دیگر یک رشته پس از ایجاد، **تغییر ناپذیر**[1] است. شاید از خود بپرسید چرا می‌توانیم به یک رشته مقداری دیگر دهیم؟ پاسخ این است که هر بار که متغیر رشته‌ای مقدار دیگری می‌گیرد مانند این است که دوباره تعریف شده است! اما ما نمی‌توانیم مثلاً یک حرف از آن را تغییر دهیم.

اگر متغیری را از نوع string تعریف نماییم، این متغیر دارای متدهایی خواهد بود که می‌توانند عملیات سودمندی را انجام دهند. مثلاً متدی به نام ()ToLower قادر است حروف انگلیسی موجود در رشته را به حروف کوچک تبدیل کند و در مقابل آن، متد ()ToUpper می‌تواند حروف انگلیسی رشته را به حروف بزرگ تبدیل کند.

۱ ـ Immutable

جدول ۲ــ۴ــ برخی متدهای رشته‌ای از کلاس String[1]

متد	شرح کار متد	نوع برگشتی
CompareTo()	مقایسه دو رشته	bool
IndexOf()	موقعیت وجود یک کاراکتر در رشته را برمی‌گرداند.	int
ToLower ()	تبدیل تمام کاراکترهای یک رشته به حروف کوچک	string
ToUpper ()	تبدیل تمام کاراکترهای یک رشته به حروف بزرگ	string
Insert()	درج یک کاراکتر یا یک رشته در درون یک رشته دیگر	string
Length	تعداد کاراکترهای یک رشته را برمی‌گرداند.	int
Replace()	یک کاراکتر در یک رشته را با کاراکتر یا یک رشته دیگری عوض می‌کند.	string

نکته

این متدها روی خود رشته تأثیر نمی‌گذارند بلکه یک رشته دیگر بر می‌گردانند. بنابراین محتوای متغیر رشته‌ای را تغییر نمی‌دهند.

کار در کارگاه ۳

مثال ۴ــ۴ : کار با متدهای رشته‌ای :

۱ــ عملکرد متدهای مختلف کلاس string را با تایپ برنامه صفحه بعد مشاهده کنید :

۱ــ متدهای کامل در پیوست ۳ ببینید.

```
using System;
class Program
{
    static void Main(string[] args)
    {
      string s = "  This is a String!  ";
      Console.WriteLine ("the main string :  \"{0}\"", s);
      Console.WriteLine (" ToLower:  \"{0}\"", s.ToLower());
      Console.WriteLine (" ToUpper:  \"{0}\"", s.ToUpper());
      Console.WriteLine (" Insert:  \"{0}\"", s.Insert(0, "Note," ) );
      Console.WriteLine (" Length:  {0}", s.Length);
      Console.WriteLine (" index of 'i':  {0}", s.IndexOf('i'));
      Console.WriteLine (" Remove: \"{0}\"", s.Replace("This is", "we have" ) );
      Console.ReadKey();
    }
}
```

برنامه ۴ــ۴ــ آزمایش متدهای کلاس String

کار در کارگاه ۴

مثال ۵ ــ ۴ : برنامه‌ای بنویسید که یک نام کاربری و گذرواژه را بگیرد. اگر نام کاربری user و گذرواژه "pass" وارد شود پیام ورود موفقیت‌آمیز داده شود. همان‌طوری که در پست الکترونیکی و برنامه‌های مختلف دیده‌اید نام کاربری به کوچک یا بزرگی حروف حساس نیست ولی گذرواژه باید دقیقاً مطابقت داشته باشد.

الگوریتم و روش انجام کار : در این برنامه پس از دریافت نام کاربری و گذرواژه، باید بررسی شود که برابر با مقدار خواسته شده می‌باشد یا خیر. برای آنکه بررسی نام کاربری حساس به نوع حروف نباشد نامی که کاربر وارد می‌کند را به رشته‌ای با حروف کوچک تبدیل می‌نماییم. چرا؟

```
using System;
class Program
{
    static void Main(string[] args)
    {
        Console.Write("enter username: ");
        string username = Console.ReadLine();
        Console.Write("enter password: ");
        string password = Console.ReadLine();
        if (username.ToLower() == "user" && password == "pass")
            Console.WriteLine("your login was successful. Welcome!");
        else
            Console.WriteLine("username or password is wrong");
        Console.ReadKey();
    }
}
```

برنامه ۵ــ۴ــ بررسی نام کاربری حساس به کوچکی و بزرگی حروف

سؤال: اگر بخواهیم از ToUpper به‌جای ToLower استفاده کنیم کد چه تغییری می‌کند؟

سؤال: این برنامه را طوری تغییر دهید که تا واردنشدن نام و گذرواژه صحیح، با دادن پیام مناسب، عمل دریافت تکرار شود.

۳ــ۴ــ کلاس Array

در فصل ۲ با مبحث آرایه و کاربردهای آن آشنا شدید. برخی از برنامه‌هایی که با استفاده از آرایه‌ها نوشتیم دارای کدهای طولانی و وقت‌گیری است که با کلاس Array و متدهای آن قابل جایگزینی است. از ویژگی‌های این کلاس، وجود متدهای مختلف استاتیک، برای عملیات بر روی آرایه‌ها است. این متدها کاملاً آزمایش شده و مطمئن هستند. برای مثال عمل جستجوی خطی، دودویی و مرتب سازی آرایه با استفاده از متدهای کلاس Array به راحتی قابل انجام است. در جدول ۳ــ۴ لیست این متدها و توضیحات آنها را می‌بینید.

جدول ۳ــ۴ــ برخی متدهای کلاس Array

نام متد	کاربرد	شکل کلی	مثال
Sort	مرتب سازی لیست	Array.Sort (نام آرایه)	Array.Sort (names)
Reverse	وارونه کردن عناصر آرایه	Array.Reverse (نام آرایه)	Array.Sort (names)
IndexOf	جستجوی یک مقدار در آرایه و برگرداندن مکان اولین مورد پیدا شده	Array.IndexOf (نام آرایه، مقدار)	Array.IndexOf(names,"ali")
LastIndexOf	جستجوی یک مقدار در آرایه و برگرداندن مکان آخرین مورد پیدا شده	Array.LastIndexOf (نام آرایه،مقدار)	Array.LastIndexOf(names, "ali")
BinarySearch	جستجوی یک مقدار در آرایه مرتب شده صعودی و برگرداندن مکان اولین مورد پیدا شده	Array.BinarySearch (نام آرایه، مقدار)	Array.BinarySearch(names, name));
Copy	کپی کردن عناصر یک آرایه در آرایه دیگر از اولین عنصر تا تعداد تعیین شده	Array.Copy (تعداد، نام آرایه مبدأ عناصر، نام آرایه مقصد)	Array.BinarySearch(names, family,10);
Clear	پاک کردن و تنظیم مقدار عناصر تعیین شده آرایه یا مقدار پیش فرض نوع داده آرایه	Array.Clear (تعداد عناصر، شروع اندیس، نام آرایه)	Array.BinarySearch(names, 4,10);

کار در کارگاه ۵

مثال ۶ــ۴ : برنامه‌ای بنویسید که اسامی دانش‌آموزان یک کلاس را دریافت کرده و سپس آنها را براساس نام، مرتب کرده و نمایش دهد.

الگوریتم یا روش انجام کار : ابتدا تعداد دانش آموزان از کاربر دریافت و به همان میزان، آرایه‌ای ایجاد و اسامی از کاربر دریافت می‌شود. در مرحله بعد، اسامی وارد شده مرتب و در نهایت

اسامی چاپ می‌شود.

```
using System;
class Program
    {
    static void Main(string[] args)
        {
        Console.Write("enter number of students: ");
        byte count = byte.Parse(Console.ReadLine());
        string[] studentName = new string[count];
        for (int i = 0; i < count; i++)
        {
            Console.Write("Student Name {0}: ", i + 1);
            studentName[i] = Console.ReadLine();
        }
        Array.Sort(studentName);
        Console.WriteLine("students in name order: ");
        for (int i = 0; i < count; i++)
        Console.WriteLine("{0,3} ==> {1,-25}", i + 1, studentName[i]);
        Console.ReadKey();
    }
}
```

برنامه ۶ــ۴ــ مرتب‌سازی اسامی با استفاده از متد Sort

۲ــ آرایه را پس از مرتب شدن برعکس نمایید.

۳ــ در انتهای برنامه و قبل از دستور Console.ReadKey() قطعه برنامه زیر را که نتیجه جستجوی دودویی و خطی یک نام در آرایه است را اضافه کنید.

```
Console.Write(" \n enter a name for search in students: ");
string name = Console.ReadLine ( );
Console.WriteLine("Binary Search Result: " + Array.Binary Search
(studentName, name) );
Console.WriteLine("linear search Result: " + Array.Index Of (student Name,
name) );
```


شکل ۵ــ۴

۴ــ در اجرای برنامه، یکی از اسامی موجود در لیست دانش آموزان را برای جستجو وارد کنید. نتایج را مشاهده کنید. نتیجه جستجوی دودویی ۱- خواهد بود. این پاسخ به معنای یافت نشدن نام مورد نظر است. چرا نتیجه چنین است؟

۱ــ هر متد از سمت راست را با توضیح سمت چپ مطابقت دهید (یک توضیح اضافی است).

الف) Math.Sqrt	۱ــ گرد کردن عدد
ب) Math.Pow	۲ــ مرتب کردن آرایه
ج) Math.Round	۳ــ وارون نمودن آرایه
د) Array.BinarySearch	۴ــ توان رساندن عدد
و) Array.Sort	۵ــ جستجوی خطی کلید
هـ) Array.Reverse	۶ــ محاسبه جذر عدد
	۷ــ جستجوی دودویی در یک آرایه مرتب

۲ــ با متدهایی که تاکنون شناخته‌اید جدول زیر را تکمیل کنید.

کلاس math	کلاس string	کلاس array

۳ــ راجع به کاربرد متدهایی که در جدول بالا نوشته‌اید با هم کلاس خود بحث کنید.

۴ــ خروجی برنامه زیر چیست؟

```
using System;
class Convert
{
    static void Main(string[] args)
    {
        int a, b;
        double d = 0.75;
        a = (int) Math. Round (d);
        b = (int)d;
        Console. WriteLine (a);
        Console. WriteLine (b);
    }
}
```

۵ــ حاصل عبارت زیر چیست؟

```
Math.Truncate(Math.Round (Math.Pow(2,3)/5) +2.8)
```

۶ــ خروجی دستورات زیر چیست؟

```
static void Main(string[] args)
{
        int [] numbers = { 7, 42, 16, 8, 16, 14 };
        Array.Reverse(numbers);
        Array.Sort(numbers);
        for (int i = 0; i < 6; i++)
           Console.WriteLine(numbers[i]);
         Console.ReadKey();
}
```

۷ــ در سؤال قبل اگر به‌جای دستور for دستور زیر را می‌نوشتیم چه چیزی نمایش داده می‌شد؟

```
Array.IndexOf (numbers, 16);
```

۸ــ دستورات زیر چه کاری انجام می‌دهند؟

```
static void Main(string[] args)
{
    string exp = "i am a good student";
    for (int i = 0; i < strings.Length; i++)
       Console.WriteLine(strings[i]);
    Console.ReadKey();
}
```

۱ــ می‌خواهیم ۲۰ نفر از دانش آموزان را برای نشستن در کارگاه رایانه، به گروه‌های دو نفری تقسیم کنیم. برنامه ای بنویسید که نام این ۲۰ نفر را دریافت کند و با مرتب کردن نام آنها هر دو نفر را برای گروه‌های شماره بندی شده از ۱ تا ۱۰ نمایش دهد.

۲ــ برنامه‌ای بنویسید که یک رشته متنی را دریافت کند. کلمات آن را برعکس نمایش دهد. به عنوان مثال اگر رشته متنی "I am a good student" را به برنامه بدهیم متن "student good a am I" را نمایش دهد.

۳ــ می‌خواهیم در کلاس خود یک فعالیت پژوهشی با حضور ۱۰ دانش‌آموز اول لیست کلاسی انجام دهیم. برنامه‌ای بنویسید که اسامی ۳۰ دانش‌آموز را دریافت کرده و اسامی ۱۰ نفر اول دفتر کلاسی (می‌دانید که اسامی دانش‌آموزان در دفتر کلاسی بر اساس حروف الفبا مرتب شده است) را که در دفتر کلاسی پژوهش (آرایه جدید) نیز وارد می‌کند، نمایش دهد.

فعالیت

متن زیر از MSDN با موضوع آرایه برداشت شده است. آن را با کمک هم کلاسی خود ترجمه کنید و به کلاس ارائه نمایید.

Initializing Arrays

C# provides simple and straightforward ways to initialize arrays at declaration time by enclosing the initial values in curly braces ({}). The following example shows the way to initialize an array.

Note If you do not initialize an array at the time of declaration, the array members are automatically initialized to the default initial value for the array type. Also, if you declare the array as a field of a type, it will be set to the default value null when you instantiate the type.

```
int[] numbers = new int[5] {1, 2, 3, 4, 5};
string[] names = new string[3] {"Matt", "Joanne", "Robert"};
```

واژگان و اصطلاحات انگلیسی فصل چهارم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Area	
۲	Array	
۳	Immutable	
۴	Upper	
۵	Lower	
۶	Radius	

فصل پنجم

ایجاد برنامه‌های ویندوزی با استفاده از ویژوال استودیو

خروجی تمام برنامه‌هایی که در کتاب برنامه سازی یک و همچنین فصل‌های قبلی در این کتاب نوشتید در محیط کنسول قابل مشاهده بود. در این فصل می‌خواهیم با روش برنامه نویسی در محیط ویندوز آشنا شویم تا بتوانیم برنامه‌هایی با ظاهری زیباتر، مانند برنامه‌های کاربردی رایج در ویندوز بنویسیم. برای محقق شدن این هدف، در دو فصل آینده برنامه‌های خود را در محیط ویندوزی خواهیم نوشت.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ تفاوت بین برنامه‌های کنسولی و ویندوزی را توضیح دهد.

۲ــ واسط گرافیکی کاربر و کاربرد کلاس From را توضیح دهد.

۳ــ گزینه‌های مورد نیاز پنجره‌ها، منوها و نوار ابزارهای IDE را در محیط ویندوزی مورد استفاده قرار دهد.

۴ــ پروژه ویندوزی با یک فرم ایجاد نماید و ویژگی‌های مورد نیاز را تنظیم نماید.

۵ ــ کنترل‌های برچسب و جعبه تصویر را به فرم خود اضافه کند و برخی ویژگی‌های آنها را به دلخواه تغییر دهد.

۶ــ با استفاده از VS برنامه ویندوزی را ترجمه و اجرا نماید.

۱ــ۵ــ تفاوت برنامه‌های کنسولی و برنامه‌های ویندوزی

برنامه‌های کنسولی که تاکنون نوشته‌اید برای یادگیری مفاهیم اساسی و مقدماتی زبان برنامه‌نویسی C# و به‌دست آوردن تجربهٔ کدنویسی خوب است اما برای تولید برنامه‌های کاربردی که در زندگی روزمره کاربرد داشته باشد، چندان مناسب نیست. امروزه بیشتر کاربران کامپیوتر، با برنامه‌های کاربردی رایج کار کرده‌اند و توقع دارند برنامه‌ای که شما می‌نویسید دارای ظاهر و عملکردی شبیه برنامه‌های رایج در ویندوز باشند. در اجرای برنامه‌هایی که از این به بعد می‌نویسید انتظار می‌رود که با اجرای آن، یک پنجره باز شود و در داخل آن منوها و ابزارهای مختلفی دیده شود که با کلیک کردن روی هر منو یا روی گزینه‌ای از یک منو، پنجره دیگری ظاهر شود. در این پنجره‌ها، اطلاعاتی از کاربر در قالب یک فرم دریافت می‌شود و یا اطلاعاتی نشان داده می‌شود. در هر فرم، قسمت‌هایی برای ورود اطلاعات و کلیدهایی برای تأیید و ثبت اطلاعات و یا لغو کردن عملیات وجود دارد. به عنوان نمونه در شکل زیر نمونه ای از برنامه حدس عدد را که در کتاب برنامه‌سازی ۱ در محیط کنسول ساخته بودید، در محیط ویندوز مشاهده می کنید. این برنامه را در ادامه می‌سازید.

شکل ۱ــ۵ــ یک فرم و برنامه کاربردی

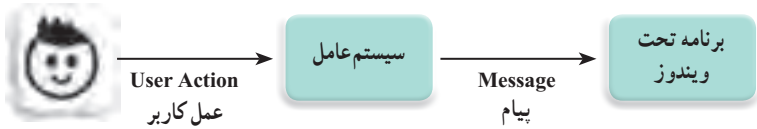
در برنامه‌های کنسولی، اجرای برنامه از متدی به نام Main() شروع می‌شود و دستورات داخل آن، به ترتیب و خط به خط اجرا می‌شوند. در چنین برنامه‌هایی با استفاده از متدهای موجود مانند ReadLine() برای دریافت یک رشته، درخواست‌هایی به سیستم عامل داده می‌شود که عملیاتی را برای ما انجام دهد.

شکل ۲ــ۵

در برنامه‌های ویندوزی نیز اجرای برنامه با متدی به نام Main() شروع می‌شود اما برخلاف برنامه‌های کنسول، برنامه در حالت انتظار[1] قرار می‌گیرد تا یک اتفاق یا رویداد[2] رخ دهد که در این صورت نسبت به آن واکنش نشان دهد. مثلاً وقتی کاربر با ماوس بر روی دکمه[3] در شکل ۱ــ۵ کلیک می‌کند، سیستم عامل که مدیر وکنترل کننده منابع سیستم است متوجه می‌شود و این اتفاق را به عنوان یک رویداد به برنامه اطلاع می‌دهد.

رویداد چیست؟ رویداد یک اطلاع[4] است که از طرف سیستم عامل به برنامه داده می‌شود تا نشان دهد که یک اتفاق رخ داده است.

برنامه‌های ویندوزی باید به رویدادهایی که برای آنها مهم است، عکس العمل نشان دهند.



شکل ۳ــ۵

برای مقایسه بهتر برنامه‌های کنسولی و برنامه‌های ویندوزی، مثال زیر را در نظر بگیرید :

فرض کنید شما در منزل هستید و قرار است برای شما مهمان بیاید. اگر بخواهید به موقع درب منزل را برای وی باز کنید می‌توانید به یکی از روش‌های زیر عمل کنید :

۱ــ هر چند لحظه یک بار، به سمت در منزل بروید و بیرون در را نگاه کنید که آیا مهمان شما آمده است.

۲ــ می‌توانید کارهای دیگر خود را رها کنید و در جلوی در منزل خود، منتظر شوید تا وی بیاید.

۳ــ می‌توانید در داخل منزل مشغول انجام کارهای خود باشید، هر وقت که مهمان آمد، زنگ زد و شما را از ورودش مطلع کرد، به استقبال وی بروید و در را برایش باز کنید.

شکل ۴ــ۵

۱ــ Wait state

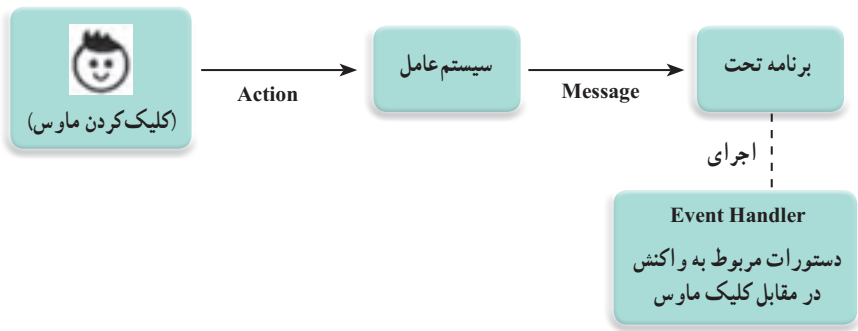
۲ــ Event

۳ــ Button

۴ــ Notification

در برنامه‌های کنسولی، در دستوراتی مانند دستور دریافت داده، اجرای برنامه متوقف می‌شود و تا زمانی که کلید Enter زده شود دستورات بعدی اجرا نمی‌شوند. پس از اینکه کاربر اطلاعات را وارد کرد و کلید Enter را زد، خطوط بعدی برنامه به ترتیب اجرا می‌شوند. در مقایسه با مثال بالا می‌توان گفت در برنامه‌های کنسولی از روشی مانند روش ۱ و یا روش ۲ استفاده می شود. به نظر شما در برنامه‌های ویندوزی، از کدام روش استفاده می‌شود؟

اما در محیط ویندوز، هنگامی‌که کاربر عملی[1] را انجام می‌دهد مثلاً کلیدی از صفحه کلید را فشار می‌دهد و یا با استفاده از ماوس بر روی منویی و یا کلیدی کلیک می‌کند، از طرف سیستم عامل به برنامه اطلاع داده می‌شود که یک رویداد رخ داده است. در واقع برنامه‌های ویندوز با دریافت پیام[2]هایی از سیستم عامل، از رخ دادن رویداد مطلع می‌شوند. بنابراین در برنامه‌های ویندوزی شما به عنوان برنامه نویس، باید پیش بینی کنید که اگر کاربر عملی را انجام دهد، برنامه چگونه نسبت به آن واکنش نشان دهد و برای این منظور باید متدهایی را بنویسید که در مواجهه با یک رخداد یا رویداد مانند کلیک ماوس در یک محل، به کامپیوتر اعلام کند که چه کاری باید انجام شود. به این متدها، Event Handler گفته می‌شود که برای سادگی در این کتاب به جای آن به اختصار EH را به کار می‌بریم.



شکل ۵ــ۵

به این ترتیب هنگامی‌که کاربر بر روی کلیدی با ماوس کلیک می‌کند، این رویداد از طریق سیستم‌عامل شناسایی می‌شود و از طرف سیستم عامل یک پیام به برنامه فرستاده می‌شود و متد EH مربوط به آن رویداد که قبلاً نوشته‌ایم به طور خودکار اجرا می‌شود. طبیعی است که اگر برنامه فاقد

۱ــ Action

۲ــ Message

متدی برای یک رویداد باشد، هر چقدر که سیستم عامل پیام بفرستد برنامه در مقابل آن واکنشی نشان نخواهد داد. شکل کلی یک متد EH به صورت زیر است :

```
void نام متد (فرستنده پیام، جزئیات رویداد)
{
دستورات واکنش به رویداد
}
```

شکل ۶ــ ۵

هر متد EH دارای دو پارامتر ورودی است که اولی نوع شئ[1] فرستنده پیام را مشخص می‌کند و پارامتر دوم مربوط به جزئیات رویداد است مثلاً در مورد کلیک ماوس شامل اطلاعاتی در مورد موقعیت مکانی ماوس در لحظه کلیک و کلیدی (کلید چپ، راست یا وسط) از ماوس است که فشار داده شده است.

هنگامی‌که یک برنامه ویندوزی مانند برنامه ماشین حساب را اجرا می‌کنید، این برنامه پس از نمایش پنجره خود، در حالت انتظار قرار می‌گیرد، تا شما مثلاً بر روی دکمه ای کلیک کنید و یا از طریق صفحه کلید، علامت یا رقمی‌را وارد کنید که در این صورت یک رویداد رخ می‌دهد و EH مربوطه اجرا می شود.

یک تفاوت دیگر بین برنامه ویندوزی و کنسولی این است که، برنامه‌های ویندوزی پس از واکنش به رویدادها و انجام عملیات مربوطه، مجدداً در حالت انتظار برای رویداد بعدی به سر می‌برند تا در نهایت کاربر از برنامه خارج گردد.

در ضمن با توجه به اینکه رویدادهای مختلفی ممکن است از طریق کاربر رخ دهد که قابل پیش‌بینی نیست، بنابراین با اجرای برنامه، مشخص نیست که کدام رویداد ممکن است ابتدا رخ دهد و به عبارت دیگر ترتیبی برای رویدادها، قابل پیش بینی نیست.

۲ــ۵ــ واسط گرافیکی کاربر

در طول دوره‌های مختلف تحصیلی، ممکن است، یک روزنامه دیواری با دوستان خود درست کرده باشید. برای ایجاد روزنامه دیواری ابتدا به یک مقوا نیاز دارید تا مطالب خود را بر روی آن

۱ــ Object

بنویسید و یا عکس‌ها و بریده‌های مجلات را بر روی آن بچسبانید. در ساخت یک برنامه ویندوزی نیز باید صفحه یا فرمی در اختیار داشته باشید تا انواع دکمه‌ها، منوها، تصاویر و نوشته‌ها را روی آن قرار دهید. این فرم، واسط گرافیکی کاربر یا GUI[1] نامیده می شود.

فرم، صفحه‌ای است که اجزای گرافیکی گوناگونی بر روی آن قرار می‌گیرد. اندازه یک فرم متناسب با تعداد و اندازه اشیای گرافیکی است که قرار است روی آن جای گیرند (شکل ۷ــ۵).

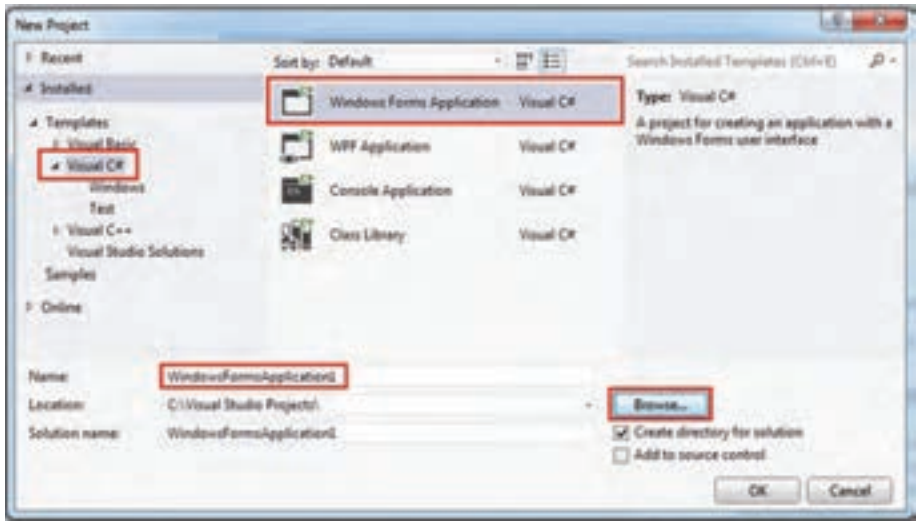
شکل ۷ــ۵ــ یک فرم خالی مانند یک مقوای روزنامه دیواری

همان طور که برای خرید مقوای روزنامه دیواری، لازم است به یک فروشگاه لوازم التحریر مراجعه کنید، در زبان برنامه نویسی C# نیز، برای ایجاد یک فرم باید به کتابخانه ارزشمند Net. مراجعه کنید. در این کتابخانه، کلاسی به نام Form تعریف شده است که برای ایجاد یک فرم لازم است از آن به عنوان پایه و مبنای کار خود استفاده کنیم.

۳ــ۵ــ ایجاد یک پروژه ویندوزی با کمک VS

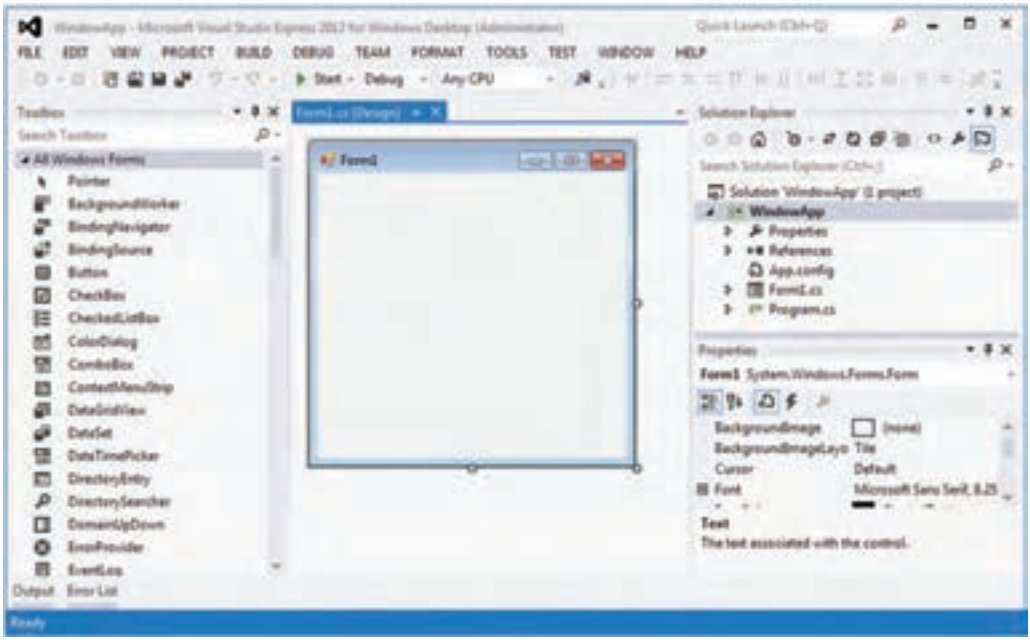
برای ایجاد یک برنامه ویندوزی، VS را اجرا کنید و در صفحه آغازین، روی گزینهٔ New Project کلیک کنید در پنجره‌ای که مطابق شکل ۸ــ۵ نمایان می‌شود، گزینهٔ Visual C# و سپس Windows Forms Application را انتخاب کنید.

۱ــ Graphical User Interface



شکل ۸ــ۵ــ پنجره ایجاد پروژه جدید

می‌توانید نام و مسیر ذخیره سازی پروژه را مانند برنامه‌های کنسولی، انتخاب کنید. پس از تأیید و کلیک بر روی دکمه OK، وارد محیط IDE (شکل ۹ــ۵) می‌شوید.



شکل ۹ــ۵ــ پنجره IDE پروژه ویندوزی

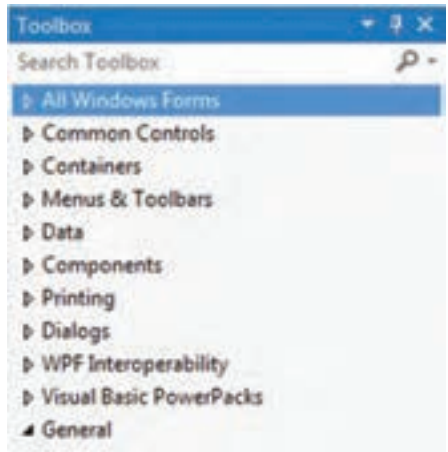
همان‌طور که در شکل ۹ــ۵ مشاهده می‌شود خیلی با این محیط غریبه نیستیم و شبیه محیط تولید برنامه کنسولی است. در اینجا گذری بر روی پنجره‌های مختلف آن خواهیم داشت.

۱ــ۳ــ۵ــ آشنایی با پنجره‌های IDE

در بالای صفحه، منوها و نوار ابزارها[1] مانند محیط تولید برنامه‌های کنسولی وجود دارد. البته همان‌طور که در شکل ۱۰ــ۵ دیده می‌شود یک منو به نام Format اضافه شده است که از گزینه‌های آن برای شکل‌دهی ظاهری و تنظیم جای دهی کنترل‌های یک فرم استفاده می‌شود.

شکل ۱۰ــ۵ــ نوار منوها و نوار ابزارهای IDE در یک پروژه ویندوزی

در سمت چپ صفحه (IDE)، پنجره ای به نام جعبه ابزار[2] دیده می‌شود. در پنجره جعبه ابزار، نام و آیکن کنترل‌های قابل استفاده، لیست شده است. کنترل‌ها در جعبه ابزار، دسته بندی شده‌اند و هنگامی که بر روی نام یک گروه، کلیک می کنید کنترل‌های مربوطه نمایش داده می شوند و اگر مجدداً روی نام آن گروه کلیک کنید، لیست مربوطه بسته می‌شود. در شکل ۱۱ــ۵ جعبه ابزار به صورت گروه بندی نشان داده شده است.

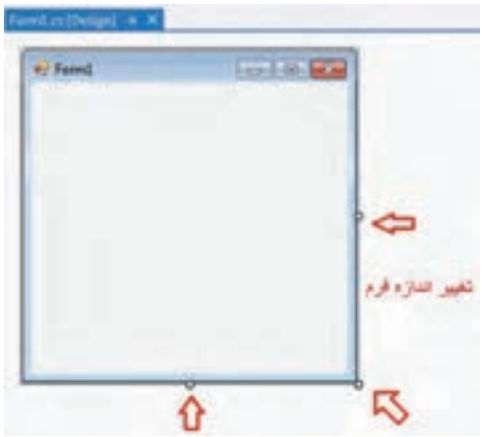


شکل ۱۱ــ۵ــ پنجره جعبه ابزار

۱ــ Tool bar

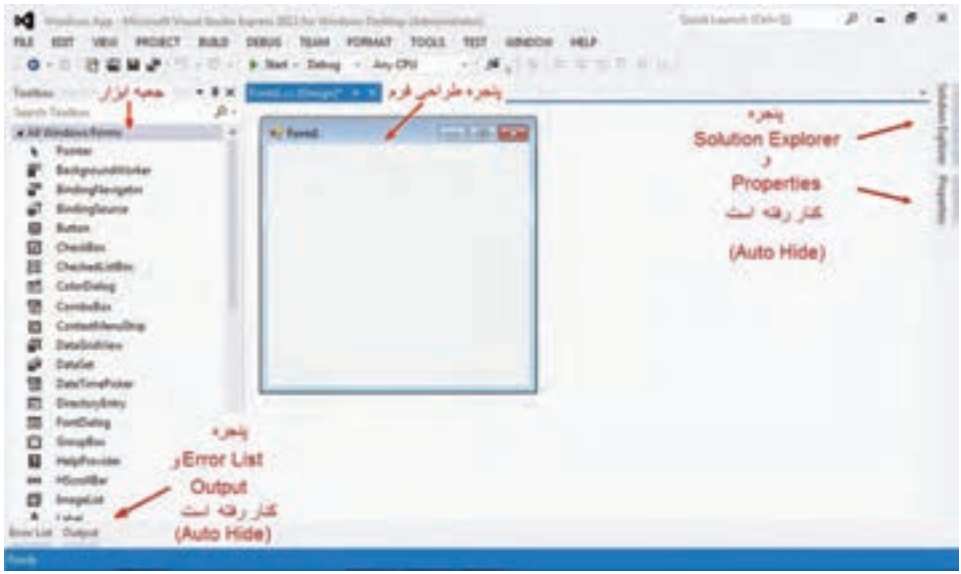
۲ــ Tool box

در وسط صفحه، یک فرم دیده می‌شود که از آن برای طراحی واسط کاربری و قراردادن کنترل‌ها مانند مقوای روزنامه دیواری، استفاده خواهیم کرد. اندازه طول و عرض فرم به راحتی با استفاده از ماوس و با کشیدن گوشه‌های آن، که در شکل ۱۲ــ۵ مشخص شده است، قابل تغییر است. با این پنجره که پنجره طراحی فرم[1] نامیده می‌شود، زیاد کار داریم.



شکل ۱۲ــ۵ــ پنجره طراحی فرم

با توجه به اینکه محیط VS قابل تنظیم است، ممکن است صفحه‌ای که روی کامپیوتر شما دیده می شود با آنچه که در شکل ۹ــ۵ مشاهده می‌کنید، کمی‌متفاوت باشد. مثلاً در شکل ۱۳ــ۵، پنجره‌هایی در حالت Auto Hide، قرار دارند و با کلیک بر روی نام آنها، پنجره دیده می‌شود و با کلیک در نقطه دیگری از صفحه، پنجره‌ها به طور خودکار کنار می‌روند و فقط نام پنجره‌ها دیده می‌شوند.



شکل ۱۳ــ۵ــ حالت Auto Hide پنجره‌ها

۱ــ Form Designer

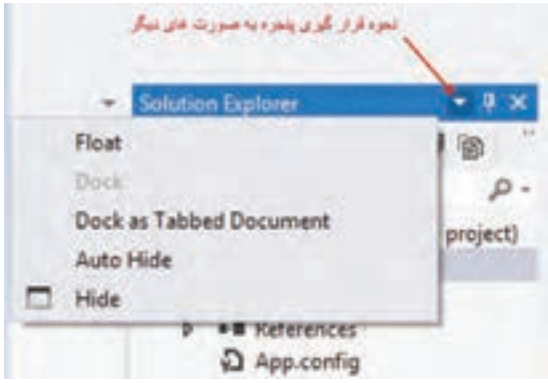
از قابلیت Auto Hide برای کنار رفتن موقتی پنجره‌ها و برای بزرگ‌شدن محیط کار استفاده می‌شود. برای اینکه پنجره‌ای را از حالت Auto Hide خارج کنید و یا این قابلیت را به آن بدهید، کافی است در روی خط عنوان پنجره بر روی آیکن پونز[1] کلیک کنید. شکل ۱۴ــ۵ را مشاهده کنید.

شکل ۱۴ــ۵ــ قابلیت کنار رفتن پنجره به طور خودکار فعال است.

اگر پونز در حالت افقی مانند شکل ۱۴ــ۵ باشد، قابلیت Auto Hide فعال است و اگر روی آن کلیک کنید، پونز به حالت عمودی قرار می‌گیرد و پنجره ثابت می‌ماند. شکل ۱۵ــ۵ را مشاهده کنید.

شکل ۱۵ــ۵ــ حالت کنار رفتن خودکار پنجره غیر فعال است.

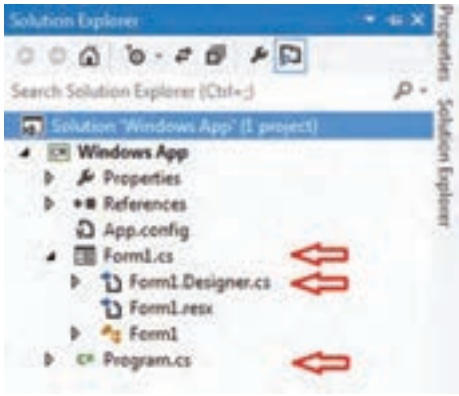
در کنار آیکن پونز، آیکن مثلثی شکلی مانند شکل ۱۶ــ۵ نیز قرار دارد که با کلیک بر آن منوی Windows Posit ion باز می‌شود تا روش‌های دیگری برای قرارگیری پنجره فراهم کند. در قسمت کار در کارگاه آنها را تجربه خواهید کرد.



شکل ۱۶ــ۵ــ نحوه قرار گیری و نمایش پنجره

۱ــ Pushpin

به پنجره Solution Explorer مطابق شکل ۱۷ــ۵ دقت کنید در این پنجره، نسبت به آنچه که در برنامه کنسولی مشاهده کردید، موارد بیشتری دیده می‌شود.



شکل ۱۷ــ۵ــ پنجره Solution Explorer در یک پروژه ویندوزی

علاوه بر فایل Program.cs که نام آن برای شما آشنا است، فایل‌های دیگری از جمله دو فایل به نام‌های Form1.cs و Form1.Designer.cs دیده می‌شوند. دستورات یک برنامه ویندوزی در این سه فایل قرار می‌گیرد. اگر در پنجره Solution Explorer روی نام فایل Program.cs، دوبار کلیک کنید، پنجره‌ای باز می‌شود و محتویات آن را نشان می‌دهد. در این فایل کلاس اصلی برنامه و متد Main()مطابق شکل ۱۸ــ۵ قرار دارد.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Windows_App
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```

شکل ۱۸ــ۵ــ محتوای فایل Program.cs

اگر در پنجره Solution Explorer بر روی کلمه Form1.cs کلیک راست کنید، منویی مانند شکل ۱۹ ــ ۵ ظاهر می‌گردد که شامل گزینه‌هایی برای انجام عملیات مختلف بر روی آن فایل است. به این منوها که گزینه‌های آن مربوط به یک موضوع خاص است منوی موضوعی[1] یا محلی گفته می‌شود.

شکل ۱۹ــ۵ــ منوی موضوعی در مورد فایل Form1.cs

در منوی ظاهر شده، اگر بر روی گزینه View Designer کلیک کنید یا از میان‌بر[2] آن Shift+F7 استفاده کنید، پنجره طراحی فرم را خواهید دید.

نکته

در هنگام برنامه‌نویسی در محیط VS، سعی کنید از کلیدهای میان‌بر استفاده کنید تا سرعت شما در انجام عملیات بالا رود. مثلاً هنگامی که در پنجره Solution Expolrer فایل Form1.cs انتخاب شده است، برای مشاهده دستورات آن و یا برای مشاهده پنجره طراحی فرم که بارها مورد استفاده قرار می‌گیرد، از میان‌برهای زیر استفاده کنید.

مشاهده دستورات فایل فرم	Ctrl + Alt +0
مشاهده پنجره طراحی فرم	Shift + F7

۱ــ Context menu
۲ــ Shortcut

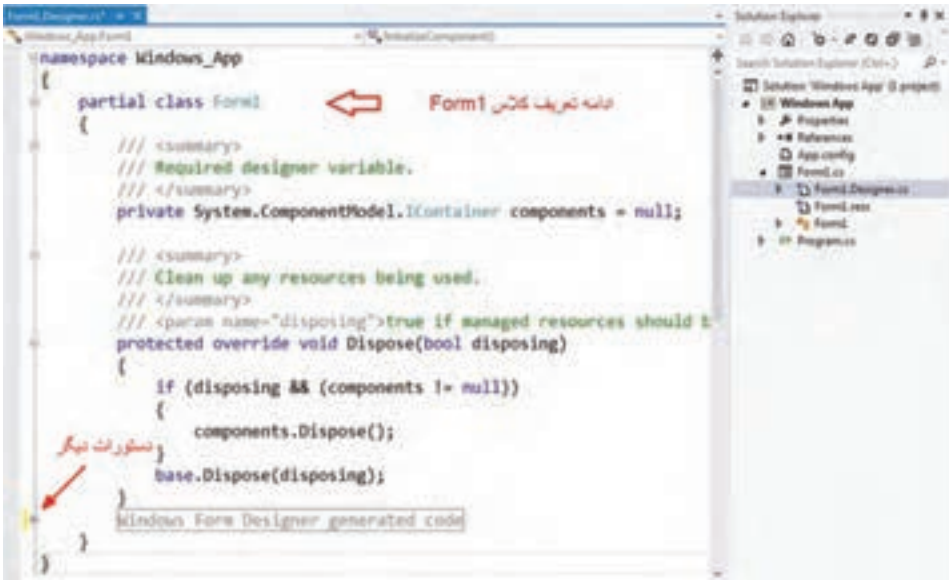
در منوی موضوعی فایل Form1.cs، اگر بر روی گزینه View Code کلیک کنید، محتوای فایل Form1.cs را مانند شکل ۲۰ــ۵ مشاهده خواهید کرد، که قسمتی از تعریف کلاس Form1 است.

شکل ۲۰ــ۵ــ محتوای دستورات و برنامه‌فایل Form1.cs

به خط عنوان تعریف کلاس Form1 توجه کنید، کلمه کلیدی partial در خط عنوان کلاس نشان می‌دهد که بخشی از تعریف کلاس Form1 در فایل دیگری قرار دارد. قسمت دیگر تعریف کلاس Form1 کجا است؟

فایل دیگری که VS ایجاد می‌کند و شما نباید محتویات آن را دستکاری کنید، فایل Form1.Designer.cs است. هنگامی که از جعبه ابزار سمت چپ صفحه، کنترلی را انتخاب و آن را در روی فرم قرار می‌دهید و ویژگی‌های آن را مطابق سلیقه خود تنظیم می‌کنید، VS تنظیمات شما را به دستورات و کدهای زبان C# تبدیل می‌کند. این دستورات به عنوان بخشی از تعریف کلاس Form1 به وسیلهٔ VS در فایل Form1.Designer.cs به‌طور خودکار نوشته می‌شود و کار شما را بسیار آسان می‌کند. بنابراین بقیه تعریف کلاس Form1 در این فایل قرار دارد. در پنجره Solution Explorer آن را پیدا کرده و با دو بار کلیک، با احتیاط آن را باز می‌کنیم. همان‌طور که در شکل ۲۱ــ۵ مشاهده می‌کنید ادامه تعریف کلاس Form1 دیده می‌شود. دستورات و محتویات این فایل، با

قراردادن کنترل‌ها بر روی فرم، افزایش می‌یابد. برای جلوگیری از دستکاری دستورات این فایل و در دسترس قرار ندادن آنها، مترجم آنها را در لحظه اول نمایش نمی‌دهد.



شکل ۲۱ــ۵ــ محتوای دستورات و برنامه فایل Form1.Designer.cs

برای نمایش دستورات ، در سمت چپ پنجره‌ای که محتوای فایل Designer را نشان می‌دهد، یک خط عمودی را مشاهده می‌کنید که در بین آن علامت‌های – و در بعضی قسمت‌ها علامت + قرار دارد. اگر بر روی علامت + کلیک کنید ، منطقه‌ای نمایان می‌شود که تعدادی از دستورات در آن قرار گرفته است.[۱] لطفا این دستورات را بدون آگاهی تغییر ندهید که در این صورت قسمت طراحی فرم دچار مشکل می‌شود.

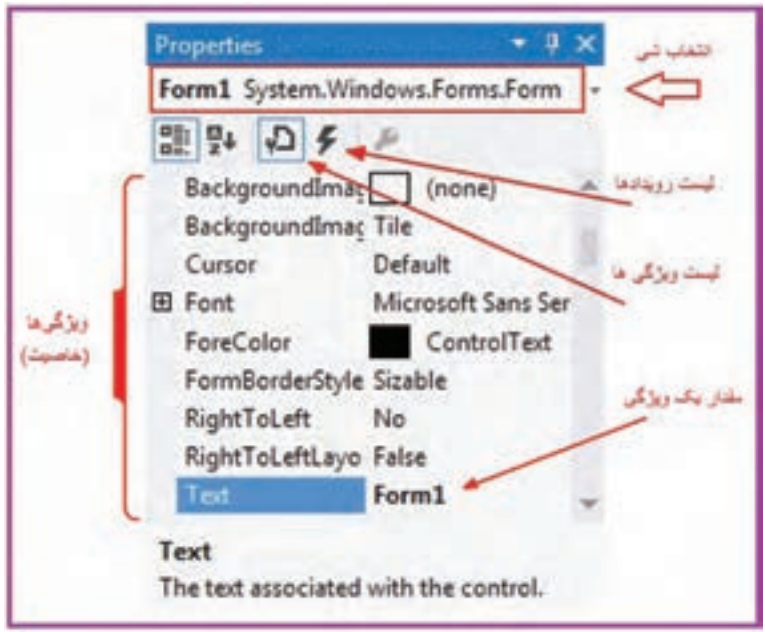
۲ــ۳ــ۵ــ پنجره ویژگی‌ها و خواص اشیاء

پنجره‌ای که در برنامه‌های ویندوزی از آن زیاد استفاده خواهیم کرد، پنجره Properties است که در شکل ۲۲ــ۵ مشاهده می‌کنید.

۱ــ با استفاده از راهنمای region# و endregion#، این امکان مهیا می‌شود که تعدادی دستور در لحظه اول نشان داده نشوند و در صورت لزوم با کلیک بر روی علامت +، نمایان شوند.

Formatted: Font: 11 pt, Complex Script Font: 11pt

Formatted: Font: 11 pt, Complex Script Font: 11pt



شکل ۲۲ــ۵ ــ پنجره Properties

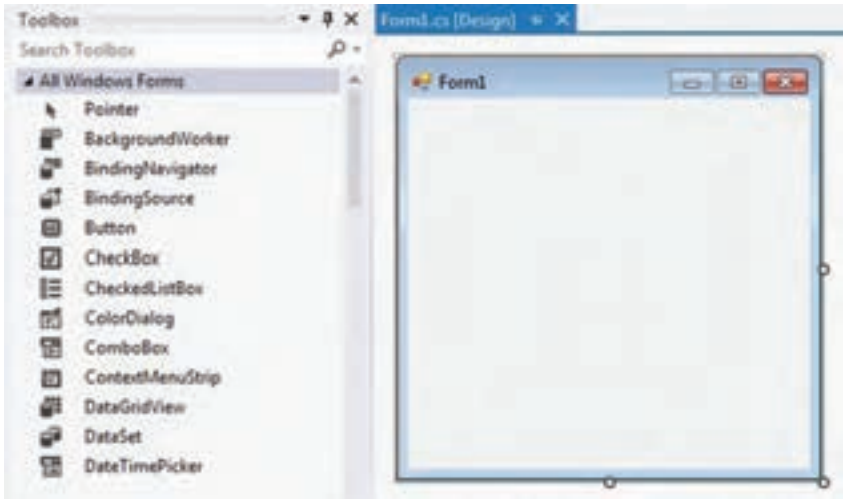
در این پنجره ویژگی‌ها و خواص شئ و همچنین واکنش به رویدادهای هر شئ را می‌توانیم مقداردهی و تنظیم کنیم. در قسمت بالای این پنجره، نام یک شئ که در فرم انتخاب شده، نشان داده شده است و از طریق مثلث کوچک کنار آن می‌توان شئ دیگری را از بین اشیای موجود روی فرم انتخاب کنیم. در زیر نام شئ، می‌توان با کلیک بر روی آیکنی که نظیر رعد و برق است، لیست رویدادهای شئ مشاهده و یا بر روی آیکن کنار آن کلیک کرد تا لیست ویژگی‌های شئ را مشاهده شود. در هر کدام از انتخاب‌ها، لیست نشان داده شده را می‌توان به ترتیب نام[1] و یا به ترتیب عملکرد که دسته‌بندی[2] شده است، مشاهده نمایید. در شکل ۲۳ــ۵، آیکن‌های مربوطه دیده می‌شوند.

شکل ۲۳ــ۵ــ آیکن‌های ترتیب نمایش ویژگی‌ها در پنجره Properties

۱ــ Alphabetical view ۲ــ Categorized view

۳ــ۳ــ۵ــ استفاده از جعبه ابزار و قراردادن کنترل‌ها در روی فرم

همان‌طورکه در فصل قبل بیان شد، یک برنامه ساده ویندوزی از یک فرم تشکیل می‌شود که به عنوان نگهدارنده[۱] برای قرار گرفتن اجزای گرافیکی دیگر، استفاده می‌شود. شکل ۲۴ــ۵ را مشاهده کنید.



شکل ۲۴ــ۵ــ پنجره طراحی و جعبه ابزار

در حالی که پنجره طراحی فرم را مشاهده می‌کنید، می‌توانید از جعبه ابزار استفاده کرده و کنترل مورد نظر خود را پیدا کنید. برای قرار دادن یک کنترل بر روی فرم، می‌توانید روی کنترل مورد نظر دوبار کلیک کنید و یا با درگ کردن کنترل و رها کردن آن روی فرم، کنترل مورد نظر را روی فرم بچسبانید. البته بعضی از کنترل‌ها مانند کنترل زمان‌سنج (تایمر)[۲]، با توجه به عملکردشان، ممکن است در قسمت پایین فرم قرار گیرند که با آنها در این کتاب آشنا می‌شوید.

اندازه فرم که برای طراحی استفاده می‌کنید قابل تغییر است. توجه داشته باشید که اگر در پنجره Form Designer، در روی فرم دوبار کلیک کنید، دستوراتی به فایل Form1.cs اضافه می‌شود که مربوط به الگوی، یک EH است که در شکل ۲۵ــ۵ نشان داده شده است، به دلیل اینکه به طور اتفاقی و اشتباهاً دوبار کلیک کرده‌اید، لذا دستورات اضافه شده را حذف کنید و یا با احتیاط، عمل انجام‌شده را لغو و یا بی‌اثر کنید.

۱ــ Container

۲ــ Timer Control

```
namespace Windows_App
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }
    }
}
```

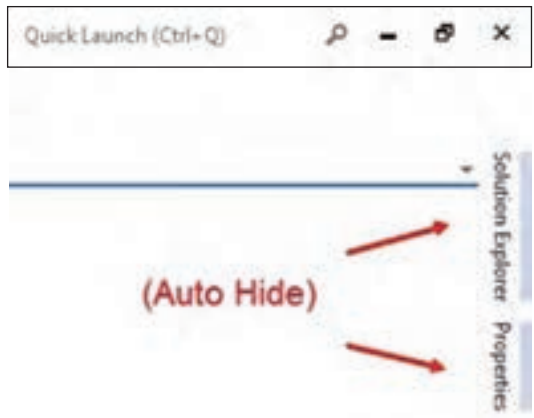
شکل ۲۵ــ۵ــ دستورات اضافه شده در اثر دو بار کلیک بر روی فرم

کار در کارگاه ۱: جابه‌جایی پنجره‌ها در VS

۱ــ برنامه VS را اجرا کنید و یک پروژه Windows Forms Application ایجاد کنید.

۲ــ پنجره Solution Explorer را در حالت AutoHide قرار دهید.

۳ــ پنجره Properties را نیز در حالت AutoHide مانند شکل ۲۶ــ۵ قرار دهید.

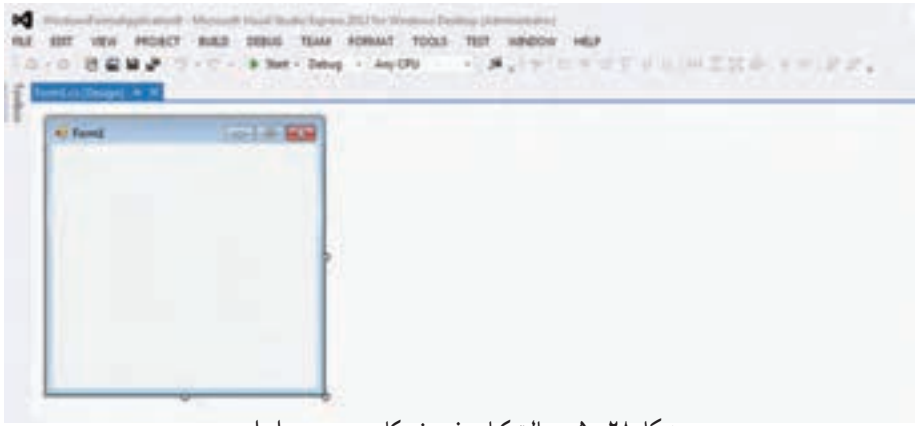


شکل ۲۶ــ۵ــ حالت کنار رفتن خودکار

۴ــ هر دو پنجره را از حالت Auto Hide خارج کرده و به حالت چسبیده[1] مانند شکل ۲۷ــ۵ قرار دهید.

شکل ۲۷ــ۵ــ پنجره‌ها در حالت چسبیده (Dock)

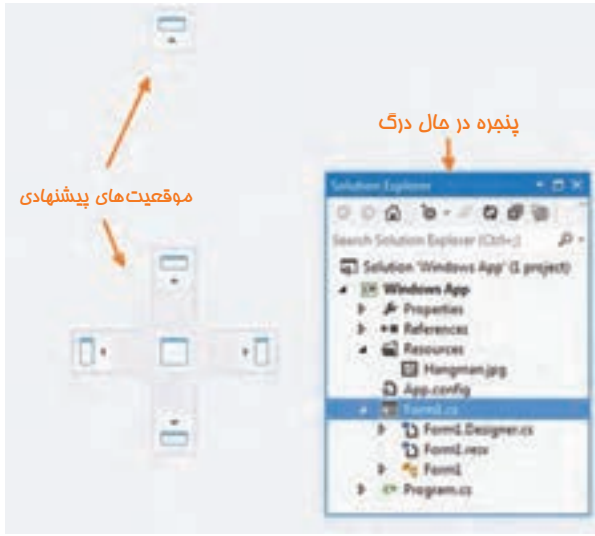
۵ــ پنجره Tool box را در حالت Auto Hide قرار دهید تا قسمت طراحی فرم مانند شکل ۲۸ــ۵ بزرگ‌تر شود. اگر از قبل در حالت Auto Hide است آن را به حالت چسبیده تغییر دهید.



شکل ۲۸ــ۵ــ حالت کنار رفتن خودکار پنجره جعبه ابزار

۱ــ Dock

۶ــ ماوس را روی خط عنوان پنجره Solution Explorer برده و با استفاده از درگ کردن، آن را به نقاط دیگر ببرید. توجه کنید که VS مکان‌هایی را مانند شکل ۲۹ــ۵ به شما پیشنهاد می‌دهد. از آنها استفاده و موقعیت‌های مختلف را تجربه کنید. سعی کنید که مفهوم موقعیت‌های پیشنهادی را بفهمید.



شکل ۲۹ــ۵ــ انتقال پنجره به نقاط دیگر صفحه

۷ــ برای اینکه هر دو پنجره Solution Explorer و Properties از کل فضای سمت راست صفحه، به طور مشترک استفاده نمایند، آنها را به صورت Tab Docked مانند شکل ۳۰ــ۵ قرار دهید. برای این منظور پنجره Properties را درگ کرده و وقتی روی خط عنوان پنجره Solution Explorer رسید، آن را رها کنید.

۸ــ مجدداً هر دو پنجره را به صورت چسبیده، مانند شکل ۲۷ــ۵ قرار دهید.

شکل ۳۰ــ۵ــ پنجره‌ها در حالت Tab Docked

اکنون که با پنجره‌های محیط طراحی برنامه ویندوزی در VS آشنا شدیم، می‌توانیم یک نمونه پروژه ویندوزی را قدم به قدم دنبال کنیم.

۴ــ۵ــ ساخت یک واسطه کاربری با استفاده از قابلیت طراحی تصویری در VS

هنگامی که از امکانات VS برای طراحی واسطه کاربری استفاده می‌کنید، بسیاری از دستورات و کدهای لازم، بدون اینکه شما متوجه شوید، به طور خودکار به وسیلهٔ VS تولید شده و در قالب فایل‌هایی که در ابتدای این فصل معرفی شد، به برنامه اضافه می‌شوند.

در قسمت کار در کارگاه ۲ ، مراحل ساخت یک نمونه فرم ساده را بدون نوشتن حتی یک خط برنامه، دنبال می‌کنیم. بدیهی است برای تولید فرم‌های کاربردی و واکنش به رویدادها، علاوه بر استفاده از ابزارها و امکانات VS ، باید چند خط برنامه نیز بنویسیم.

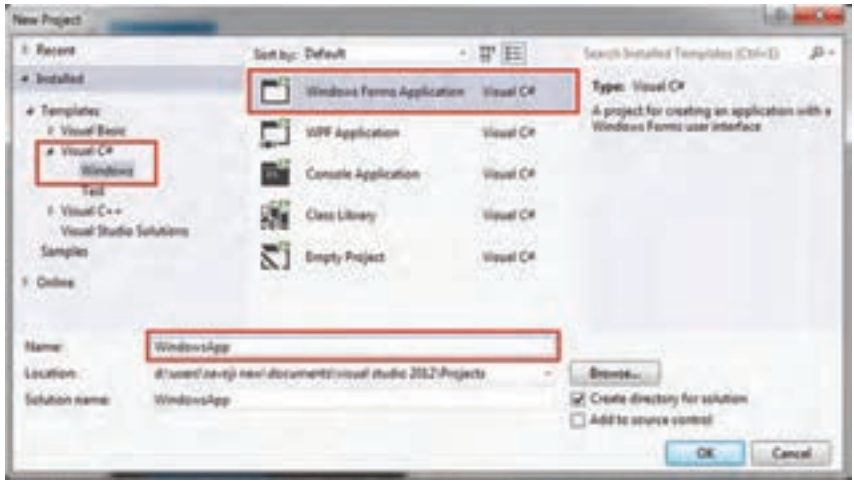
کار در کارگاه ۲: ساخت اولین فرم

ساخت یک برنامه یا یک فرم مانند شکل ۳۱ــ۵

شکل ۳۱ــ۵ــ اولین برنامه در ویژوال C#

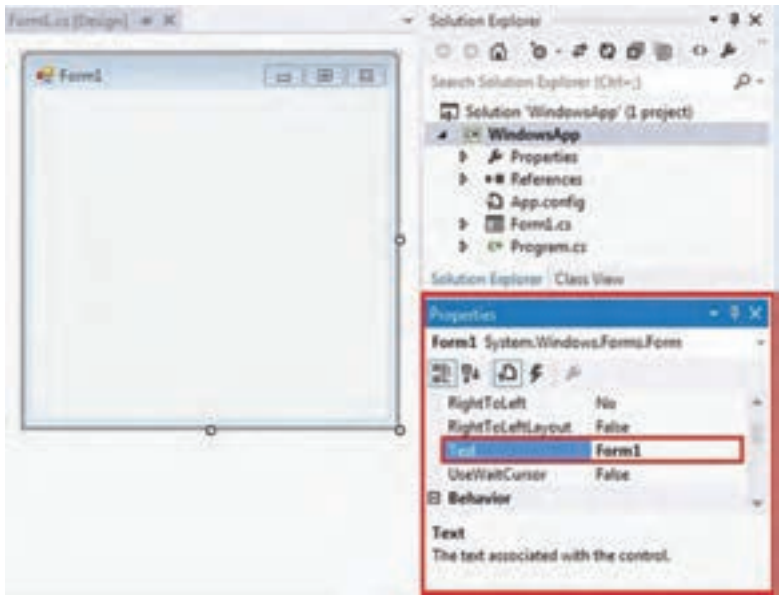
الگوریتم یا روش انجام کار : برای ایجاد چنین فرمی، مراحل زیر را به دقت انجام دهید :

۱ــ ایجاد یک پروژه ویندوزی: وارد برنامه VS شوید و یک پروژه جدید از نوع Windows Forms Application با نام پروژه WindowsApp و در مسیر مشخصی مانند شکل ۳۲ــ۵ بسازید. اگر در حال حاضر در داخل VS هستید از طریق منوی File، انتخاب گزینه Close Solution ، پروژه فعلی را ببندید و سپس یک پروژه جدید مطابق با آنچه که گفته شد، بسازید.



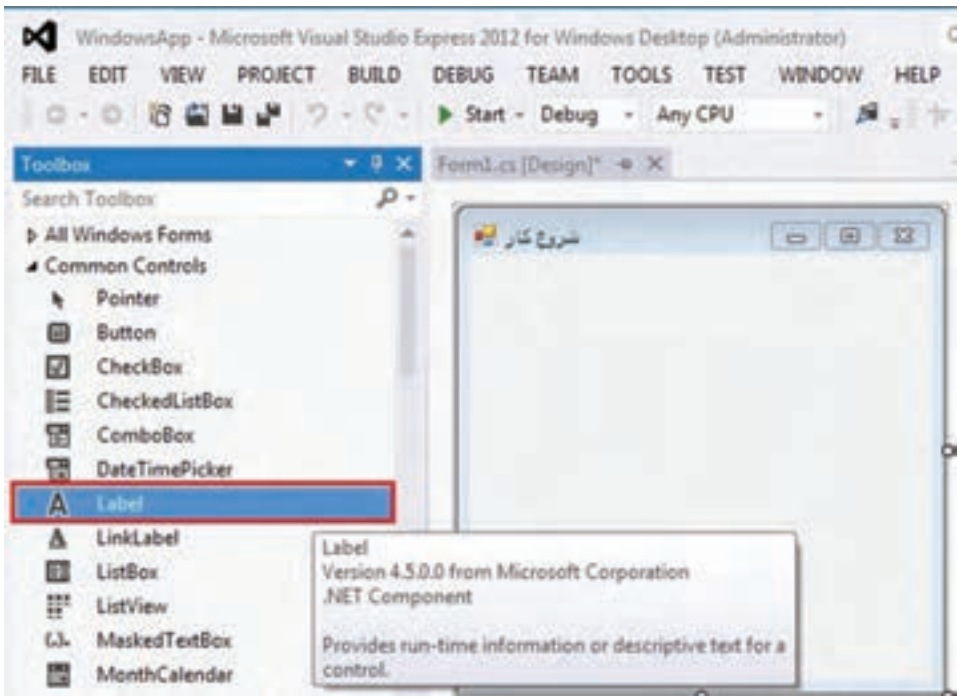
شکل ۳۲ــ۵ــ ایجاد پروژه جدید

۲ــ تعیین خط عنوان فرم: اگر به خط عنوان فرم در شکل ۳۱ــ۵ توجه کنید، عبارت فارسی «شروع کار» دیده می‌شود. برای تغییر عنوان فرم به عبارت خواسته شده، به پنجره ویژگی‌ها(Properties) رفته و مانند شکل ۳۳ــ۵ خاصیت Text را پیدا کرده و متن داخل آن یعنی "Form1" را پاک کرده و عبارت جدید «شروع کار» را تایپ کنید. اگر پنجره Properties دیده نمی‌شود از منوی View، بر روی گزینه Properties Window کلیک کنید و یا از میان بر مربوطه استفاده نمایید.



شکل ۳۳ــ۵ــ تغییر خاصیت Text مربوط به فرم از طریق پنجره Properties

۳ــ *اضافه کردن یک متن نمایشی بر روی فرم :* برای نمایش یک متن یا عبارت بر روی فرم، از کنترل برچسب (Label) استفاده می‌کنیم. به سراغ پنجره جعبه ابزار رفته و کنترل برچسب (Label) را پیدا کرده و آن را بر روی فرم قرار می‌دهیم. شکل ۳۴ــ۵ را مشاهده کنید.



شکل ۳۴ــ۵ــ کنترل برچسب در جعبه ابزار

با یکی از روش‌های زیر می‌توانید کنترل‌ها را بر روی فرم قرار دهید :

۱ــ با استفاده از دابل کلیک بر روی کنترل

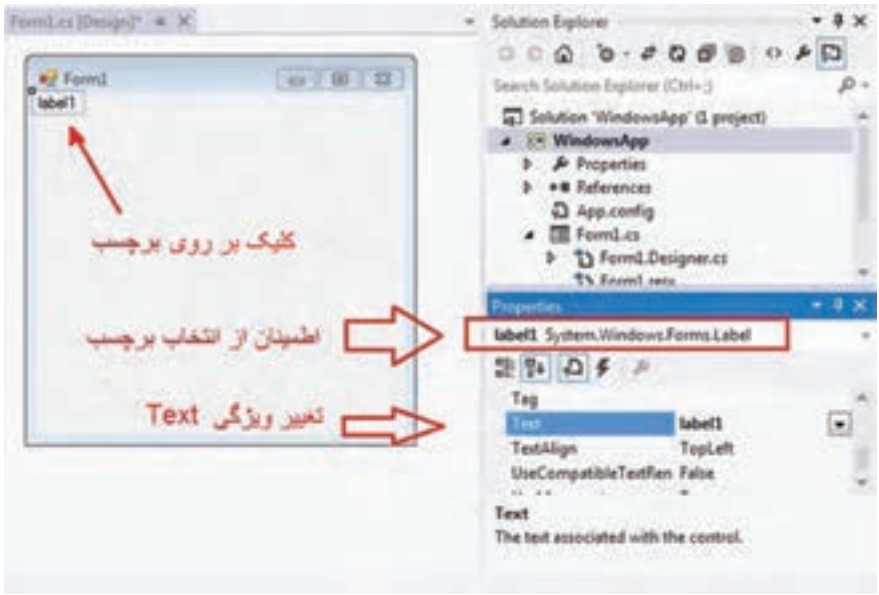
۲ــ با استفاده از روش Drag & Drop

۴ــ *تعیین ویژگی Text برای برچسب :* به پنجره طراحی فرم مراجعه کرده، بر روی برچسب یک بار کلیک کنید تا انتخاب شود. با انتخاب کنترل برچسب روی فرم، پنجره Properties، ویژگی‌های آن را نشان می‌دهد. نام این کنترل به طور خودکار به وسیلهٔ VS انتخاب می‌شود که label1 است. در واقع این نام، نام شئ است که از کلاس Label ایجاد و یا نمونه سازی شده است. در خط

اول پنجره Properties، نام شئ به همراه نام کلاس آن، نشان داده شده است.

(Label1 System.windows.Forms.Label)

حال در پنجره Properties، پس از اطمینان از اینکه برچسب انتخاب شده، خاصیت Text را مانند شکل ۳۵ــ۵ پیدا کنید.

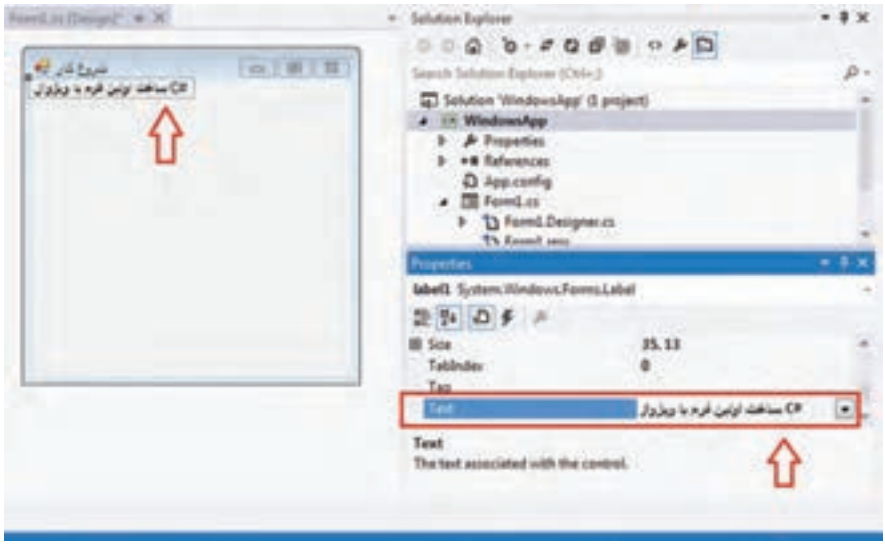


شکل ۳۵ــ۵ــ تغییر ویژگی متن کنترل برچسب

برای اطمینان از اینکه برچسب انتخاب شده است به قسمت بالای پنجره Properties توجه کنید، باید نام برچسب را مشاهده کنید. اگر چنین نیست روی فلش کوچک کنار آن کلیک کنید، نام برچسب را پیدا کرده و آن را انتخاب کنید.

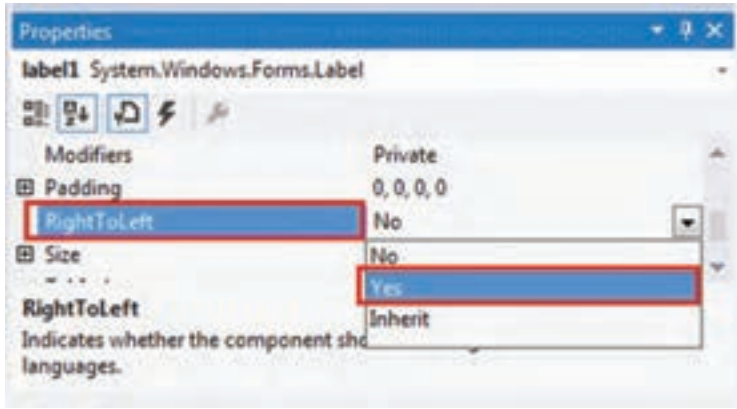
پس از انتخاب کنترل برچسب، عبارت «ساخت اولین فرم با ویژوال C#» را در قسمت Text برچسب وارد کنید.

همان‌طور که در شکل ۳۶ــ ۵ مشاهده می‌شود، عبارت مورد نظر در خاصیت Text برچسب، وارد شده اما به هم ریخته است. چون عبارت وارد شده، ترکیبی از کلمات فارسی و همچنین حروف انگلیسی است ، لذا مشکل به‌هم ریختگی کلمات پیش آمده و عبارت «C#» علاوه بر اینکه در جای خود (انتهای جمله) قرار ندارد به شکل برعکس یعنی «C#» دیده می‌شود. برای رفع این مشکل چه باید کرد؟



شکل ۳۶ــ۵ ــ متن شامل کلمات فارسی و انگلیسی

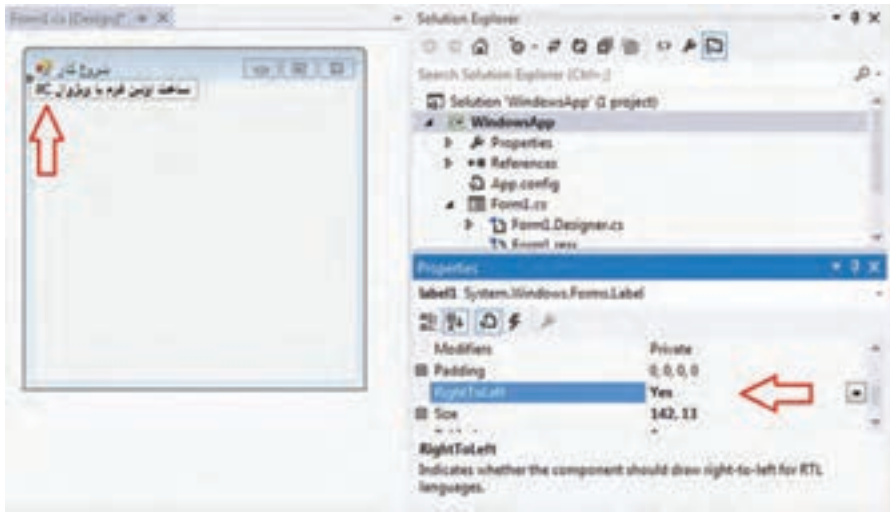
قبلاً با چنین مشکلاتی در برنامه‌هایی نظیر Word روبه‌رو شده بودید. در اغلب برنامه‌های کاربردی، از زبان‌های محاوره‌ای پشتیبانی می‌شود که در آنها کلمات و جملات از سمت راست به چپ نوشته می‌شود. در.NET نیز چنین است. در کنترل‌ها، خاصیتی به نام RightToLeft برای پشتیبانی از زبان‌های راست به چپ (RTL)[1]، پیش بینی شده است. بنابراین به سراغ خاصیت RightToLeft بروید و مقدار آن را به Yes تغییر دهید. شکل ۳۷ــ۵ را ملاحظه کنید.



شکل ۳۷ــ۵ ــ ویژگی RightToLeft برای پشتیبانی از زبان‌های RTL

۱ــ Right to Left Languages

با تغییر دادن مقدار ویژگی RightToLeft ، به مقدار Yes ، عبارت نوشته شده در برچسب، به صورت شکل ۳۸ــ۵ خواهد شد :



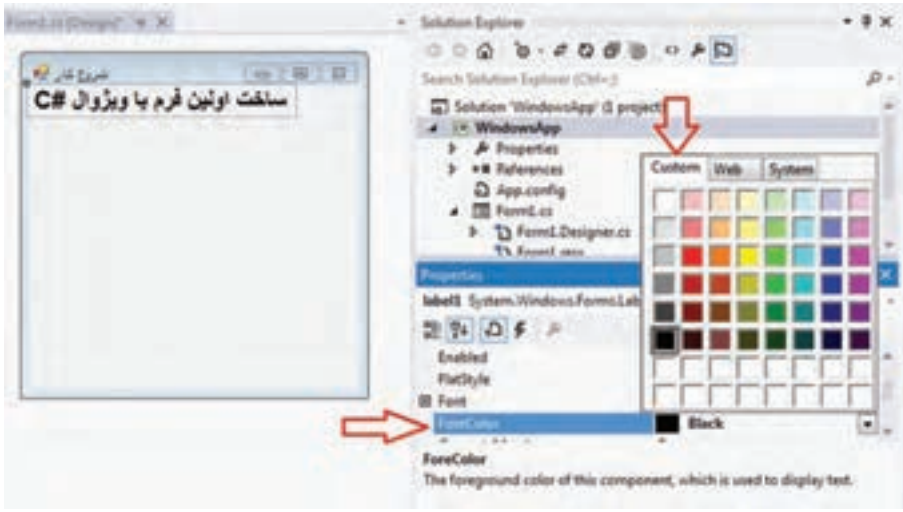
شکل ۳۸ــ۵ــ تغییر ویژگی RightToLeft

همان‌طور که در شکل ۳۸ــ۵ مشاهده می‌کنید، تنها اشکال باقیمانده، برعکس بودن عبارت "C#" است که برای تصحیح آن به خاصیت Text بروید و آن را به صورت "C#" تایپ کنید.

۵ــ تعیین ویژگی‌های فونت ، رنگ و محل قرارگیری برچسب : برای تعیین نوع فونت کنترل برچسب، از خاصیت Font در پنجره Properties استفاده کنید و روی علامت سه نقطه کلیک کنید تا لیست فونت‌های موجود در کامپیوتر ظاهر شود. پنجره‌ای مانند شکل ۳۹ــ۵ مشاهده می‌شود که در آن می‌توانید، فونت دلخواه را به همراه اندازه و سبک آن انتخاب کنید. سعی کنید، فونتی را انتخاب کنید که نوشته‌های فارسی خوب دیده شوند.

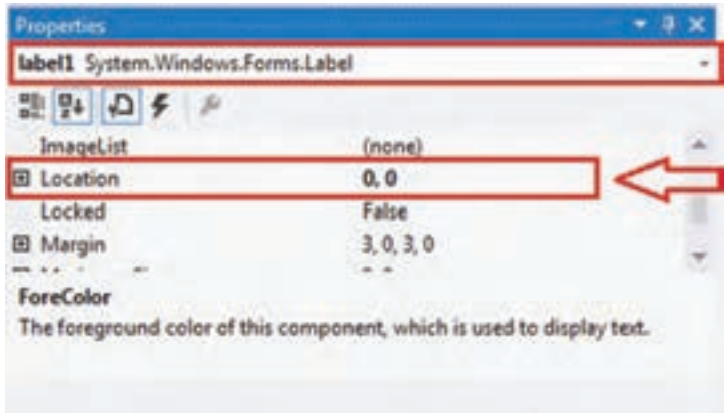
شکل ۳۹ــ۵ــ پنجره فونت

همچنین برای تعیین رنگ نوشته از خاصیت ForeColor استفاده می‌کنیم. پس از کلیک بر روی علامت فلش ، پنجره ای برای تعیین رنگ باز می‌شود، در آن پنجره مطابق شکل ۴۰ــ۵ ، بر روی سربرگ Custom کلیک کنید.



شکل ۴۰ــ۵ ــ پنجره رنگ

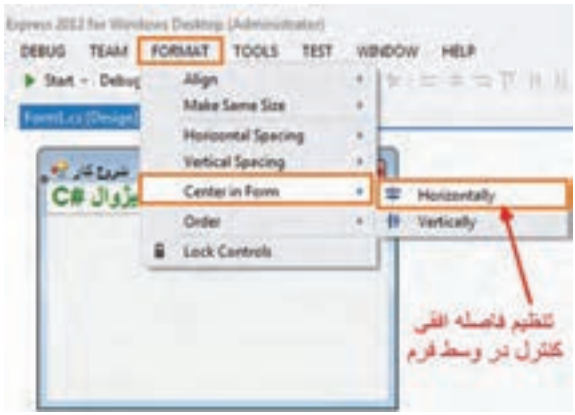
حال می‌خواهیم محل قرارگیری کنترل را تنظیم کنیم. قبل از انجام این‌کار که به سادگی با استفاده از ماوس می‌توانید انجام دهید، به پنجره Properties مراجعه کنید و ویژگی Location را مطابق شکل ۴۱ــ۵ پیدا کرده، روی علامت + کنار آن کلیک کنید.



شکل ۴۱ــ۵ ــ ویژگی موقعیت

در جلوی ویژگی Location، دو عدد ۰ و ۰ نوشته شده است. این اعداد فاصله کنترل را از سمت چپ فرم و از بالای فرم بر حسب پیکسل[1] مشخص می‌کنند. به عبارت دیگر موقعیت یک کنترل نسبت به گوشه بالا سمت چپ[2] فرم سنجیده می‌شود. بر خلاف آنچه که در ریاضیات معمول است که موقعیت یک نقطه را نسبت به مرکز صفحه (مبدأ) در نظر می‌گیرند، در کامپیوتر مبدأ سنجش موقعیت، گوشه بالا سمت چپ صفحه است.

برای تنظیم محل قرارگیری کنترل می‌توانید از ماوس استفاده کنید و یا اعداد مناسبی را به عنوان فاصله از چپ و بالا مشخص کنید. در اینجا با استفاده از ماوس، کنترل برچسب را گرفته و در نقطه دلخواه فرم قرار دهید به‌طوری که از دو طرف فرم به یک اندازه باشد. همچنین می‌توانید مانند شکل ۴۲ــ۵، از منوی Format گزینه Center in Form استفاده کنید تا با دقت بیشتری کنترل را در وسط فرم قرار دهید.



شکل ۴۲ــ۵ــ تنظیم موقعیت افقی با استفاده از منوی Format

نکته

هنگامی‌که بیش از یک کنترل در روی فرم قرار داشته باشد، اگر آنها را با هم انتخاب کنید، ابزارهای تنظیم موقعیت آنها نسبت به یکدیگر که در روی نوار ابزار و همچنین در منوی Format وجود دارد فعال شده، به سادگی و با دقت و سرعت عمل بالا می‌توانید آنها را تنظیم کنید.

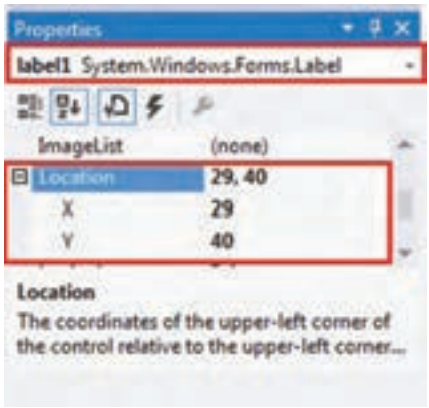
شکل ۴۳ــ۵

۱ــ Pixel

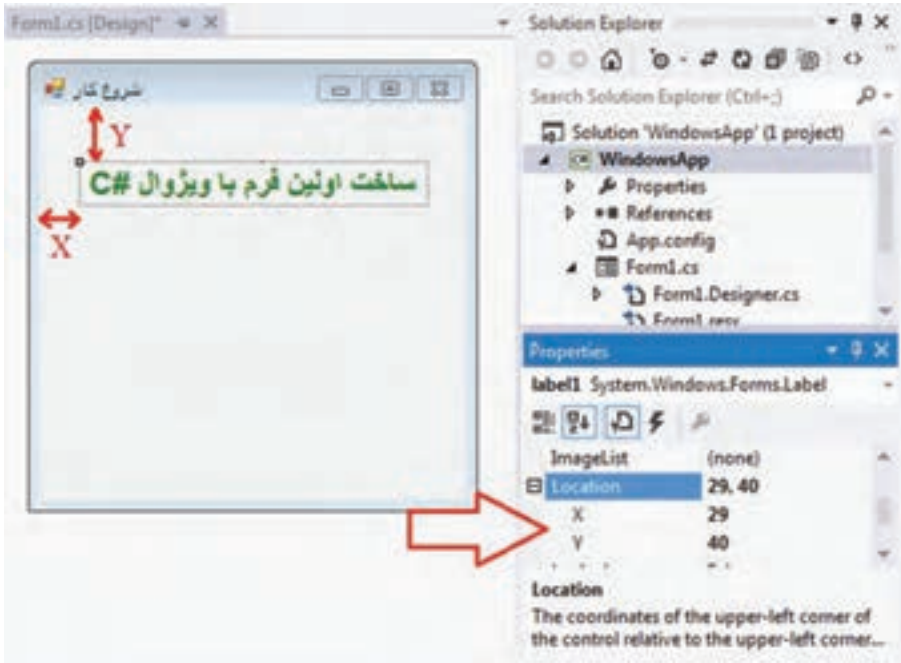
۲ــ Upper_left corner

بعد از جا به جایی کنترل برچسب و قرارگرفتن در یک موقعیت مناسب، به پنجره Properties رفته و مقدار اعداد در ویژگی Location را مشاهده کنید.

همان‌طور که در شکل ۴۴ــ۵ مشاهده می‌کنید، در اثر جا به جایی کنترل، مقدار ۰ و ۰ به اعداد دیگری مانند ۴۰ ,۲۹ تغییر کرده است. ارتباط این دو عدد با محل قرار گیری برچسب نسبت به فرم در شکل ۴۵ــ۵ نشان داده شده است. فاصله افقی[1] نسبت به لبه سمت چپ فرم با مقدار X و فاصله عمودی[2] نسبت به لبه بالای فرم با مقدار Y مشخص می‌گردد.



شکل ۴۴ــ۵ــ مقادیر ویژگی موقعیت کنترل برچسب



شکل ۴۵ــ ۵ــ ارتباط مقادیر ویژگی موقعیت با لبه‌های فرم

۱ــ Horizontally

۲ــ Vertically

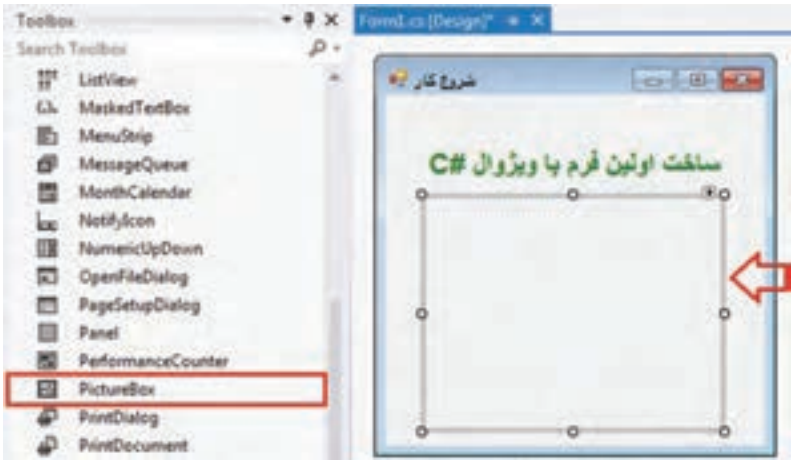
قبل از رفتن به مرحله بعدی بهتر است یک بار ویژگی‌هاو ویژگی‌های کنترل برچسب را که تنظیم کرده ایم در جدول ۱ــ۵ مشاهده کنید. رنگ نوشته و نوع فونت ممکن است با آنچه شما در انجام این مثال انجام می‌دهید، متفاوت باشد که اهمیتی ندارد.

جدول ۱ــ۵ــ مقادیر ویژگی‌های کنترل برچسب

کنترل Label	
ویژگی	**مقدار**
Name	label1
Text	ساخت اولین فرم با ویژوال C#
RightToLeft	Yes
Font	Arial Size : 16 Bold
ForeColor	Green
Location	Center in Form | Horizontally

در سطر اول جدول ۱ــ۵، نام کنترل نوشته شده است و در ستون اول نام ویژگی‌های کنترل و در ستون دوم مقدار هر ویژگی بیان شده است. با توجه به اینکه جدول ۱ــ۵ به طور واضح و گویا ویژگی‌های کنترل برچسب را نشان می‌دهد، از چنین جداولی در انجام پروژه‌های بعدی و یا حتی در امتحانات عملی، برای مشخص کردن ویژگی‌های هر کنترل که باید تنظیم شود، استفاده خواهیم کرد و دیگر لازم به ارائه توضیحات نیست.

۶ــ اضافه کردن عکس بر روی فرم : در این مرحله باید عکسی را به فرم اضافه نماییم. برای نمایش یک عکس می‌توانید از کنترل PictureBox، استفاده کنید که یک کنترل جالب و کاربردی است و آن را کنترل جعبه تصویر می‌نامیم. برای قراردادن این کنترل بر روی فرم، به جعبه ابزار مراجعه کرده ، آن را پیدا کرده و به فرم اضافه کنید. شکل ۴۶ــ۵ را مشاهده کنید.

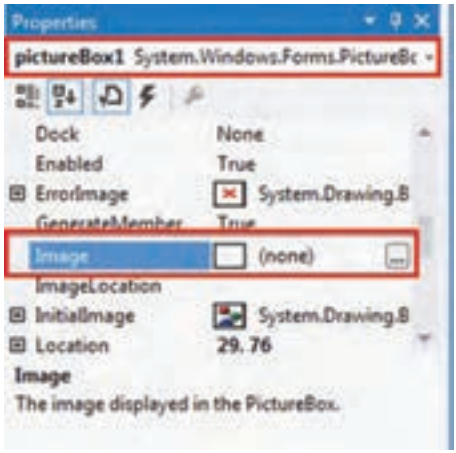


شکل ۴۶ــ۵ ــ اضافه کردن کنترل جعبه تصویر به فرم

به محض قرار گیری این کنترل بر روی فرم، پنجره Properties به طور خودکار، ویژگی‌ها این کنترل را نشان می‌دهد. نام این شئ pictureBox1 است که به وسیلهٔ VS انتخاب می‌شود و از کلاس PictureBox ساخته شده است.

برای تعیین تصویری که می‌خواهیم در جعبه تصویر نمایش داده شود، از ویژگی Image این کنترل استفاده می‌کنیم. برای مقداردهی ویژگی Image دو روش وجود دارد :

۱ــ استفاده از پنجره Properties، انتخاب شئ PictureBox و سپس کلیک روی علامت... (شکل ۴۷ــ۵).



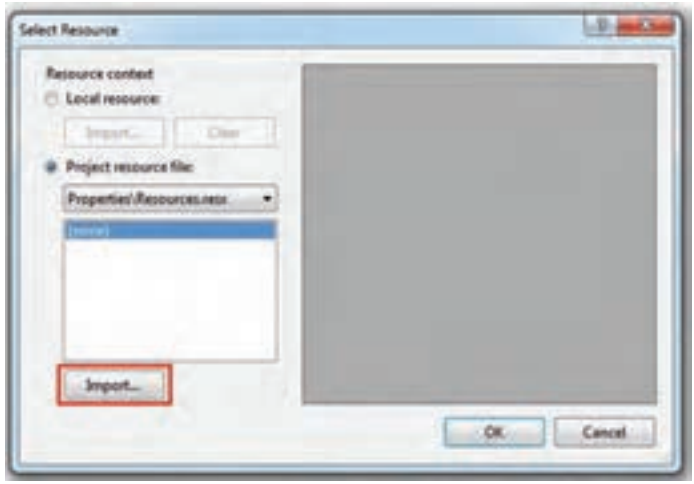
شکل ۴۷ــ۵ــ ویژگی Image برای تعیین تصویر

۲ــ کلیک روی مثلث کوچک سیاه رنگ در کادر اطراف کنترل جعبه تصویر و استفاده از گزینه Choose Image

در کادر اطراف کنترل جعبه تصویر، یک مثلث کوچک سیاه رنگ دیده می‌شود. با کلیک بر آن، منویی مانند شکل ۴۸ــ۵ ظاهر می‌گردد که در آن چند ویژگی کنترل نشان داده می‌شود.

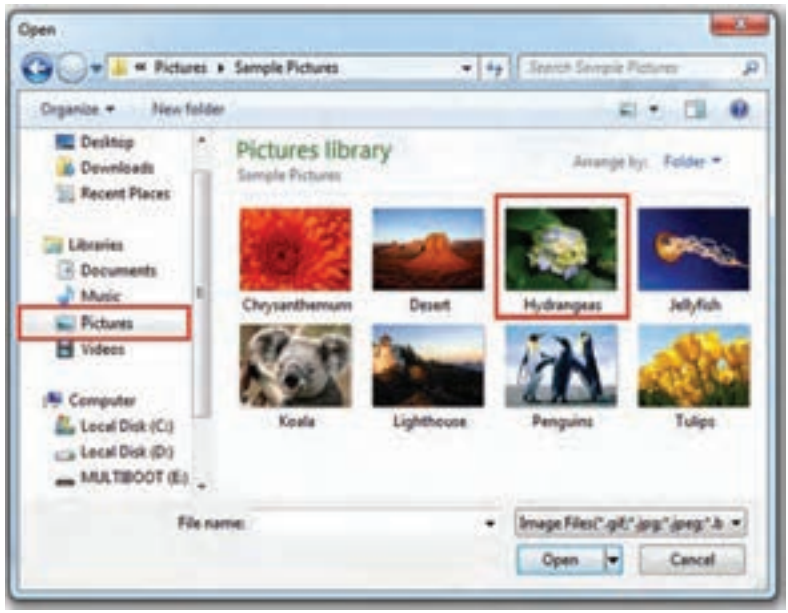
شکل ۴۸ــ۵ ــ کلیک بر روی مثلث و ظاهر شدن منو

به وسیلهٔ گزینه Choose Image در شکل ۴۸ــ۵ نیز علاوه بر پنجره Properties، می‌توانید ویژگی Image را تعیین کنید. در هر حال با انتخاب گزینه Choose Image، پنجره‌ای به نام انتخاب منبع (Select Resource)، شکل ۴۹ــ۵ باز می‌شود که در آن باید فایل مورد نظر برای اضافه‌شدن به پروژه را مشخص کنید. اگر در شکل ۴۹ــ۵ بر روی کلید Import کلیک کنید پنجره انتخاب فایل باز می‌شود تا یک کپی از فایل مورد نظر شما به لیست فایل‌های پروژه اضافه شود.



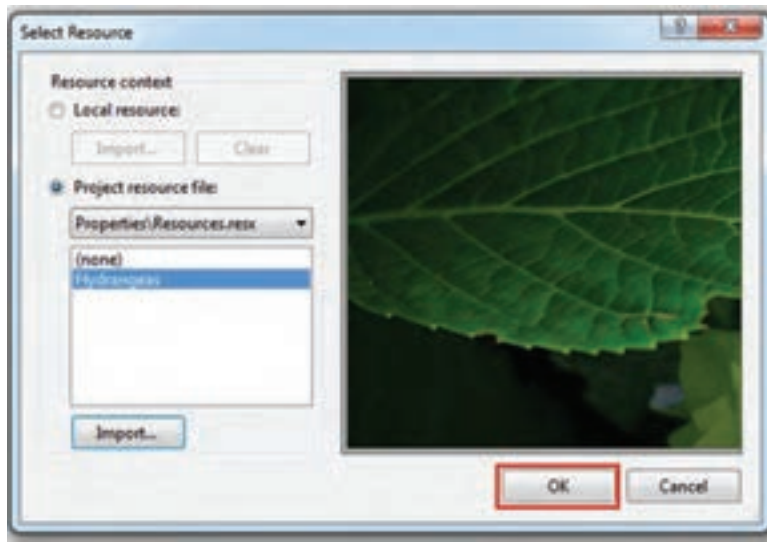
شکل ۴۹ــ۵ ــ انتخاب فایل تصویری جهت اضافه شدن به پروژه

پس از اینکه پنجره مربوط به تعیین نام فایل ظاهر شد وارد پوشهٔ Pictures شوید و یک فایل تصویری انتخاب کنید. در این مثال تصویر گل ادریسی انتخاب شده است.



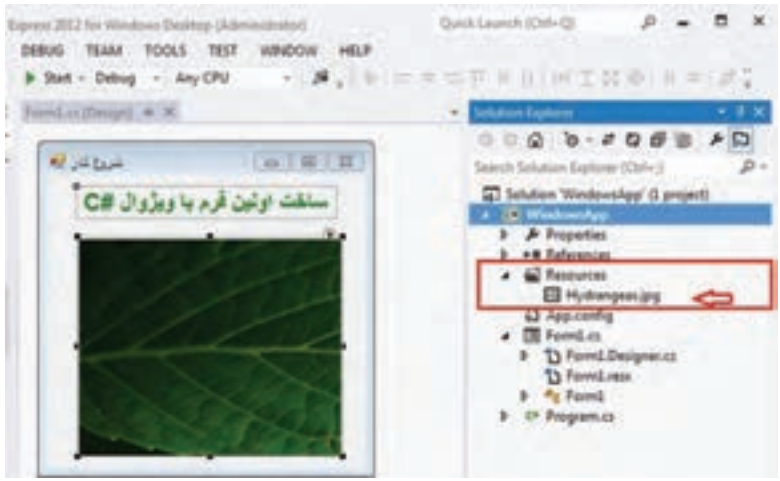
شکل ۵۰ــ۵ــ انتخاب فایل تصویری جهت نمایش در کنترل جعبه تصویر

پس از انتخاب فایل مورد نظر، به پنجره انتخاب منابع باز می‌گردید. با کلیک روی دکمه OK عملیات انجام شده را تأیید کنید (شکل ۵۱ ــ۵).



شکل ۵۱ــ۵ــ پنجره انتخاب منبع پس از انتخاب فایل تصویری

پس از انتخاب فایل تصویری و کلیک بر روی دکمه OK ، تصویر انتخاب شده در جعبه تصویر قرار می‌گیرد. اگر مانند این مثال، اندازه عکس، بزرگ‌تر از اندازه جعبه تصویر باشد، فقط بخشی از آن مانند شکل ۵ــ۵۲ دیده خواهد شد. نگران نباشید این مشکل را برطرف می‌کنیم!



شکل ۵ــ۵۲ــ اضافه شدن فایل تصویری در فولدر Resources

اگر به پنجره Solution Explorer در شکل ۵ــ۵۲ ، دقت کنید فایل تصویر انتخاب شده در پوشه Resources دیده می‌شود. اگر از طریق My Computer به مسیری بروید که در ابتدا، برای پروژه تعیین کرده‌اید، پوشه‌ای به نام Resources را پیدا خواهید کرد که در داخل آن یک کپی از فایل تصویر انتخاب شده قرار دارد. به همین دلیل به این پوشه ، پوشه منابع گفته می‌شود.

برای برطرف کردن مشکل عدم تطبیق اندازه عکس و اندازه جعبه تصویر چه کنیم؟

شاید راه حل‌های مختلفی برای برطرف کردن اِشکال فوق، از جمله کاهش اندازه عکس به وسیلهٔ برنامه‌های گرافیکی نظیر Paint و Photoshop و یا بزرگ کردن اندازه جعبه تصویر به طوری که بتواند تمام عکس را در خود جای دهد، به نظر برسد. اما بهترین راه حل این است که از ویژگی SizeMode کنترل جعبه تصویر استفاده کنیم و مقدار آن را مانند شکل ۵ــ۵۳ برابر StretchImage قرار دهیم.

شکل ۵۳ــ۵ــ تغییر خاصیت SizeMode به StretchImage

حال برنامه را اجرا می‌کنیم ، خروجی آن مانند شکل ۵۴ ــ ۵ است.

شکل ۵۴ــ۵ــ خروجی اولین برنامه نوشته شده در محیط Visual C#

تبریک می‌گوییم که توانستید اولین برنامه ویندوزی با استفاده از VS را بنویسید. در این مثال یک خط برنامه ننوشتید و همه دستورات این برنامه به وسیلهٔ VS به طور خودکار انجام شد. بر روی فایل Form1.Designer.cs دوبار کلیک کنید و مستقیم به سراغ متد ()InitializeComponent بروید و روی علامت + کلیک کنید تا محتوای آن را ببینید. توجه داشته باشید که محتویات این فایل، بخصوص این متد را نباید بدون اطمینان و آگاهی تغییر دهید و یا دستکاری کنید. شکل ۵۵ــ۵ را مشاهده کنید. دستورات مربوط به برچسب و جعبه تصویر مشخص شده است. دستورات قسمت پایین در شکل ۵۵ــ۵ مربوط به چیست؟

```
/// Required method for Designer support. do not modify
/// the contents of this method with the code editor. محتویات این متد را با ادیتور تغییر ندهید
/// </summary>
private void InitializeComponent()
{
    this.label1 = new System.Windows.Forms.label();
    this.pictureBox1 = new System.Windows.Forms.pictureBox();
    ((System.ComponentModel.IsupportInitialize)(this.pictureBox1)).BeginInit();
    this.SuspendLayout();
    //label1
    //
    this.label1.AutoSize = true;
    this.label1.Font = new System.Drawing.Font("Arial", 15.75F, System.Drawing.
    this.label1.ForeColor = System.Drawing.Color.Green;
    this.label1.Location = new System.Drawing.Point(12,13);
    this.label1.Name = "label1";
    this.label1.RightTOLeft = System.Windows.Forms.RightTOLeft.Yes;
    this.label1.Size = new System.Drawing.Size(227,24);
    this.label1.TabIndex = 0;
    this.label1.Text = "ساخت اولین فرم با ویژوال #C";
    //
    // picturBox1
    //
    this.pictureBox1.Imag=global::WindowsApp.Properties.Resources.Hydrangeas
    this.pictureBox1.Location = new System.Drawing.Point(12, 57);
    this.pictureBox1.Name = "pictureBox1";
    this.pictureBox1.Size = new System.Drawing.Size(233, 193);
    this.pictureBox1.SizeMode=System.Windows.Forms.PictureBoxSizeMode.Stretcr
    this.pictureBox1.TabIndex = 1;
    this.pictureBox1.TabStop = false;
    //
    // Form1
    //
```

مقداردهی ویژگی‌های برچسب

مقداردهی ویژگی‌های جعبه تصویر

```
this. AutoScaleDimensions = new System. Drawing. SizeF (6F, 13F);
this. AutoScaleMode = System.Windows. Forms. AtoScaleMode.Font;
this. ClientSize = new System. Drawing. Size (284, 262);
this. Controls. Add (this.pictureBox1);
this. Controls. Add (this.label1);
this. Name = "Form1" ;
this. Text = «شروع کار» ;
((System. ComponentMode 1. IsupportInitialize) (this.pictureBox1)). EndInit ( );
this. ResumeLayout (false);
this. PerformLayout ( );
```

شکل ۵۵ــ۵ــ محتویات متد InitalizeComponent() در فایل Form1.Designer.cs

برای مطالعه

کنترل‌ها و ابزارهای شخص ثالث برای.NET

علاوه بر کنترل‌هایی که در جعبه ابزار VS وجود دارد، کنترل‌های دیگر و ابزارهای پیشرفته‌ای نیز توسط شرکت‌های نرم‌افزاری که با شرکت مایکروسافت همکاری دارند، ساخته شده‌اند. به این نوع کنترل‌ها که قابلیت اضافه شدن به فرم‌های ویندوزی را دارند و توسط شرکت دیگری ساخته می‌شود، کنترل‌های شخص ثالث نامیده می‌شوند. بعضی از این ابزارها و کنترل‌ها در محیط ویژوال استودیو اضافه می شوند.

شرکت DevExpress از جمله شرکت‌هایی است که مجموعه کنترل‌ها و ابزارهای واسطه کاربری را برای تولیدکنندگان برنامه‌های کاربردی، از جمله برنامه‌نویسانی که از ویژوال استودیو استفاده می‌کنند، فراهم و به صورت تجاری عرضه کرده است. با استفاده از این کنترل‌ها، واسطه‌های کاربری زیباتری، بدون کدنویسی می‌توان ایجاد کرد.

یکی از ابزارهایی که این شرکت به رایگان عرضه می‌نماید، CodeRush Express است

که می‌توانید آن را از آدرس زیر دانلود نمایید. با نصب این ابزار در محیط VS، امکانات بیشتری در هنگام برنامه‌نویسی فراهم می‌شود. مثلاً منوی Refactoring با گزینه‌های متنوعی را در اختیار خواهید داشت.

https: //www.devexpress.com/Products/CodeRush/

کار در کارگاه ۳: تنظیم اندازه کنترل

تنظیم خودکار اندازه کنترل‌ها در هنگام اجرای برنامه

۱ــ پروژه WindowsApp که در این فصل ساختید را اجرا کنید.

شکل ۵۶ــ۵ ــ تغییر اندازه پنجره برنامه در حال اجرا

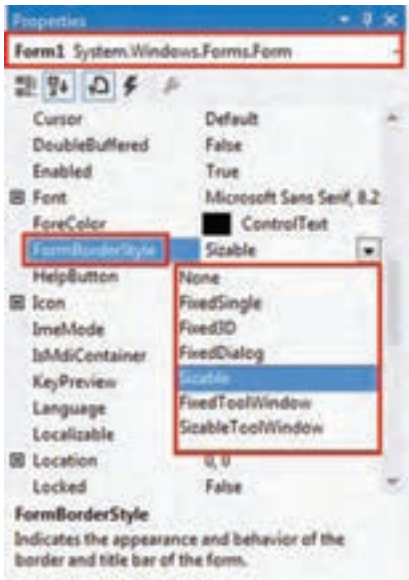
۲ــ همان‌طورکه در شکل ۵۶ــ۵ مشاهده می‌کنید، با استفاده از ماوس گوشه سمت راست پایین پنجره برنامه را گرفته و به سمت راست بکشید تا پنجره برنامه بزرگ‌تر شود.

شکل ۵۷ ــ ۵

همان‌طور که در شکل ۵ــ۵۷ مشاهده می‌کنید، در اثر تغییر اندازه پنجره، فضای خالی در سمت راست ایجاد می‌شود. اما کاربران انتظار دارند با بزرگ شدن پنجره، اندازه تصویر نیز افزایش یابد و صفحه پر شود. ولی در برنامه بالا چنین نیست و توازن صفحه به هم می‌خورد.

برای رفع اشکال بالا، دو راه حل داریم :

الف) کاربر اجازه تغییر اندازه پنجره برنامه را نداشته باشد. در واقع صورت مسئله را پاک می‌کنیم و به کاربر اجازه نمی‌دهیم که اندازه فرم را تغییر دهد تا چنین مشکلی به وجود آید. فرم‌ها دارای ویژگی به نام FormBorderStyle هستند که به وسیلهٔ آن می‌توانیم سبک یا شکل ظاهری کادر دور فرم را تعیین کنیم. در بعضی از سبک‌ها از امکان تغییر اندازه پنجره در حین اجرای برنامه جلوگیری می‌شود. مقدار پیش فرض این ویژگی Sizable است، یعنی فرم قابل تغییر اندازه است. برای انتخاب سبک‌های دیگر در حالی که فرم انتخاب شده است، در پنجره Properties به سراغ ویژگی FormBorderStyle رفته (شکل ۵ــ۵۸) و مقدار دیگری مانند FixedSingle که به معنای اندازه ثابت فرم و با کادر تک خطی می‌باشد را انتخاب کنید.



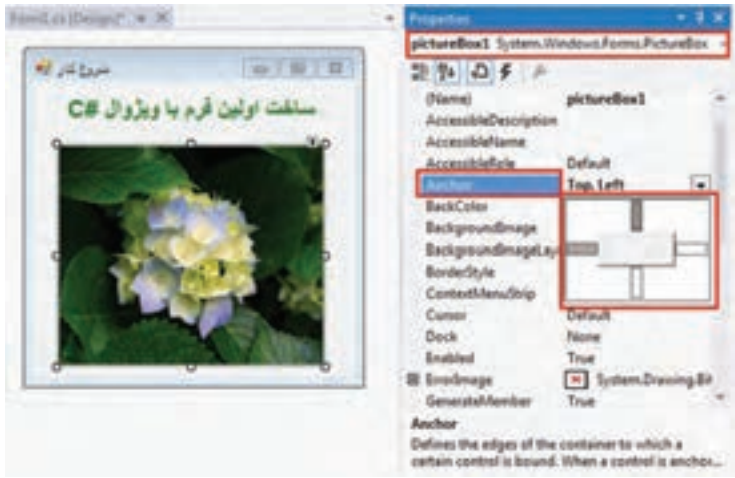
شکل ۵ــ۵۸ــ ویژگی FormBorderStyle فرم

نکته

می‌توانید منوی کنترل پنجره و آیکن‌های بیشینه، کمینه و خروج را پنهان کنید . برای این منظور ویژگی ControlBox مربوط به فرم را برابر false کنید. البته در این صورت کاربر، مجبور است برای پایان دادن به اجرای برنامه از ترکیب کلیدی Alt +F4 استفاده کند.

ب) در روش دوم محدودیتی برای کاربر ایجاد نکرده و مانع تغییر اندازه فرم نمی‌شویم. بنابراین باید اندازه جعبه تصویر در هنگام تغییر اندازه پنجره، به طور خودکار افزایش یابد. و به عبارت جالب‌تر، باید فاصله جعبه تصویر از چهار لبه فرم همواره ثابت بماند. خیلی نگران نباشید در VS برای انجام این

کار نیازی به برنامه نویسی خود شما نیست و فقط کافی است در محیط طراحی، از ویژگی Anchor کنترل جعبه تصویر برای این منظور استفاده کنید. این ویژگی‌ها می‌تواند مقادیر Left , Right , Top , Bottom و یا ترکیبی از آنها را بپذیرد. هر یک از مقادیر باعث می‌شوند که فاصله کنترل از سمت مربوطه ثابت بماند. در حالت پیش فرض مقدار ویژگی Anchor کنترل جعبه تصویر برابر Left, Top است، بنابراین فاصله کنترل از دو طرف چپ و بالای فرم ، ثابت باقی می‌ماند. شکل ۵۹ــ۵ را مشاهده کنید.



شکل ۵۹ــ۵ــ ویژگی Anchor

شکل ۶۰ــ۵ــ تغییر اندازه پنجره برنامه در حال اجرا پس از تنظیم ویژگی Anchor کنترل pictureBox

اگر در ویژگی Anchor مربوط به جعبه تصویر هر چهار مقدار را قرار دهید (از ماوس کمک بگیرید و روی دو کادر خط چین راست و پایین در شکل ۵۹ــ۵ کلیک کنید تا رنگ آنها خاکستری شود)، فاصله آن از چهار طرف ثابت می‌ماند. به عبارت دیگر اندازه آن هماهنگ با افزایش اندازه پنجره تغییر می‌کند. شکل ۶۰ــ۵ پنجره برنامه را پس از افزایش توسط کاربر نشان می‌دهد.

۳ــ در شکل فوق مشاهده می‌کنید که برچسب در وسط پنجره قرار ندارد. برای اینکه موقعیت برچسب نیز بتواند با تغییر اندازه پنجره تغییر کند و همواره در وسط باقی بماند لازم است فقط فاصله

آن تا بالای پنجره ثابت باشد. البته توجه داشته باشید برچسب باید از ابتدا در وسط فرم باشد. بنابراین برچسب را انتخاب کرده و ویژگی Anchor آن را برابر Top قرار دهید تا به طور خودکار موقعیت آن همواره در وسط پنجره قرار گیرد.

۴ــ برنامه را اجرا کنید و اندازه پنجره را تغییر دهید. آیا اندازه تصویر و موقعیت برچسب نیز مطابق با آن به طور خودکار تغییر می‌کند و کار رضایت بخش است؟

۵ــ در خیلی از برنامه‌های کاربردی نظیر Word و Excel که در سال قبل آشنا شدید، برای وارد کردن اطلاعات به زبان فارسی و یا نمایش صحیح اطلاعات فارسی، جهت صفحه را به حالت *راست به چپ*[1] تغییر می‌دادید. در فرم‌ها و کنترل‌هایی که برای نمایش و یا دریافت اطلاعات فارسی به‌کار می‌روند نیز بهتر است ظاهر و ترتیب قرارگیری اجزای فرم و همچنین ترازبندی از راست به چپ تنظیم شوند. مثلاً در شکل ۶۱ــ۵ خط عنوان فرم به صورت راست به چپ تنظیم شده است.

شکل ۶۱ــ۵ ــ خط عنوان از راست به چپ

همان‌طور که در شکل مشاهده می‌کنید، خط عنوان فرم تغییر کرده است و حالت آینه‌ای نسبت به فرم‌های معمولی دارد. در این فرم، آیکن‌های بستن پنجره ، بیشینه و کمینه در جهت سمت چپ قرار گرفته‌اند.

در فرم‌های ویندوزی خاصیت RightToLeft و خاصیت RightToLeftLayout برای پشتیبانی از زبان‌های راست به چپ ، پیش بینی شده است. برای ایجاد چنین حالتی، باید مقدار هر دو خاصیت را به true تغییر دهید.

---

۱ ــ Right to Left

خودآزمایی فصل پنجم

الف) درستی یا نادرستی هر عبارت را تعیین کنید.

۱ــ برنامه‌های کنسولی کاربرد زیادی در سیستم عامل ویندوز دارند.

۲ــ در برنامه‌های ویندوزی، واسط کاربری به صورت گرافیکی است.

۳ــ برای نمایش یک متن روی فرم از کنترل Label استفاده می‌کنیم.

۴ــ عنوان یک فرم از طریق ویژگی Text مقداردهی می‌شود.

۵ــ می‌توان منوی کنترل پنجره و آیکن‌های بیشینه، کمینه و خروج را پنهان کرد.

ب) در سؤالاتِ چند گزینه‌ای زیر، پاسخ صحیح را انتخاب نمایید.

۶ــ کدام ویژگی یک برچسب مربوط به متن داخل آن است؟

الف) RightToLeft　　ب) Text

ج) Enabled　　د) TextAlign

۷ــ برای تغییر ویژگی‌های یک کنترل از پنجره.....استفاده می‌کنیم.

الف) Solution Explorer　　ب) Properties

ج) Toolbox　　د) Class View

۸ــ برای دیدن پروژه و فایل‌های آن از پنجره..... استفاده می‌شود

الف) Solution Explorer　　ب) Properties

ج) Toolbox　　د) Class View

۹ــ اگر بخواهیم در محیط طراحی، یک کنترل را به فرم اضافه کنیم از پنجره..... آن را به داخل فرم می‌آوریم.

الف) Solution Explorer　　ب) Properties

ج) Toolbox　　د) Class View

۱۰ــ ویژگی SizeMode برای تنظیم چه چیزی از کنترل PictureBox به کار برده می‌شود؟

الف) مکان تصویر　　ب) تصویر درون کنترل

ج) نوع تعیین اندازه تصویر　　د) اندازه کنترل

۱۱ــ برای ثابت نگه داشتن فاصله از طرفین برای یک کنترل از کدام ویژگی استفاده می‌نماییم؟

الف) Dock　　ب) Text

ج) Location　　د) Anchor

۱۲ــ اگر ویژگی FormBorderStyle برابر با Sizable شود فرم..... خواهد بود.

الف) با اندازه ثابت　　ب) یک کادر محاوره‌ای

ج) بدون دکمه‌های بیشینه و کمینه　　د) با اندازه متغیر

ج) جاهای خالی را با عبارات مناسب پرکنید.

۱۳ــ اطلاعی که از طرف سیستم عامل به برنامه داده می‌شود.....نامیده می‌شود.

۱۴ــ ویژگی RightToLeft برای زبان‌های..... پیش‌بینی شده است

۱۵ــ در کنترل PictureBox، ویژگی که تصویر را در بر می‌گیرد.....است.

تمرینات برنامه نویسی فصل پنجم

۱ــ فرم زیر را برای نمایش اطلاعات شخصی خود (نام ، نام خانوادگی، نام پدر، تاریخ تولد و عکس) طراحی کنید.

۲ــ فرم تمرین ۱ برای نمایش اطلاعات به زبان فارسی که یک زبان RTL است، مناسب نیست. با تغییر ویژگی‌های فرم و برچسب‌ها، فرم را برای زبان RTL مناسب کنید.

فعالیت

متن زیر از MSDN با موضوع برنامه‌های ویندوزی برداشت شده است. آن را با کمک هم‌کلاسی خود ترجمه کنید و به کلاس ارائه دهید.

Windows Forms provide your project with components, such as dialog boxes, menus, buttons, and many other controls, that make up a standard Windows application user interface (UI). Fundamentally, these controls are just classes from the .NET Framework class library. The Designer view in Visual C# Express Edition enables you to drag the controls onto your application's main form and adjust their size and position. As you do this, the IDE automatically adds the source code to create an instance of the appropriate class and initialize it.

واژگان و اصطلاحات انگلیسی فصل پنجم

ردیف	واژه انگلیسی	معنی به فارسی
۱	Action	
۲	Alphabetical view	
۳	Button	
۴	Categorized view	
۵	container	
۶	Context menu	
۷	Event	
۸	Form Designer	
۹	Graphical User Interface	
۱۰	Horizontally	
۱۱	Message	
۱۲	Notification	
۱۳	Object	
۱۴	Pixel	
۱۵	Pushpin	
۱۶	Right to Left	
۱۷	Right to left languages	
۱۸	Shortcut	
۱۹	Timer Control	
۲۰	Tool Bar	
۲۱	Tool Box	
۲۲	Upper–left corner	
۲۳	Vertically	
۲۴	Wait state	

فصل ششم

ایجاد برنامه‌های ویندوزی با واکنش نسبت به رویدادها

در فصل ۵ با برنامه‌های ویندوزی و کنترل‌های فرم، برچسب و جعبه تصویر آشنا شدید. در محیط VS برنامه‌های ویندوزی ایجاد کرده، فرم ساده طراحی کردید و بدون نوشتن حتی یک خط برنامه، آن را اجرا نمودید؛ در این فصل علاوه بر آشنایی با کنترل‌های جدید، برای رویدادها و عملیاتی که کاربر انجام می‌دهد برنامه نوشته می‌شود.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ در برنامه‌های ویندوزی از کنترل دکمه استفاده نماید و ویژگی‌های مورد نیاز را تنظیم نماید.

۲ــ مفهوم رویداد را توضیح دهد و در رویدادِ کلیکِ ماوس، برنامه بنویسد.

۳ــ برای واکنش به رویدادها Event Handler بنویسد.

۴ــ به برنامه‌های خود کنترل جعبه متن اضافه کند و ویژگی‌های آن را تعیین کند.

۵ ــ با استفاده از کنترل زمان سنج و تنظیم ویژگی‌های آن، برنامه‌های کاربردی بنویسد.

۶ــ با کنترل‌های Dialogs برای تغییر رنگ و قلم و تنظیمات پرونده، برنامه بنویسد.

مثال ۱ــ۶ : برنامه ویندوزی ساده با واکنش به رویداد کلیک

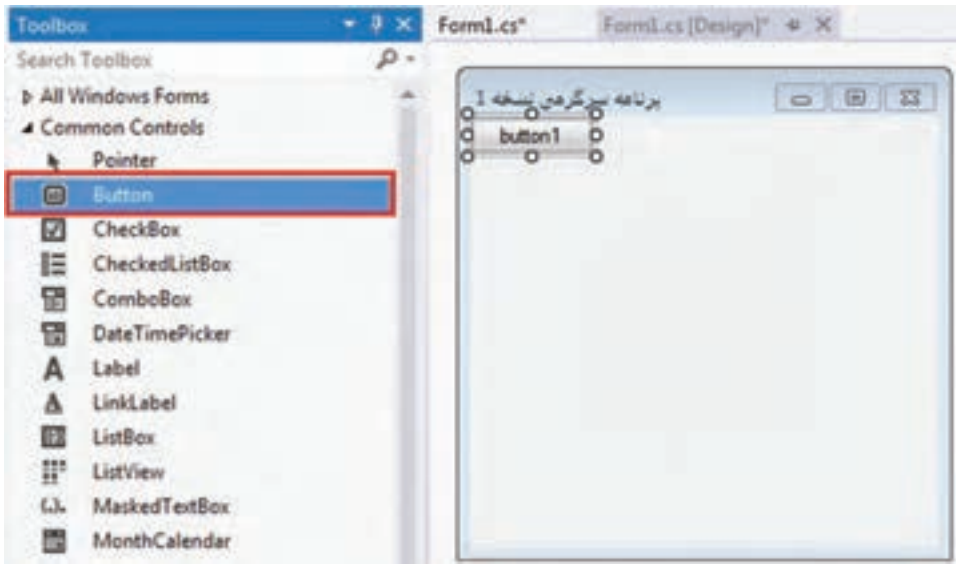
۱ــ یک پروژه جدید ویندوزی بسازید.

۲ــ ویژگی‌های فرم را مانند جدول ۱ــ۶ تنظیم کنید.

جدول ۱ــ۶ــ ویژگی‌های فرم

Form	
ویژگی	**مقدار**
Name	form1
Text	برنامه سرگرمی نسخه ۱
RightToLeft	Yes

۳ــ به جعبه ابزار مراجعه کرده و کنترل Button را پیدا کرده و آن را روی فرم قرار دهید. شکل ۱ــ۶ را مشاهده کنید.



شکل ۱ــ۶ــ قرار دادن کنترل Button روی فرم

۴ــ ویژگی‌های دکمه را مطابق جدول زیر تنظیم کنید.

جدول ۲ــ۶ــ ویژگی‌های Button

Button	
ویژگی	**مقدار**
Name	button1
Text	لبخند بزنید
RightToLeft	Yes
Font	Arial Size : 16
AutoSize	true
Location	Center in Form | Horizontally Center in Form | Vertically

۵ــ اکنون برنامه را اجرا کنید باید فرم ایجاد شده مانند شکل ۲ــ۶ باشد.

شکل ۲ــ۶

۶ــ با اجرای برنامه و کلیک کردن بر روی دکمه ، هیچ اتفاقی نمی‌افتد. نه تنها لبخندی در کار نیست شاید کاربر کمی هم دلگیر شود که چرا برنامه، کار خاصی انجام نمی‌دهد. در برنامه بعدی با روش ایجاد عکس العمل و واکنش نسبت به رویداد کلیک آشنا می‌شوید.

۷ــ گاهی اوقات علامت‌ها و نمادها می‌توانند خیلی بهتر از نوشته‌ها پیام خود را برسانند. مثلاً یک کنترل تلویزیون را در نظر بگیرید، در روی بعضی از دکمه‌های آن، علامت‌هایی حک شده است که به سرعت عملکرد کلید را نشان می‌دهد. در این مرحله می‌خواهیم به جای عبارت «لبخند بزن» یک نماد مثلاً ☺ را قرار دهیم.

۸ــ دو ویژگی Font و Text دکمه را که در جدول ۳ــ۶ با رنگ قرمز نشان داده شده است تغییر دهید. توجه داشته باشید که در ویژگی Text، حرف J بزرگ است.

جدول ۳ــ۶ــ ویژگی‌های Button

Button	
ویژگی	**مقدار**
Name	button1
Text	J
RightToLeft	No
Font	Wingdings Size : 28
AutoSize	True
Location	Center in Form | Horizontally Center in Form | Vertically

۹ــ اکنون برنامه را اجرا کنید. نتیجه کار باید شبیه شکل ۳ــ۶ باشد

سؤال: چگونه می‌توانید رنگ نماد لبخند، را مانند شکل ۳ــ۶ به رنگ سبز درآورید؟

شکل ۳ــ۶

۱۰ــ ویژگی Text دکمه را به حرف K (بزرگ) تغییر دهید و برنامه را اجرا کنید. اگر مایل هستید یک بار هم با حرف L (بزرگ) امتحان کنید. اگر به استفاده از این نمادها علاقمند هستید، وارد برنامه Word شوید و فونت Wingdings را قرار داده و با زدن کلیدهای صفحه کلید، نمادهای مختلفی را شناسایی کرده و از آنها در برنامه خود استفاده نمایید.

شکل ۴ــ۶ــ ویژگی‌های کنترل Button

۱۱ــ علاوه بر نوشته یا نماد، می‌توانید برای روشن شدن کاربرد یک دکمه، تصویری را نیز به عنوان زمینه دکمه با استفاده از ویژگی BackgroundImage معین کنید. اگر اندازه تصویر مورد نظر با اندازه دکمه متفاوت است، مقدار ویژگی BackgroundImageLayout را برابر با Stretch قرار داده تا کل تصویر در روی دکمه جای داده شود. توجه داشته باشید که اگر متنی را نیز در ویژگی Text نوشته باشید، متن روی عکس زمینه قابل مشاهده است. شکل ۴ــ۶ را مشاهده کنید.

۱۲ــ یکی دیگر از ویژگی‌هایی که در شکل ۴ــ۶ مشاهده می‌کنید، ویژگی Image است که برای انتخاب یک تصویر بر روی دکمه استفاده می‌شود. در صورت استفاده از چنین ویژگی، نوشته موجود در ویژگی Text دیده نمی‌شود چون تصویر روی آن را می‌پوشاند.

کار در کارگاه ۲: کار با دکمه

در این بخش خواهید دید که VS، در تولید برنامه‌هایی که نسبت به عملیات کاربر، واکنش نشان می‌دهد، بسیاری از دستورات و عملیات را در پشت صحنه انجام می‌دهد و برنامه نویس تنها با چند کلیک ماوس و نوشتن دستورات متد مربوطه، می‌تواند برنامه‌های واکنش دار را بسازد.

مثال ۲ــ۶ : ایجاد یک فرم شامل یک دکمه و واکنش نسبت به رویداد کلیک ماوس

در این مثال می‌خواهیم با اجرای برنامه، فرمی مطابق با شکل ۵ــ۶ نشان داده شود که ظاهراً فقط شامل یک دکمه است. برنامه در وضعیت انتظار باقی می‌ماند تا کاربر، عملی را انجام دهد.
هنگامی که کاربر بر روی دکمه «بگو سلام» کلیک کرد، پیامی مطابق با شکل ۶ــ۶ ظاهر می‌گردد.

شکل ۵ــ۶ــ برنامه در حالت انتظار

شکل ۶ــ۶ــ نمایش پیام به محض کلیک بر روی دکمه

مراحل اجرای این مثال را قدم به قدم به صورت زیر انجام می‌دهیم :

۱ــ **ایجاد پروژه** :وارد برنامه VS شوید و یک پروژه جدید از نوع Windows Forms Application با نام پروژه EventDemo و در مسیر مشخصی بسازید.

۲ــ تعیین ویژگی‌های فرم : ویژگی‌های فرم را به صورت جدول ۴ــ۶ تنظیم کنید.

جدول ۴ــ۶ ــ ویژگی‌های فرم

Form		
ویژگی		**مقدار**
Name		Form1
Text		واکنش به رویداد کلیک
Size	Width	300
	Height	300

۳ــ اضافه کردن یک متن نمایشی بر روی فرم : یک کنترل برچسب بر روی فرم قرار دهید و ویژگی‌های آن را مانند جدول ۵ــ۶ تعیین کنید.

جدول ۵ــ۶ــ ویژگی‌های برچسب

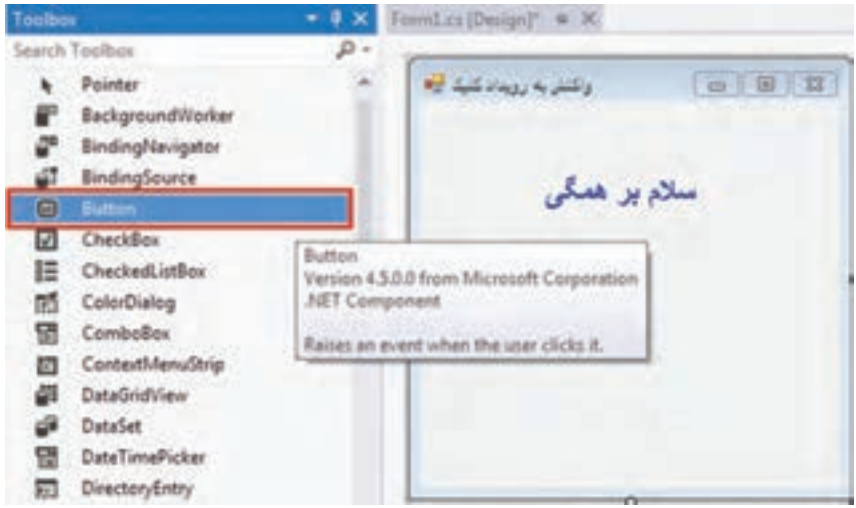
Label		
ویژگی		**مقدار**
Name		Label1
Text		سلام بر همگی
RightToLeft		True
Font		Arial Size : 16 Bold
ForeColor		Blue
Location	X	Center in Form | Horizontally
	Y	50
Visible		false

اگر برنامه را در این مرحله اجرا کنید، مشاهده خواهید کرد که فقط یک فرم خالی دیده می‌شود. شاید تعجب کنید که، با وجود اینکه در روی فرم، یک کنترل برچسب قرار دادید، اما در هنگام اجرای برنامه دیده نمی‌شود. فکر می‌کنید علت چیست؟

اگر به ردیف آخر جدول ۵ــ۶ دقت کنید، متوجه می‌شوید که یک ویژگی جدید به نام Visible استفاده شده است که مقدار آن را false قرار داده‌ایم. با توجه به معنی کلمه Visible (قابل مشاهده) و مقداری که برای این ویژگی تعیین کردیم، باید انتظار داشت که کنترل برچسب نشان داده نشود. هرگاه مقدار true در این ویژگی قرار دهیم کنترل برچسب نشان داده می‌شود.

حال طراحی برنامه را ادامه می‌دهیم و در این مرحله می‌خواهیم کاری کنیم تا کاربر با کلیک بر روی ماوس، برچسب را نمایان کند.

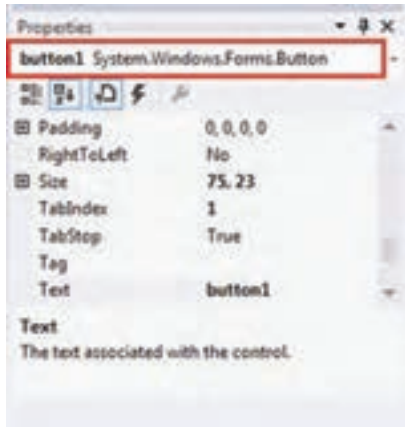
۴ــ اضافه کردن دکمه بر روی فرم : در این مرحله یک دکمه به فرم اضافه می‌کنیم برای این منظور به جعبه ابزار مراجعه و کنترل دکمه را مطابق شکل ۷ــ۶ پیدا کرده و به فرم اضافه کنید.



شکل ۷ــ۶ــ انتخاب کنترل دکمه از جعبه ابزار

به محض قرارگیری کنترل دکمه بر روی فرم، پنجره Properties به طور خودکار، ویژگی‌های این کنترل را نشان می‌دهد. نام شئ دکمه، button1 است که به وسیلهٔ VS انتخاب می‌شود. این شئ از کلاس Button ساخته شده است.

حال ویژگی‌های دکمه را مانند جدول ۶ــ۶ تنظیم کنید.



شکل ۸ــ۶ــ پنجره Properties برای نمایش ویژگی‌های دکمه

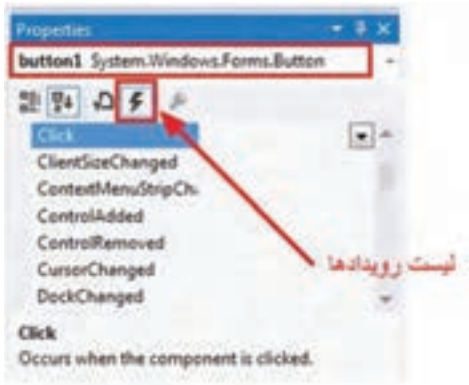
جدول ۶ــ۶ ــ ویژگی‌های Button

دکمه Button		
ویژگی		مقدار
Name		button1
Text		بگو سلام
Location	X	Center in Form | Horizontally
	Y	200

حال برنامه را اجرا و روی دکمه «بگو سلام» کلیک کنید، مشاهده می‌شود که هیچ اتفاقی نمی‌افتد. طبیعی است که برنامه عکس‌العملی نسبت به رویداد کلیک نداشته باشد چون هیچ متدی برای رویداد کلیک بر روی دکمه ایجاد نکردیم و همچنین هیچ دستوری برای نمایش پیام و تغییر ویژگی Visible برچسب نوشته نشده است. حال از برنامه خارج شده و به محیط طراحی فرم باز می‌گردیم.

در پنجره Properties علاوه بر لیست ویژگی‌ها، می‌توانید لیست رویدادهای مربوط به دکمه را نیز مشاهده کنید. با کلیک بر روی آیکون رعد و برق، مانند شکل ۹ــ۶ لیست رویدادها مشاهده خواهد

شد. مقدار تمام رویدادها خالی است، یعنی هیچ واکنشی نسبت به رویدادها ثبت نشده است.



شکل ۹ــ۶ ــ پنجره Properties برای نمایش رویدادهای دکمه

در ردیف اول لیست رویدادها، رویداد Click را مشاهده می‌کنید. این رویداد مربوط به کلیک ماوس است. در مرحله بعدی خواهید دید که چگونه واکنشی نسبت به این رویداد ثبت می‌کنیم.

۵ــ واکنش برنامه نسبت به رویداد کلیک ماوس: هنگامی‌که کاربر بر روی دکمه «بگو سلام»، کلیک کرد، باید پیام «سلام بر همگی» دیده شود. برای انجام این کار، باید دستوری اجرا گردد تا ویژگی Visible برچسب برابر true شود.

```
label1 . Visible = true;
```

سؤال: این دستور کجا باید نوشته شود؟

برای نوشتن برنامه‌هایی که باید به عمل کاربر واکنش نشان دهند، لازم است دو مرحله زیر انجام شود :

۱ــ نوشتن یک متد (EH)

۲ــ ثبت کردن رویداد ماوس و برقرار کردن ارتباط رویداد و متد

خوشبختانه VS در انجام دو عمل فوق، شما را تنها نمی‌گذارد و بیشتر عملیات را خودش انجام می‌دهد فقط بایستی دستورات داخل متد EH را بنویسیم.

مرحله اول : نوشتن متد EH : برای اینکه بتوانید متد را بنویسید در حالی که در صفحه طراحی فرم هستید، روی دکمه «بگو سلام» دوبار کلیک[۱] کنید. در این صورت پنجره کد نویسی مانند شکل ۱۰ــ۶ باز می‌شود که در آن دستورات مربوط به خط عنوان متد به طور خودکار نوشته شده است.

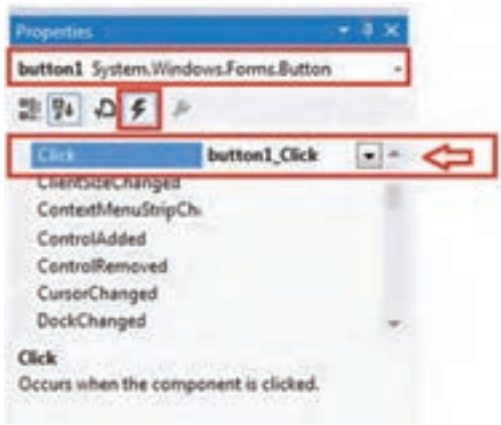
۱ــ Double Click

شکل ۱۰ــ۶ــ پنجره کد نویسی برای رویداد کلیک دکمه

نام متد، button1_Click است که به وسیلهٔ VS انتخاب شده است. فقط بایستی در داخل متد، دستورات مورد نظر را بنویسیم. بنابراین دستور را مانند برنامه ۱ــ۶ می‌نویسیم :

```
using System. Collections. Generic;
using System. ComponentMode 1;
using System. Drawing;
using System. Text;
using System. Windows. Forms;

namespace EventDemo
{
    public partial class Form1: Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void button1_ Click (object sender, EventArgs e)
        {
        label1. Visible = true;
        }
    }

}
```

برنامه ۱ــ۶ــ دستورات مربوط به رویداد کلیک دکمه



شکل ۱۱ــ۶ ــ تعیین متد EH مربوط به رویداد کلیک دکمه

مرحله دوم : ثبت متد EH برای رویداد کلیک : باید متد EH را برای یک رویداد ثبت کنیم. خوشبختانه این کار را VS به طور خودکار انجام می‌دهد و شما هیچ کاری لازم نیست انجام دهید. در این برنامه، VS به طور خودکار متد button1_Click() را برای رویداد Click دکمه ثبت می‌کند. برای اینکه مطمئن شوید که این کار صورت گرفته، کافی است به صفحه طراحی فرم برگشته و در پنجره Properties بر روی آیکون رعد و برق کلیک نمایید تا لیست رویدادها ظاهر گردد. در لیست به رویداد Click توجه کنید. در جلوی رویداد Click، نام متدی نوشته شده است که در آن متد، دستور ظاهر کردن قرار دارد. شکل ۱۱ــ۶ را مشاهده کنید.

اکنون برنامه را اجرا کنید و رویداد کلیک را آزمایش نمایید. از برنامه خارج شده و آن را مجدداً اجرا نمایید.

سؤال: چگونه از طریق پنجره Properties ، می‌توانید واکنش نسبت به رویداد Click دکمه را از کار بیاندازید؟

تبریک می‌گوییم که توانستید اولین برنامه ویندوزی با واکنش نسبت به کلیک ماوس را با همیاری VS بنویسید. در این مثال فقط یک خط دستور نوشتید و بقیه عملیات را با ماوس انجام دادید.

توسعه و بهبود برنامه : یک جعبه تصویری به فرم قبل اضافه کنید به طوری که بین پیام و دکمه مانند شکل ۱۱ــ۶ قرار گیرد. تصویر گل لاله[۱] را در آن قرار دهید و اگر لازم است موقعیت برچسب و دکمه را کمی‌تغییر دهید تا تصویر بهتر در روی فرم جای گیرد. با اجرای برنامه ، یک فرم خالی نمایش داده شود(تصویر و پیام هر دو دیده نشوند) اما هنگامی که کاربر بر روی دکمه «بگو سلام» کلیک کرد هر دو نشان داده شوند.

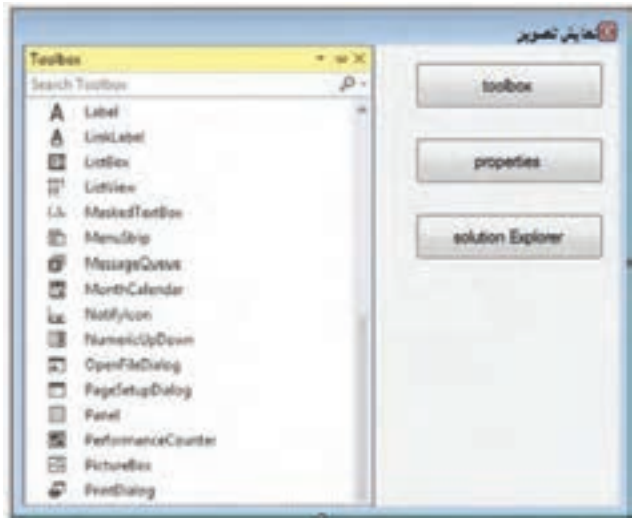
۱ــ Tulip

شکل ۱۲ــ۶ ــ اضافه کردن یک جعبه تصویری

کار در کارگاه ۳: کار با جعبه تصویر

مثال ۳ــ۶ : ایجاد فرمی با سه دکمه و یک جعبه تصویر برای نمایش سه تصویر

در این مثال می‌خواهیم فرمی مانند شکل ۱۳ــ۶ با سه دکمه ایجاد کنیم و سه تصویر مختلف را در PictureBox نمایش دهیم به صورتی که با کلیک روی هر دکمه، تصویر متناظر با آن دکمه در PictureBox قرار گیرد. ما از تصویر پنجره‌های Properties، Toolbox و Solution محیط IDE ویژوال استودیو استفاده می‌کنیم.



شکل ۱۳ــ۶ــ فرمی با سه دکمه و PictureBox برای نمایش تصویر

۱ــ ایجاد پروژه: وارد برنامه VS شوید و پروژه جدیدی از نوع Windows Froms Application با نام Picture Show در مسیر مشخص بسازید.

۲ــ تعیین ویژگی‌های فرم

جدول ۷ــ۶ ــ ویژگی‌های فرم

Form		
ویژگی		مقدار
Name		Form1
Text		نمایش تصویر
FormBorderStyle		FixedToolWindow
RightToLeft		Yes
Size	Width	440
	Height	362

۳ــ تعیین ویژگی دکمه‌ها و جعبه تصویر

جدول ۸ــ۶ ــ ویژگی‌های دکمه اول

Button1		
ویژگی		مقدار
Name		propertiesBtn
Text		properties
Size	Width	134
	Height	33

جدول ۹ــ۶ ــ ویژگی‌های دکمه دوم

Button2		
ویژگی		مقدار
Name		toolboxBtn
Text		toolbox
Size	Width	134
	Height	33

جدول ۱۰ــ۶ــ ویژگی‌های دکمه سوم

Button3		
ویژگی		مقدار
Name		solutionBtn
Text		solution Explorer
Size	Width	134
	Height	33

جدول ۱۱ــ۶ ــ ویژگی‌های جعبه تصویر

PictureBox		
ویژگی		مقدار
Name		pic
Dock		Left
SizeMode		StretchImage
Size	Width	250
	Height	328

ویژگی Image کنترل PictureBox را در پنجره Properties پیدا کرده روی علامت ... کلیک کنید تا جعبه محاوره‌ای انتخاب منبع (SelectResource) باز شود با استفاده از دکمهٔ Import سه تصویر را به لیست اضافه کنید و در پایان یکی از تصاویر را از لیست انتخاب کرده و روی دکمه Ok کلیک کنید.

تصاویر اضافه شده در پنجره Solution Explorer در پوشهٔ Resources قرار می‌گیرد.

۴ــ ایجاد واکنش نسبت به رویداد کلیک ماوس: با کلیک هر یک از دکمه‌ها تصویر PictureBox باید تغییر کند بنابراین باید ویژگی Image آن را تغییر دهید. همانطور که قبلاً مشاهده کردید در هنگام انتخاب عکس برای کنترل pictureBox در جعبه محاوره‌ای انتخاب منبع، گزینه Project resource file را انتخاب کردید. به این ترتیب با وارد کردن یک یا چند تصویر، یکی را به عنوان تصویر مورد نظر خود انتخاب نمودید. در واقع با این کار،منابعی از جنس تصویر برای نرم افزار ایجاد کردید و از آن استفاده نمودید. این منابع در داخل نرم افزار شما جایگذاری می‌شوند.

اگر بخواهید از منابع معرفی شده در نرم‌افزار خود در زمان اجرا (در کد) استفاده کنید باید از کلاس Resources که در فضای نام Properties است، به صورت زیر استفاده نمایید.

```
Properties.Resources.نام منبع
```

در اینجا منابع ما از جنس تصویر (image) هستند و برای این مثال، چون نام تصویر جعبه ابزار toolbox بوده است به همین نام در منابع موجود است و از آن می‌توانیم استفاده نماییم :

```
Properties.Resources.toolbox
```

درفضای طراحی فرم روی دکمه‌ها دابل کلیک کنید تا الگوی رویداد کلیک آنها نوشته شود، سپس کد را مانند زیر تکمیل نمایید.

```
private void propertiesBtn_Click(object sender, EventArgs e)
{
   pic.Image = Properties.Resources.properties;
}

private void solutionBtn_Click(object sender, EventArgs e)
{
   pic.Image = Properties.Resources.solution;
}

private void toolboxBtn_Click(object sender, EventArgs e)
{
   pic.Image = Properties.Resources.toolbox;
}
```

برنامه ۲ــ۶

توسعه و بهبود برنامه : در برنامه فوق یک کنترل برچسب اضافه کنید که نام تصویری را نشان دهد که در PictureBox قرار دارد؟

کار در کارگاه ۴: بزرگ کردن تصاویر

مثال۴ ــ۶ : در این مثال می‌خواهیم فرمی‌مانند شکل ۱۴ــ۶ با سه دکمه ایجاد کنیم و سه تصویر مختلف را روی دکمه‌ها قرار دهیم و با کلیک روی هر دکمه، تصویر آن دکمه در PictureBox قرار گیرد. در این مثال به‌جای متن، روی دکمه‌ها تصویر قرار می‌دهیم.

شکل ۱۴ــ۶ــ فرمی با سه دکمه و جعبه تصویر برای نمایش تصویر

۱ــ ایجاد پروژه : وارد برنامه VS شوید و پروژه جدیدی از نوع WindowsFromsApplication با نام PictureZoom در مسیر مشخص بسازید.

۲ــ تعیین ویژگی‌های فرم

جدول ۱۲ــ۶ ــ ویژگی‌های فرم

Form		
ویژگی		مقدار
Name		Form1
Text		بزرگ کردن تصویر
RightToLeft		Yes
Size	Width	440
	Height	362

۳ــ تعیین ویژگی‌های دکمه‌ها و جعبه تصویر

جدول ۱۳ــ۶ــ ویژگی‌های دکمه اول

Button1		
ویژگی		مقدار
Name		pic1Btn
Text		
BackgroundImage		تصویر دلخواه۱
BackgroundImageLayout		Stretch
Size	Width	126
	Height	90

جدول ۱۴ــ۶ــ ویژگی‌های دکمه دوم

Button2		
ویژگی		مقدار
Name		pic2Btn
Text		
BackgroundImage		تصویر دلخواه۲
BackgroundImageLayout		Stretch
Size	Width	126
	Height	90

جدول ۱۵ــ۶ــ ویژگی‌های دکمه سوم

Button3		
ویژگی		مقدار
Name		pic3Btn
Text		
BackgroundImage		تصویر دلخواه۳
BackgroundImageLayout		Stretch
Size	Width	126
	Height	90

جدول ۱۶ــ۶ــ ویژگی‌های جعبه تصویر

PictureBox		
ویژگی		مقدار
Name		piclarge
SizeMode		StretchImage
Size	Width	250
	Height	328

برای قرار دادن تصویر روی دکمه‌ها،در فضای طراحی روی دکمه کلیک کنید تا در پنجره Properties ویژگی‌های آن نمایش داده شود. ویژگیBackgroundImage دکمه راپیدا کرده و روی علامت ... کلیک کنید تا جعبه محاوره‌ای انتخاب منبع باز شود وسپس با استفاده از دکمهٔ Import

سه تصویر را به لیست اضافه کنید. تصاویر اضافه شده در پنجره Solution Explorer در پوشهٔ Resources قرار می‌گیرد. در پایان یکی از تصاویر اضافه شده به لیست را انتخاب کرده و روی دکمه Ok کلیک کنید.

برای اینکه تصویر روی دکمه‌ها و PictureBox کامل دیده شود باید ویژگی BackgroundImageLayout دکمه‌ها و SizeMode کنترل PictureBox را مساوی Stretch قرار دهید.

مقدار ویژگیText دکمه‌ها را پاک کنید تا روی تصاویر متنی نباشد و فقط تصویر زمینه دیده شود.

۴ــ ایجاد واکنش نسبت به رویداد کلیک ماوس : با کلیک هر یک از دکمه‌ها، تصویر دکمه باید درPictureBox نمایش داده شود. بنابراین باید ویژگی Image آن را مساوی BackgroundImage دکمه قرار دهیم.

درفضای طراحی فرم روی دکمه‌ها دابل کلیک کنید تا الگوی رویداد کلیک آنها نوشته شود، سپس کد را مانند زیر تکمیل نمایید.

```
private void pic1Btn_Click(object sender, EventArgs e)
{
   piclarge.Image = pic1Btn.BackgroundImage;
}

private void pic2Btn_Click(object sender, EventArgs e)
{
   piclarge.Image = pic2Btn.BackgroundImage;
}

private void pic3Btn_Click(object sender, EventArgs e)
{
   piclarge.Image = pic3Btn.BackgroundImage;
}
```

برنامه ۳ــ۶

مثال ۵ــ۶ : حرکت دادن برچسب با کمک دکمه‌ها

در این مثال می‌خواهیم فرمی‌مانند شکل ۱۵ــ۶ طراحی کنیم که کاربر بتواند یک برچسب را در چهارجهت اصلی بالا، پایین، چپ و راست حرکت دهد و از جهت آخرین حرکت برچسب با اطلاع باشد.

الگوریتم و روش انجام کار : برای تعیین جهت حرکت از دکمه استفاده می‌کنیم. با کلیک روی هر دکمه فرمان، برچسب در جهت متناظر با آن دکمه فرمان ، ۱۰ واحد جابه‌جا می‌شود و جهت حرکت نیز روی برچسب نمایش داده می‌شود. بنابراین به چهار دکمه فرمان و یک کنترل برچسب نیاز داریم.

شکل ۱۵ــ۶ ــ فرم جابه‌جایی برچسب

۱ــ ایجاد پروژه : وارد برنامه VS شوید و پروژه جدیدی از نوع WindowsFromsApplication به نام MoveLabelProj ایجاد نمایید.

۲ــ درج کنترل‌ها : روی فرم چهار کنترل دکمه و یک برچسب قرار دهید.

۳ــ تنظیم ویژگی‌های برچسب و فرم : ویژگی‌های فرم و برچسب را طبق جداول زیر تنظیم کنید.

جدول ۱۷ــ۶ ــ ویژگی‌های برچسب

Label		
ویژگی		مقدار
Name		flowLbl
Text		
Backcolor		رنگ دلخواه
Location	X	180
	Y	9

جدول ۱۸ــ۶ ــ ویژگی‌های فرم

Form		
ویژگی		مقدار
Name		Form1
Text		جابه‌جایی برچسب
Size	Width	452
	Height	362

۴ــ تنظیم ویژگی‌های دکمه‌ها : چهار کنترل دکمه را به فرم اضافه کرده و ویژگی‌های آنها را طبق جداول زیر تنظیم کنید.

جدول ۱۹ــ۶ــ ویژگی‌های دکمه left

Button	
ویژگی	مقدار
Name	right
Width	43
Height	28
Font	Wingdings
Text	←

جدول ۲۰ــ۶ــ ویژگی‌های دکمه right

Button	
ویژگی	مقدار
Name	left
Width	43
Height	28
Font	Wingdings
Text	→

جدول ۲۱ــ۶ــ ویژگی‌های دکمه down

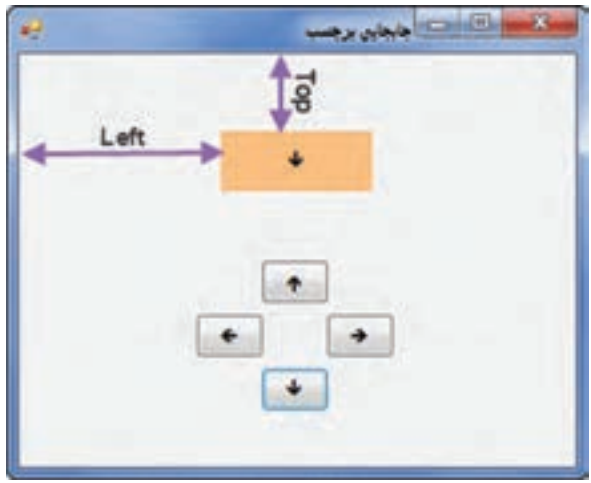
Button	
ویژگی	مقدار
Name	down
Width	43
Height	28
Font	Wingdings
Text	↓

جدول ۲۲ــ۶ــ ویژگی‌های دکمه up

Button	
ویژگی	مقدار
Name	up
Width	43
Height	28
Font	Wingdings
Text	↑

برای قرار دادن علامت جهت روی دکمه‌ها می‌توان از تصویر استفاده کرد و ویژگی BackgroundImage دکمه را تغییر داد و یا از نمادها و ویژگی Text دکمه استفاده کرد. در اینجا ما از روش دوم استفاده می‌کنیم. ویژگی Font دکمه‌ها را مساوی Wingdings قرار می‌دهیم. در این فونت نمادها، دارای کد اسکی ۲۳۱، ۲۳۲، ۲۳۳ و ۲۳۴ هستند. برای قرار دادن این نمادها در ویژگی Text هر دکمه، کلید Alt را نگه داشته و کد اسکی کلید را با کلیدهای ماشین حساب صفحه کلید وارد کنید.

۵ــ نوشتن رویداد click برای دکمه‌ها : در زمان طراحی فرم موقعیت هر کنترل نسبت به فرمی که روی آن قرار دارد با ویژگی Location که دارای دو مشخصه X و Y است تعیین می‌شود ولی برای تغییر موقعیت کنترل در زمان اجرا از دو ویژگی Left و Top که به ترتیب معادل X و Y هستند استفاده می‌کنیم. بنابراین مشخصه Left ، فاصله کنترل از سمت چپ فرم و مشخصه Top ، فاصلهٔ کنترل از بالای فرم است(شکل ۱۶ــ۶). بنابراین هر چه به مقدار مشخصه Left اضافه شود، فاصلهٔ کنترل از سمت چپ فرم بیشتر می‌شود و کنترل به سمت راست جابه‌جا می‌شود. هر چه از مقدار مشخصه Left کم شود، فاصلهٔ کنترل از سمت چپ فرم کمتر می‌شود و کنترل به سمت چپ جابه‌جا می‌شود. به همین ترتیب هر چه به مقدار مشخصه Top اضافه شود، فاصله کنترل از بالای فرم بیشتر می‌شود و کنترل به سمت پایین جابه‌جا می‌شود. هر چه از مقدار مشخصه Top کم شود، فاصله بالای کنترل از بالای فرم بیشتر می‌شود و کنترل به سمت بالا جابه‌جا می‌شود.



شکل ۱۶ــ۶ــ ویژگیTop و Left کنترل

روی هر یک از دکمه‌ها دابل کلیک کرده، تا الگوی رویداد کلیک آن نوشته شود، سپس رویداد click دکمه‌ها را به صورت زیر بنویسید :

```
private void right_Click(object sender, EventArgs e)
{
    flowLbl.Left += 10;
    flowLbl.Text = right.Text;
}

private void up_Click(object sender, EventArgs e)
{
  flowLbl.Top -= 10;
  flowLbl.Text = up.Text;
 }

private void down_Click(object sender, EventArgs e)
{
    flowLbl.Top += 10;
    flowLbl.Text = down.Text;
 }

private void left_Click(object sender, EventArgs e)
{
    flowLbl.Left -= 10;
    flowLbl.Text = left.Text;
  }
```

برنامه ۴ــ۶

در کد نوشته شده، برچسب به هر جهتی حرکت کند آن جهت روی برچسب نشان داده می‌شود. به این منظور با کلیک هر دکمه مقدار ویژگی Text آن دکمه را در مشخصه Text برچسب قرار داده‌ایم.

توسعه و بهبود برنامه : با اجرای برنامه متوجه خواهید شد که با کد نوشته شده برچسب می‌تواند

از محدوده فرم خارج می‌شود. حداکثر مقداری که می‌توان در دو ویژگی Left و Top برچسب قرار داد تا از فرم خارج نشود چقدر است؟ کد برنامه را تغییر دهید تا با تغییر مقدار این دو مشخصه، برچسب از محدوده فرم خارج نشود.

کار در کارگاه ۶: حدس تصویر

مثال ۶ــ۶ : کنترل جعبه متن[۱]

در این مثال با ایجاد فرمی‌مانند شکل ۱۷ــ۶ یک بازی می‌سازیم که در آن کاربر یک تصویر پنهان را حدس بزند! بازیکن برای اینکه بتواند تصویر مورد نظر را حدس بزند باید روی بخش‌هایی از تصویر کلیک کند تا آن بخش نمایش داده شود. هر زمان که کاربر تصویر را تشخیص داد، باید تشخیص خود را در جعبه متن بنویسد. در صورت درست بودن تشخیص کاربر، امتیاز او نمایش داده می‌شود.

شکل ۱۷ــ۶ــ فرم بازی حدس تصویر

الگوریتم یا روش انجام کار : برای اینکه عملیات نمایش قسمتی از تصویر را بتوانیم شبیه‌سازی کنیم در این برنامه از یک روش ساده اما کارا استفاده می‌کنیم. سطح تصویر را با چندین Label می‌پوشانیم. با کلیک هر برچسب ، آن برچسب پنهان خواهد شد و قسمتی از تصویر که زیر آن قرار دارد نمایش داده می‌شود.

روش امتیاز دهی را نیز به این صورت پیاده سازی می‌کنیم که ۹ برچسب برای پنهان کردن

---

۱ ــ Text Box

تصویر قرار می‌دهیم و به صورت زیر برای هر یک امتیازی مشخص می‌کنیم.

۵ Label1	۱۰ Label2	۵ Label3
۱۰ Label4	۱۰ Label5	۱۰ Label6
۵ Label7	۱۰ Label8	۵ Label9

شکل ۱۸ــ۶

دلیل آنکه به گوشه‌ها امتیاز کمتری داده‌ایم این است که معمولاً در گوشه‌های تصویر جزئیات کمتری برای شناخت آن وجود دارد. با پنهان شدن هر برچسب امتیاز آن از مجموع امتیاز (۷۰) کم می‌شود. با وارد کردن حدس داخل جعبه متن و زدن دکمه ، حدس مورد نظر با جواب مقایسه می‌شود و پیام مناسب نمایش داده می‌شود.

۱ــ ایجاد پروژه : وارد برنامه VS شوید و پروژه جدیدی از نوع WindowsFromsApplication به نام GussePictureProj ایجاد نمایید.

۲ــ درج کنترل‌ها : در قسمت پایین فرم یک کنترل جعبه متن (TextBox)، یک دکمه و یک برچسب برای حدس و نشان دادن امتیاز کاربر قرار دهید. در بخش میانی یک جعبه تصویر (PictureBox) و ۹ برچسب برای پنهان کردن تصویر قرار دهید.

۳ــ تنظیم ویژگی‌های برچسب‌ها ، جعبه متنی و دکمه : ویژگی‌های جعبه متن، دکمه و برچسب‌ها را طبق جداول زیر تنظیم کنید.

جدول ۲۳ــ۶ ــ ویژگی‌های جعبه متن

TextBox	
ویژگی	مقدار
Name	guess
Text	

جدول ۲۴ــ۶ ــ ویژگی‌های دکمه

Button	
ویژگی	مقدار
Name	check
Text	درست است؟

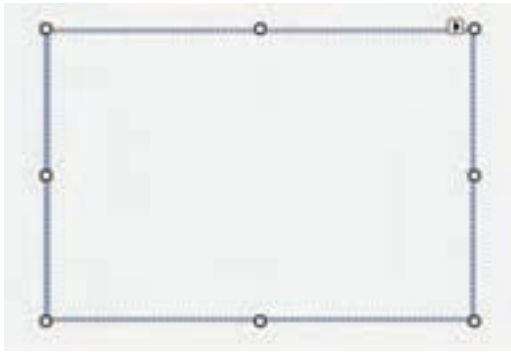
جدول ۲۵ــ۶ ــ ویژگی‌های برچسب امتیاز

Label (نمایش امتیاز)	
ویژگی	مقدار
Name	result
Text	حذف مقدار

جدول ۲۶ــ۶ ــ ویژگی‌های برچسب‌های پنهان کننده تصویر

Label10,……, Label3, Label2(برچسب‌هایی برای پنهان کردن تصویر)		
ویژگی	مقدار	توضیح
AutoSize	false	غیر فعال کردن اندازه خودکار
BackColor	#93aae1	تغییر رنگ پس زمینه
Text		متن داخل برچسب را خالی قرار می‌دهد.
Width	91	عرض کنترل را بر حسب پیکسل مشخص می‌کند.
Height	60	ارتفاع کنترل را بر حسب پیکسل مشخص می‌کند.

بعد از قراردادن یکی از برچسب‌هایی که برای پنهان کردن تصویر لازم است روی فرم در محل مناسب ، و مقداردهی ویژگی‌های آن براساس جدول ۲۶ــ۶، با نسخه برداری (copy) از آن، برچسب‌ها را به ۹ عدد می‌رسانیم. سپس این برچسب‌ها را به صورت دقیقی کنار هم قرار می‌دهیم تا در کنار هم، شکل مستطیلی را ایجاد کنند.



شکل ۱۸ــ۶ــ هم اندازه کردن کنترل کادر تصویر با مستطیل شامل برچسب‌ها

۴ــ تنظیم ویژگی‌های جعبه تصویر

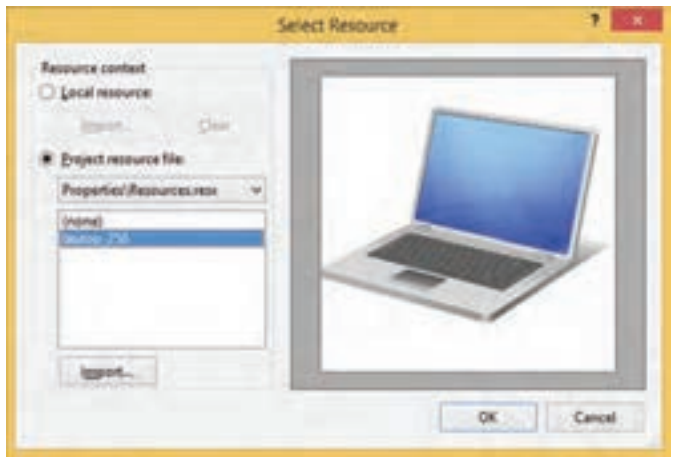
تصویر : یک کنترل pictureBox به فرم اضافه می‌کنیم. این عمل را توسط کشیدن ماوس روی فرم، طوری انجام می‌دهیم که اندازه آن برابر با اندازه مستطیل ساخته شده توسط برچسب‌ها باشد.

ویژگی‌های جعبه تصویر را مطابق جدول زیر تنظیم می‌نماییم :

جدول ۲۷ــ۶ ــ ویژگی‌های جعبه تصویر

PictureBox	
ویژگی	**مقدار**
Name	Pic
SizeMode	StretchImage

همچنین توسط ویژگیImage، تصویر آن را مشخص می‌کنیم.



شکل ۱۹ــ۶ــ تنظیم ویژگی Image کنترل کادر تصویر

همان‌طور که ملاحظه می‌نمایید بعد از انتخاب عکس ،به دلیل آنکه کنترل PictureBox بعد از Labelها ایجاد شده است روی آنها قرار می‌گیرد.

شکل ۲۰ــ۶ــ قرار گرفتن کادر تصویر روی برچسب‌ها

برای آنکه کنترل PictureBox پشت برچسب‌ها قرار گیرد، روی آن راست کلیک کرده، گزینه Send To Back را انتخاب کنید.

شکل ۲۱ــ۶ــ انتقال کنترل به پشت کنترل‌های دیگر

به این ترتیب تصویر توسط برچسب‌ها پنهان خواهد شد.

۵ــ تعریف متغیر برای نگهداری امتیاز : در کلاس Form روی فرم دابل کلیک کنید تا پنجره کد باز شود.

در کلاس فرم متغیر score را تعریف می‌کنیم تا بتوانیم در تمام رویدادهای فرم از آن استفاده کنیم.

```
public partial class Form1 : Form
{
 byte score = 70;
....
```

این متغیر را از نوع byte در نظر گرفته ایم و آن را برابر با ۷۰ قرار داده ایم. زیرا مجموع امتیازها ۷۰ است.

سؤال: چرا برای نگهداری امتیاز کاربر متغیری از نوع Byte تعریف کردیم؟

۶ــ نوشتن رویداد click برای برچسب‌های پنهان کننده تصویر : با کلیک روی برچسب، آن برچسب پنهان می‌شود. بنابراین در رویداد کلیک هر برچسب باید آن برچسب را پنهان کرد. همان‌طور که در کارگاه ۲ دیدید، پیدا و پنهان بودن یک کنترل به یک ویژگی کنترل بستگی دارد و آن ویژگی، Visible می‌باشد. اگر مقدار این ویژگیtrue باشد آن کنترل نمایش داده می‌شود و اگر مقدار آن false باشد، پنهان می‌شود.

برای ایجاد رویداد کلیک برچسب، همان‌طور که قبلاً گفته شد، می‌توانید روی آن دوبار کلیک کنید و یا از لیست رویدادها استفاده کنید. بعد از ایجاد کد رویداد برای برچسب label1 خواهیم داشت :

```
private void label1_Click(object sender, EventArgs e)
{
}
```

با کلیک روی این برچسب باید این کنترل پنهان شود و ۵ امتیاز از امتیاز کل کم شود؛ زیرا همان‌طور که در طرح امتیاز گفته شد، برچسب‌های گوشه ۵ امتیاز خواهند داشت.

```
private void label1_Click(object sender, EventArgs e)
{
score -= 5;
label1.Visible = false;
}
```

برای سایر برچسب‌هایی که تصویر را می‌پوشانند نیز همین عملیات را انجام می‌دهیم.اما توجه داشته باشید که در رویداد کلیک هر برچسب باید همان برچسب پنهان شود.همچنین برای برچسب‌هایی که در گوشه قرار نمی‌گیرند باید ۱۰ امتیاز کسر شود. برای برچسب‌های ۳، ۷ و ۹

مانند برچسب شماره ۱، ۵ امتیاز و برای سایر برچسب‌ها ۱۰ امتیاز کم می‌شود. به عنوان مثال برای label2 داریم :

```
private void label2_Click(object sender, EventArgs e)
{
    score -= 10;
    label2.Visible = false;
}
```

کدهای مربوط به برچسب‌ها در برنامه ۵ــ۶ آورده شده است.

```
private void label1_Click(object sender, EventArgs e)
{
    score -= 5;
    label1.Visible = false;
}
private void label2_Click(object sender, EventArgs e)
{
    score -= 10;
    label2.Visible = false;
}
private void label3_Click(object sender, EventArgs e)
{
    score -= 5;
    label3.Visible = false;
}
private void label4_Click(object sender, EventArgs e)
{
    score -= 10;
    label4.Visible = false;
}
private void label5_Click(object sender, EventArgs e)
{
    score -= 10;
```

```
        label5.Visible = false;
    }
    private void label6_Click(object sender, EventArgs e)
    {
        score -= 10;
        label6.Visible = false;
    }
    private void label7_Click(object sender, EventArgs e)
    {
        score -= 5;
        label7.Visible = false;
    }
    private void label8_Click(object sender, EventArgs e)
    {
        score -= 10;
        label8.Visible = false;
    }
    private void label9_Click(object sender, EventArgs e)
    {
        score -= 5;
        label9.Visible = false;
    }
```

برنامه ۵ــ۶ــ کد مربوط به برچسب‌ها

۷ــ نوشتن رویداد click برای دکمه : کاربر پس از وارد کردن حدس خود در جعبه متن باید روی دکمه کلیک کند. برای بررسی اینکه حدس کاربر درست است یا خیر، رویداد کلیک دکمه را می‌نویسیم . توسط شرط (if) بررسی می کنیم که آیا متن درون جعبه متنی (guess) برابر متن «لپ‌تاپ» می‌باشد یا خیر. اگر شرط درست بود پیام حدسِ درست به همراه امتیاز در برچسب result قرار می‌گیرد و در غیر این صورت پیام حدسِ اشتباه در برچسب result جای می‌گیرد. لازم به یادآوری است که متد ToString مقدار متغیر score را به متن تبدیل می‌کند.

```
private void check_Click(object sender, EventArgs e)
{
    if (guess.Text == «لپ تاپ»)
result.Text = «آفرین درست حدس زدی. امتیاز: » +score.ToString();
else
    result.Text = «حدست درست نبود!»;
}
```

برنامه ۶ــ۶

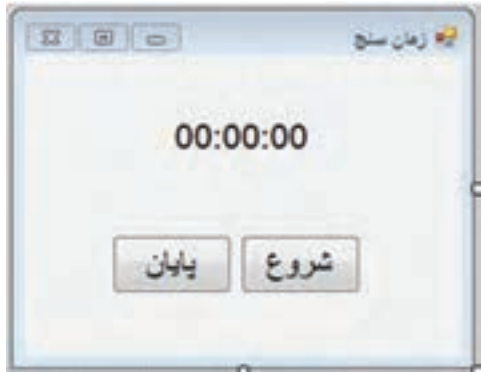
توسعه و بهبود برنامه : به کدهایی که نوشته اید فکر کنید و ببینید آیا می‌توانید ایده‌های دیگری برای این بازی بدهید؟ به عنوان نمونه بیاندیشید چگونه می‌توان برای تعداد پاسخ‌های اشتباه محدودیت گذاشت. به عنوان مثال با دادن ۲ پاسخ اشتباه، پیام اتمام بازی داده شود.

کار در کارگاه ۷: طراحی زمان‌سنج دیجیتال

مثال ۷ــ۶ : کنترل زمان‌سنج[۱]

در این مثال می‌خواهیم با طراحی فرم شکل ۲۲ــ۶ برنامه‌ای بنویسیم که کار زمان سنج را انجام دهد.

الگوریتم یا روش انجام کار : این برنامه با کلیک دکمهٔ شروع، زمان‌سنج را راه اندازی کرده و با کلیک دکمهٔ پایان، زمان‌سنج را متوقف می‌کند. بنابراین به دو دکمه فرمان، یک کنترل برچسب و یک کنترل زمان‌سنج نیاز داریم.



شکل ۲۲ــ۶ــ فرم زمان سنج دیجیتالی

۱ ــ Timer

۱ــ ایجاد پروژه : وارد برنامه VS شوید و پروژه جدیدی از نوع WindowsFromsApplication به نام TimerProj ایجاد نمایید.

۲ــ درج کنترل‌ها : روی فرم دو کنترل دکمه و یک برچسب قرار دهید.

۳ــ تنظیم ویژگی‌های دکمه‌ها، برچسب و فرم : ویژگی‌های فرم و کنترل‌ها را طبق جداول زیر تنظیم کنید.

جدول ۲۸ــ۶ــ ویژگی‌های دکمه شروع

Button	
ویژگی	مقدار
Name	startBut
Text	شروع

جدول ۲۹ــ۶ــ ویژگی‌های دکمه پایان

Button	
ویژگی	مقدار
Name	stopBut
Text	پایان

جدول ۳۰ــ۶ــ ویژگی‌های برچسب

Label	
ویژگی	مقدار
Name	timer Lbl
Text	00: 00: 00

جدول ۳۱ــ۶ــ ویژگی‌های فرم

Form		
ویژگی		مقدار
Name		Form1
Text		زمان سنج
RightToLeft		yes
RightToLeftLayout		True
Size	Width	292
	Height	228

۴ــ تنظیم ویژگی‌های زمان‌سنج : کنترل Timer را به فرم اضافه کرده و ویژگی‌های آن را طبق جدول ۳۲ــ۶ تنظیم کنید.

جدول ۳۲ــ۶ــ ویژگی‌های زمان‌سنج

Timer	
ویژگی	مقدار
Name	Timer1
interval	1000
Enabled	false

کنترل زمان‌سنج برای اطلاع از سپری شدن فاصله‌های زمانی یکسان به کار می‌رود. برای مثال اگر با رایانه کار می‌کنید، برای حفظ سلامتی لازم است هر نیم ساعت چند نرمش مناسب انجام دهید. ساعت را تنظیم می‌کنید که هر نیم ساعت زنگ بزند. زنگ ساعت به شما اطلاع می‌دهد که زمان نرمش رسیده است. کنترل زمان‌سنج این کار را برای برنامه انجام می‌دهد. فاصله زمانی در ویژگی Interval تایمر مشخص می‌شود. کنترل زمان‌سنج دارای رویداد Tick می‌باشد که مشابه زنگ ساعت است. این رویداد در فواصل زمانی یکسان که در ویژگی Interval کنترل مشخص می‌شود رخ می‌دهد. واحد زمانی ویژگی Interval میلی ثانیه است بنابراین اگر بخواهیم هرثانیه یکبار این رویداد رخ دهد، باید ویژگی Interval را مساوی ۱۰۰۰ قرار دهیم.

هرگاه کار شما با رایانه تمام شد ، دیگر نیازی نیست که ساعت زنگ بزند؛ بنابراین زنگ ساعت را قطع می‌کنید. چگونه زنگ زمان‌سنج را قطع کنیم (رویداد Tick را متوقف کنیم)؟

کنترل‌ها دارای ویژگی Enabled هستند که از نوع Boolean است و هرگاه مقدار آن false شود، کنترل غیرفعال شده و به هیچ رویدادی واکنش نشان نمی‌دهد.

۵ــ تعریف متغیرهایی برای نگهداری ساعت، ثانیه و دقیقه : زمان سپری شده را به صورت ساعت، دقیقه و ثانیه باید نگهداری کنیم؛ بنابراین نیاز به متغیر داریم. متغیرها باید در تمام فرم قابل دسترسی باشند؛ بنابراین در کلاس Form آنها را تعریف کرده، با صفر مقداردهی می‌کنیم.

روی فرم دابل کلیک کنید تا پنجره کد باز شود.

در کلاس فرم برای ثانیه، دقیقه و ساعت متغیرهایی با نام second، minute و hour و از نوع byte تعریف می‌کنیم.

```
Namespace TimerProject
{
  public partial class Form1 : Form
  {
    byte hour, second,minutes;
    public Form1()
    {
      InitializeComponent();
    }
                }
}
```

برنامه ۷ــ۶

۶ــ نوشتن رویداد click برای دکمه‌ها : در رویداد Click دکمه شروع، کنترل زمان‌سنج را فعال می‌کنیم. برای این منظور ویژگی Enabled زمان‌سنج را با true مقداردهی می‌کنیم و متغیرهایی که مقدار ثانیه، دقیقه و ساعت را نگهداری می‌کنند، صفر می‌کنیم.

```
private void startBut_Click(object sender, EventArgs e)
{
  timer1.Enabled = true;
  hour = minute =second= 0;
}
```

برنامه ۸ــ۶

در رویداد Click دکمه پایان، کنترل زمان‌سنج را غیرفعال می‌کنیم.

```
private void stopBut_Click(object sender, EventArgs e)
{
  timer1.Enabled = false;
}
```

برنامه ۹ــ۶

۷ــ نوشتن رویداد Tick برای کنترل زمان‌سنج : از آنجا که فاصله زمانی رخ داده رویداد Tick را روی ۱ ثانیه (Interval=۱۰۰۰) تنظیم کردیم، باید در رویداد Tick به متغیر ثانیه یکی اضافه کنیم و اگر مقدار آن به ۶۰ رسید آن را صفر کرده و به متغیر دقیقه یکی اضافه کنیم و اگر مقدار متغیر دقیقه به ۶۰ رسید مقدار متغیر دقیقه را صفر کرده و به متغیر ساعت یکی اضافه می‌کنیم. در انتها ویژگی text کنترل برچسب timerLbl را تغییر می‌دهیم که زمان را نشان دهد.

```
private void timer1_Tick(object sender, EventArgs e)
{
  second+=1;
  if (second==60)
  {
    second=0;
    minute +=1;
  }
```

```
        if (minute == 60)
        {
          minute = 0;
          hour += 1;
        }
timerLbl.Text=string.Format("{0:00}:{1:00}:{2:00}",
hour,minutes,second);
    }
```

برنامه ۱۰ــ۶

متد Format نوع داده string، رشته را به الگوی خاص تبدیل می‌کند. در اینجا می‌خواهیم ساعت به صورت 00:00:00 نمایش داده شود. بنابراین جای مقادیر ساعت، دقیقه و ثانیه را با پارامترهای {0}،{1} و {2} در رشته مشخص کرده و قرار دادن 00 : در پارامتر ساعت ،دقیقه و ثانیه سبب می‌شود که به صورت دو رقمی نمایش داده شوند.

توسعه و بهبود برنامه : در برنامه دکمه سومی برای توقف و ادامه (pause و resume) قرار دهید که در صورت کلیک آن زمان‌سنج به صورت موقت متوقف شود و با کلیک مجدد آن به کار خود ادامه دهد.

کار در کارگاه ۸: تغییر عکس روی فرم در زمان اجرا

مثال ۸ــ۶ : کنترل کادر محاوره‌ای انتخاب فایل[۱]

در این مثال می‌خواهیم با ایجاد فرم شکل ۲۳ــ۶ برنامه‌ای بنویسیم که کاربر بتواند یک تصویر را به‌وسیلهٔ کادر محاوره‌ای، انتخاب کرده و نمایش دهد.

شکل ۲۳ــ۶ــ فرم انتخاب تصویر از رایانه

۱ ــ Open File Dialog

الگوریتم یا روش انجام کار : می‌خواهیم برنامه‌ای بنویسیم که تصویر انتخاب شده توسط کاربر روی فرم نمایش داده شود. این برنامه با کلیک جعبه تصویر، یک جعبه انتخاب فایل را نمایش می‌دهد و با انتخاب فایل تصویر از کادر انتخاب فایل، تصویر آن در جعبه تصویر نمایش داده می‌شود.

۱ــ ایجاد پروژه : وارد برنامه VS شوید و پروژه جدیدی از نوع WindowsFromsApplication به نام ShowPictureProj ایجاد نمایید.

۲ــ درج کنترل‌ها : روی فرم یک کنترل PictureBox و یک OpenFileDialog قرار دهید.

۳ــ تنظیم ویژگی‌های جعبه تصویر : ویژگی‌های فرم و جعبه تصویر را طبق جدول زیر تنظیم کنید.

جدول ۳۳ــ۶ــ ویژگی‌های جعبه تصویر

PictureBox	
ویژگی	**مقدار**
Name	pictureBox1
BackColor	# ffc080
SizeMode	StretchImage
Width	430
Height	253

۴ــ تنظیم ویژگی‌های کادر محاوره‌ای انتخاب فایل : برای باز کردن فایل جدید در برنامه‌هایی مانند word با انتخاب گزینهٔ open از منوی file کادر محاوره‌ای open باز می‌شود و به‌وسیلهٔ آن می‌توان فایل موردنظر را انتخاب کرد. کنترل OpenFileDialog کادر محاوره‌ای open است. این کنترل پایین فرم قرار می‌گیرد و در زمان اجرا دیده نمی‌شود.

در این مثال برای انتخاب فایل تصویر به این کنترل نیاز داریم. آن را به فرم اضافه کرده و ویژگی‌های آن را طبق جدول ۳۴ــ۶ تنظیم کنید.

جدول ۳۴ــ۶ــ ویژگی‌های کادر محاوره‌ای OpenFile

OpenFileDialog		
ویژگی	مقدار	توضیحات
Name	OpenFileDialog1	تعیین نام
Filter	Jpg File|*.jpg|jpeg files|*.jpeg	تعیین فرمت فایل‌های قابل انتخاب
FileName		خالی کردن کادر نام فایل
Width	430	تعیین عرض
Height	253	تعیین کنترل

توجه کنید که می‌توانید توسط ویژگی Filter نوع فایل‌هایی که کادر محاوره‌ای نشان می‌دهد را مشخص کنید. مقداردهی این ویژگی به صورت زیر است :

.....|نوع ۲ | شرح نوع ۲ | نوع ۱ | شرح نوع ۱

مثال : Jpg Files|*.jpg|jpeg Files|*.jpeg

شکل ۲۴ــ۶

دراین نمونه ما می‌خواهیم که یک تصویر را انتخاب کنیم، بنابراین نوع فایل‌ها را jpg و jpeg مشخص کرده‌ایم. در کادر محاوره‌ای OpenFile لیست نوع فایل‌ها نشان داده شده به صورت شکل ۲۵ــ۶ است.

شکل ۲۵ــ۶ ــ انتخاب نوع فایل‌های تعیین شده در ویژگی Filter

۵ــ نوشتن رویداد click برای جعبه تصویر : در رویداد کلیک جعبه تصویر، کادر محاوره‌ای انتخاب فایل را باز می‌کنیم.

```
private void pictureBox1_Click(object sender, EventArgs e)
    {
        openFileDialog1.ShowDialog();
    }
```

برنامه ۱۱ــ۶

کادر محاوره‌ای OpenFileDialog دارای متد ShowDialog برای باز کردن کادر محاوره‌ای است.

۶ــ نوشتن رویداد FileOk برای کنترل کادر محاوره‌ای انتخاب فایل : می‌خواهیم در صورت انتخاب فایل از کادر محاوره‌ای (دکمه open زده شد یا فایل دوبار کلیک شد) تصویر مورد نظر در pictureBox بارگذاری شود. با انتخاب فایل در کادر OpenFileDialog نام و مسیر فایل انتخاب شده در ویژگی FileName کنترل قرار می‌گیرد و رویداد FileOk برای کنترل رخ می‌دهد.

برای نوشتن رویداد FileOk، روی کنترل OpenFileDialog در محیط طراحی دوبار کلیک کنید یا با زدن دکمه Eventes در پنجره Properties (با علامت رعد و برق) از لیست رویدادهای کنترل استفاده کنید.

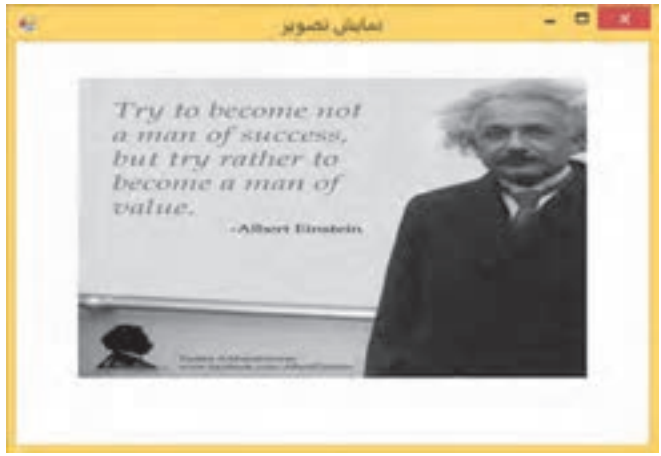
رویداد FileOk را مطابق زیر بنویسید.

```
private void openFileDialog1_FileOk(object sender, CancelEventArgs e)
{
    pictureBox1.ImageLocation = openFileDialog1.FileName;
}
```

برنامه ۱۲ــ۶

در این رویداد تصویر pictureBox1 بارگذاری می‌شود. بارگذاری از طریق آدرسی است که در ویژگی FileName کنترل openFileDialog1 قرار دارد. این آدرس به ویژگی ImageLocation کادر تصویر داده می‌شود. ویژگی ImageLocation آدرس یک تصویر را می‌گیرد و آن را در کادر تصویر بارگذاری می‌نماید.

توسعه و بهبود برنامه : روی فرم دکمه‌ای با عنوان تغییر تصویر قرار دهید و به جای کلیک جعبه تصویر با کلیکِ دکمه، تصویر جعبه تصویر را تغییر دهید.



شکل ۲۶ــ۶ــ نمایش تصویر انتخاب شده در کادر تصویر

کار در کارگاه ۹: کار با فونت و رنگ

مثال ۹ ــ۶ : کنترل کادر محاوره‌ای رنگ[۱] و کادر محاوره‌ای فونت[۲]

در این مثال می‌خواهیم یک ویرایشگر متنی ساده بسازیم که در آن بتوان رنگ پس زمینه و تنظیم‌های قلم (فونت) را مشخص کرد (شکل ۲۷ــ۶).

برای تغییر فونت از کادر محاوره‌ای فونت و برای تغییر رنگ از کادر محاوره‌ای رنگ استفاده می‌کنیم ، بنابراین در این مثال به کنترل‌های ColorDialog و FontDialog نیاز داریم.

الگوریتم یا روش انجام کار : این برنامه دارای یک جعبه متن ،دو دکمه یکی برای تغییر فونت و دیگری برای تغییر رنگ قلم، یک ColorDialog و یک FontDialog است. با زدن دکمه

شکل ۲۷ــ۶ــ فرم ویرایشگر متنی ساده

۱ ــ Color Dialog

۲ ــ Font Dialog

قلم ،کادر محاوره ای FontDialog باز می‌شود، فونت به وسیلهٔ کاربر انتخاب شده، روی جعبه متن اعمال می‌شود. با زدن دکمه «رنگ پس زمینه» ،کادر محاوره‌ای ColorDialog باز می‌شود و رنگ به وسیلهٔ کاربر انتخاب شده، ویژگی رنگ پس زمینه جعبه متن تنظیم می‌شود.

۱ــ ایجاد پروژه :واردبرنامهVS شویدوپروژه‌جدیدی‌ازنوعWindowsFromsApplication به نام EditorProj ایجاد نمایید.

۲ ــ درج کنترل‌ها :روی فرم دو کنترل Button و یک TextBox قرار دهید.

۳ ــ تنظیم ویژگی‌های دکمه‌ها و جعبه متن :ویژگی‌های جعبه متن و کنترل‌ها را طبق جداول زیر تنظیم کنید.

جدول ۳۵ــ۶ ــ ویژگی‌های جعبه متن

TextBox	
ویژگی	مقدار
Name	editor
Text	
RightToLeft	Yes
Multiline	True

جدول ۳۶ــ۶ ــ ویژگی‌های دکمه قلم

Button	
ویژگی	مقدار
Name	fontBtn

جدول ۳۷ــ۶ ــ ویژگی‌های دکمه رنگ زمینه

Button	
ویژگی	مقدار
Name	colorBtn

در کنترل TextBox در حالت عادی تنها می‌توان یک خط نوشت. اما این کنترل دارای ویژگی MultiLine است که از نوع Boolean می‌باشد و در صورتی که مقدار آن مساوی true باشد، جعبه متن چند خطی ایجاد می‌کند.

۴ ــ تنظیم ویژگی‌های کادرهای محاوره‌ای : دو کنترل FontDialog و ColorDialog را به فرم اضافه کرده و ویژگی آنها را طبق جداول زیر تنظیم کنید.

جدول۳۸ــ۶ــ ویژگی‌های کادر محاوره‌ای رنگ

ColorDialog	
ویژگی	مقدار
Name	colorDialog1

جدول۳۹ــ۶ــ ویژگی‌های کادر محاوره‌ای فونت

FontDialog	
ویژگی	مقدار
Name	fontDialog1
ShowColor	True

ویژگی ShowColor در کادر محاوره‌ای فونت امکان انتخاب رنگ قلم را نیز میسر می‌سازد.

۵ــ نوشتن رویداد click برای دکمه‌ها : در رویداد کلیک دکمه قلم، کادر محاوره‌ای تنظیمات قلم را نمایش می‌دهیم. برای این منظور متد ShowDialog() شئ fontDialog1 را فراخوانی می‌کنیم. سپس ویژگی‌های Font و ForeColor جعبه متن را با فونت و رنگ انتخاب شده در کادر محاوره‌ای فونت مقداردهی می‌کنیم.

```
private void fontBtn_Click(object sender, EventArgs e)
{
    fontDialog1.ShowDialog();
    editor.Font = fontDialog1.Font;
    editor.ForeColor = fontDialog1.Color;
}
```

برنامه ۱۳ــ۶

شکل ۲۸ــ۶ــ کادر محاوره‌ای فونت

در رویداد کلیک دکمه رنگ پس زمینه (colorBtn)، کادر محاوره‌ای انتخاب رنگ را نمایش می‌دهیم، بنابراین متد ShowDialog() شئ colorDialog1 را فراخوانی می‌کنیم، سپس رنگ پس‌زمینه جعبه متنی editor را با رنگ انتخاب شده مقداردهی می‌کنیم.

```
private void colorBtn_Click(object sender, EventArgs e)
{
   colorDialog1.ShowDialog();
   editor.BackColor = colorDialog1.Color;
}
```

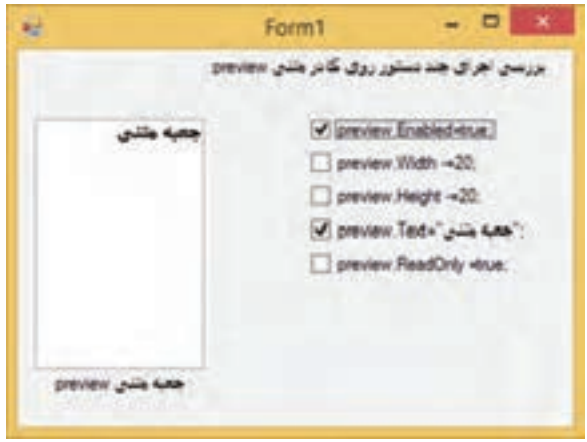
برنامه ۱۴ــ۶

کار در کارگاه ۱۰: کار با کنترل CheckBox

مثال ۱۰ــ۶ : کنترل کادر علامت[1]

در این کارگاه می‌خواهیم با نوشتن برنامه‌ای به کاربر اجازه دهیم که تغییر برخی از ویژگی‌های کنترل جعبه متن را در زمان اجرا مشاهده کند. برای دادن امکان انتخاب ویژگی‌ها به کاربر از کنترل CheckBox استفاده می‌کنیم. این کنترل در مواردی که دوحالت داریم مثل true/false یا بله / خیر کاربرد دارد. در کنار این کنترل مربع کوچکی قرار دارد که در زمان اجرا با کلیک روی آن می‌توان علامت در آن قرار داد و یا علامت آن را حذف کرد.

برنامه‌ای بنویسید که حاصل اجرای چند دستور روی یک جعبه متنی را نمایش دهد. این دستورات به صورت CheckBox قابل انتخاب باشند و حاصل اجرای آنها هم‌زمان، نمایش داده شود (شکل ۲۹ــ۶).



شکل ۲۹ــ۶ــ فرم اجرای هم‌زمان چند دستور روی جعبه متنی

الگوریتم یا روش انجام کار : این فرم دارای پنج CheckBox، دو Label و یک TextBox است. با توجه به CheckBoxهایی که دارای علامت هستند، گزینه دستوری مطابق با متن آنها، روی جعبه متنی اجرا می‌شود. اگر یک گزینه علامت‌دار باشد دستور آن اجرا می‌شود و درغیر این‌صورت عکس آن عمل اجرا می‌شود (چرا؟).

۱ــ ایجاد پروژه : وارد برنامه VS شوید و پروژه جدیدی از نوع WindowsFromsApplication

---

۱ ــ CheckBox

به نام CheckBoxProj ایجاد نمایید.

۲ــ درج کنترل‌ها : روی فرم پنج کنترل CheckBox ، دو Label و یک TextBox قرار دهید.

۳ــ تنظیم ویژگی‌های کادرهای علامت، برچسب‌ها و جعبه متنی : ویژگی‌های کادرهای علامت، برچسب‌ها و جعبه متنی را طبق جداول زیر تنظیم کنید.

جدول ۴۰ــ۶ــ ویژگی‌های کادر علامت ۱

checkBox2	
ویژگی	**مقدار**
Name	setWidth
Text	preview.Width –= 20;
RightToLeft	No
Checked	False

جدول ۴۱ــ۶ــ ویژگی‌های کادر علامت ۲

CheckBox1	
ویژگی	**مقدار**
Name	setEnabled
Text	preview.Enabled=true;
RightToLeft	No
Checked	True

کنترل CheckBox دارای ویژگی Checked است که از نوع Boolean می‌باشد و اگر مقدار آن مساوی true باشد، کنترل دارای علامت است.

برای آنکه با علامت‌دار شدن CheckBoxها عملیات مورد نظر روی جعبه متن انجام شود، در ویژگی Text کنترل‌ها دستور موردنظر قرار داده‌ایم.

جدول ۴۲ــ۶ــ ویژگی‌های کادر علامت ۴

checkBox4	
ویژگی	**مقدار**
Name	setReadOnly
Text	preview.ReadOnly= true;
RightToLeft	No
Checked	False

جدول ۴۳ــ۶ــ ویژگی‌های کادر علامت ۳

checkBox3	
ویژگی	**مقدار**
Name	setHeight
Text	preview.Height –= 20;
RightToLeft	No
Checked	False

جدول ۴۴ــ۶ــ ویژگی‌های جعبه متن

TextBox 1	
ویژگی	مقدار
Name	preview
Text	جعبه متنی
RightToLeft	Yes
Multuline	True

جدول ۴۵ــ۶ــ ویژگی‌های کادر علامت ۵

checkBox5	
ویژگی	مقدار
Name	setText
Text	preview.Text = "جعبه متنی"
RightToLeft	No
Checked	True

دو برچسب روی فرم داریم که یکی عنوان پروژه را نشان می‌دهد و دیگری عنوانی برای جعبه متنی مشخص می‌کند. ویژگی این برچسب‌ها مطابق جداول زیر است.

جدول ۴۶ــ۶ــ ویژگی برچسب۲

label 2	
ویژگی	مقدار
Name	label2
Text	جعبه متنی preview

جدول ۴۷ــ۶ــ ویژگی برچسب۱

label 1	
ویژگی	مقدار
Name	label1
Text	بررسی اجرای چند دستور روی کادر متنی preview

۴ــ نوشتن رویداد CheckedChanged برای کادرهای علامت: برای آنکه با علامت‌دار شدن یک CheckBox عملیات مورد نظر روی جعبه متن انجام شود، در ویژگی Text کنترل‌ها دستور مورد نظر را قرار داده‌ایم، بنابراین کافی است همان دستور را در کد بنویسیم. در چه رویدادی؟

با کلیک کنترل CheckBox، کنترل علامت‌دار شده یا علامت آن حذف می‌شود و برای کنترل رویداد CheckedChanged رخ می‌دهد؛ بنابراین در هنگام رخ دادن این رویداد، ابتدا باید بررسی کنیم که اگر CheckBox مورد نظر انتخاب شده (علامت‌دار) دستور ذکر شده در متنش را انجام دهیم و در غیر این‌صورت باید عملی عکس آن انجام شود تا جعبه متنی به حالت قبلش برگردد.

پس از آنکه رویداد CheckedChanged را برای همه CheckBoxها ایجاد کردیم کد آنها را به صورت برنامه ۱۵ــ۶ تکمیل کنید.

```
private void setWidth_CheckedChanged(object sender, EventArgs e)
{
    if (setWidth.Checked)
    preview.Width -= 20;
    else
    preview.Width += 20;
}
```

برنامه ۱۵ــ۶

در این رویداد با کمک ویژگی Checked کنترل، بررسی می‌شود که کادر علامت setWidth انتخاب شده است یا خیر؟ اگر جواب درست بود از عرض جعبه متنی ۲۰ پیکسل کم می‌شود و در غیراین‌صورت عکس این عمل انجام می‌شود. یعنی ۲۰ واحد به عرض جعبه متنی اضافه می‌گردد. (ویژگی Width پهنای کنترل و ویژگی Height ارتفاع کنترل‌ها را برحسب پیکسل مشخص می‌کند.)

```
private void setHeight _CheckedChanged(object sender, EventArgs e)
{
    if (setHeight.Checked)
    preview.Height -= 20;
    else
    preview.Height += 20;
}
```

برنامه ۱۶ــ۶

در این رویداد با کمک ویژگی Checked کنترل، بررسی می‌شود که کادر علامت setHeight انتخاب شده است یا خیر؟ اگر جواب درست بود از ارتفاع جعبه متنی ۲۰ پیکسل کم می‌شود و در غیر این‌صورت عکس این عمل انجام می‌شود. یعنی ۲۰ واحد به ارتفاع جعبه متنی اضافه می‌گردد.

```
private void setEnabled_CheckedChanged(object sender, EventArgs e)
{
    preview.Enabled = setEnabled.Checked;
}
```

برنامه ۱۷ــ۶

ویژگی Enabled کنترل جعبه متن مانند ویژگی Checked کنترل کادر علامت از نوع Boolean است؛ بنابراین در صورت انتخاب کنترل setEnabled باید مقدار Enabled مساوی true و درغیراین‌صورت false شود، پس در این برنامه مقدار این دو ویژگی مساوی یکدیگر است.

```
private void setText_CheckedChanged(object sender, EventArgs e)
{
    if (setText.Checked)
    preview.Text = "جعبه متنی";
    else
    preview.Text = " ";
}
```

برنامه ۱۸ــ۶

در این رویداد نیز با بررسی وضعیت انتخاب کادر علامت setText ، متن جعبه متنی را برابر با «جعبه متنی» می‌کند یا آن را خالی می‌نماید.

ویژگی ReadOnly کنترل جعبه متن مانند ویژگی Enabled از نوع Boolean است و اگر مساوی true باشد جعبه متن برای کاربر مانند کنترل برچسب می‌شود و کاربر نمی‌تواند چیزی در آن بنویسد.

```
private void setReadOnly _CheckedChanged(object sender, EventArgs e)
{
    preview.ReadOnly = setReadOnly.Checked;
}
```

برنامه ۱۹ــ۶

خودآزمایی فصل ششم

۱ــ هر ویژگی از سمت راست را با توضیح سمت چپ مطابقت دهید(یک توضیح اضافی است).

Visible	۱ــ تعیین نوع و اندازه قلم متن یک شئ
Text	۲ــ رنگ متن روی شئ
Name	۳ــ متن نمایش داده شده روی شئ
RightToLeft	۴ــ موقعیت شئ نسبت به گوشه بالا و سمت چپ فرم
Font	۵ــ تثبیت فاصله لبه‌های شئ از لبه‌های فرم
Anchor	۶ــ پشتیبانی شئ از زبان‌های راست به چپ
ForeColor	۷ــ پنهان کردن شئ
location	۸ــ نام شئ برای دسترسی به ویژگی‌ها و متدهایش
	۹ــ فعال کردن شئ

۲ــ در پنجره Properties، چطور متوجه می‌شویم که کدام شئ یا کنترل در روی فرم، انتخاب شده است؟

۳ــ لیستی از کنترل‌هایی که تاکنون با آنها آشنا شدید را روی یک طرف کاغذی به ابعاد ۵×۱۰ سانتی‌متر بنویسید و در طرف دیگر کاربرد آن را ذکر کنید. سپس فیش‌های نوشته شده را با هم کلاسی خود به اشتراک بگذارید.

۴ــ فیش نویسی سؤال ۳ را برای کنترل‌ها و ویژگی‌های آن انجام دهید. در یک طرف کاغذ نام کنترل و طرف دیگر، سه تا چهار مورد ویژگی‌هایی که کاربردی هستند را بنویسید. می‌توانید فیش‌ها را در کلاس به کمک معلم خود، در یک کارگروهی و مسابقه تمرین کنید.

۱ــ یک فرم مطابق با یک صفحه کلید بدون واکنش به رویدادها طراحی کنید.

۲ــ یک فرم مطابق با کنترل تلویزیون بدون واکنش به رویدادهای کلیک ماوس بسازید. احتمالاً این کنترل باطری ندارد!

۳ــ یک فرم شامل چهار تصویر که هر یک مربوط به یک فصل باشد، بسازید. در زیر هر یک از تصاویر، یک برچسب قرار داده و نام فصل مربوطه را بنویسید. طراحی صفحه را با دقت و با رعایت فاصله یکسان و مناسب تصاویر با یکدیگر و همچنین از لبه‌های فرم انجام دهید. از ابزارهای منوی

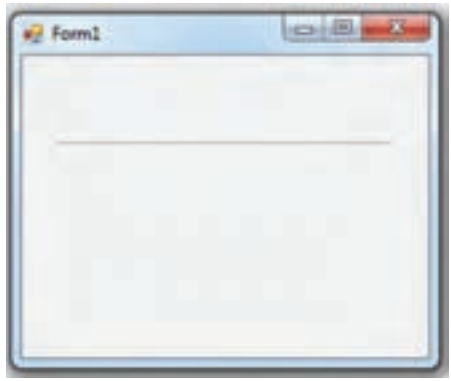
Format و یا در نوار ابزار، برای تنظیم موقعیت تصاویر و برچسب‌ها استفاده کنید.

۴ــ شبیه تمرین قبل را با موضوع تیم‌های ورزشی مورد علاقه خود بسازید. حداقل دو تیم ورزشی را در فرم معرفی کنید و لوگو تیم نشان داده شود.

۵ــ یک فرم شامل یک تصویر، یک جعبه متن و سه کادر علامت برای معرفی یکی از جاذبه‌های گردشگری شهر خود طراحی کنید. توضیحات آن جاذبه را در جعبه متن قرار دهید. سپس برنامه‌ای بنویسید که کاربر بتواند از طریق انتخاب یا عدم انتخاب کادرهای علامت، توضیحات را به زبان انگلیسی یا فارسی نمایش دهد و رنگ قلم و زمینه جعبه متن را تغییر دهد.

۶ــ تمرین زیر که به زبان انگلیسی ارائه شده است را بخوانید و سپس انجام دهید.

This example shows how to create a bevel line in Windows Forms. This line can be used as a visual separator of controls on a form.



To simulate the line in Windows Forms use a Label control. Set its Height to 2 pixels and BorderStyle to Fixed3D. That's all.

Now, create a new Windows Forms Application project and add a Label to the Form. Set its properties as shown in below table. Now, run the program.

Label Control	
Property	**Value**
AutoSize	False
BorderStyle	Fixed3D
Height	2
Width	250

فعالیت

The **OpenFileDialog** component allows users to browse the folders of their computer or any computer on the network and select one or more files to open. The dialog box returns the path and name of the file the user selected in the dialog box.

Use the **ShowDialog** method to display the dialog box and the OpenFile method to open the file.

واژگان و اصطلاحات انگلیسی فصل ششم

ردیف	واژه انگلیسی	معنی واژه به فارسی
۱	Anchor	
۲	Button	
۳	Button Control	
۴	CheckBox	
۵	Double Click	
۶	Interval	
۷	ReadOnly	
۸	Tulip	

پیوست ۱

تبدیل ضمنی انواع داده به یکدیگر در C#

گاهی هنگام استفاده از متغیرها و مقداردهی به آنها، مقداری که به متغیر منتسب می‌کنیم با نوع متغیر یکسان نیست اما سی شارپ پیام خطا نمی‌دهد. چرا؟

زیرا سی شارپ در صورتی که امکانش وجود داشته باشد به صورت ضمنی، نوع داده آن مقدار را به نوع داده متغیر تبدیل می‌کند. در جدول زیر مشخص شده که سی شارپ هر نوع داده را به کدام یک از انواع داده می‌تواند تبدیل کند.

ردیف	نوع داده	انواع داده قابل تبدیل
۱	sbyte	short, int, long, float, double, or decimal
۲	byte	short, ushort, int, uint, long, ulong, float, double, or decimal
۳	short	int, long, float, double, or decimal
۴	ushort	int, uint, long, ulong, float, double, or decimal
۵	int	long, float, double, or decimal
۶	uint	long, ulong, float, double, or decimal
۷	long	float, double, or decimal
۸	char	ushort, int, uint, long, ulong, float, double, or decimal
۹	float	double
۱۰	ulong	float, double, or decimal

تبدیل صریح (casting) انواع داده به یکدیگر در C#

گاهی هنگام انتساب یک عبارت یا مقدار به یک متغیر سی شارپ خطایی مبنی بر اینکه نمی‌تواند نوع داده مقدار یا عبارت را به نوع داده متغیر تبدیل کند می‌دهد. برای حل این مشکل می‌توان از تبدیل صریح نوع داده استفاده کرد. در جدول زیر مشخص شده که با روش تبدیل صریح هر نوع داده را به کدام یک از انواع دیگر داده می‌توان تبدیل کرد.

جدول تبدیل صریح (casting) انواع داده به یکدیگر

ردیف	نوع داده	انواع داده قابل تبدیل
1	sbyte	byte , ushort, uint, ulong, or char
2	byte	sbyte or char
3	short	sbyte , byte, ushort, uint, ulong, or char
4	ushort	sbyte , byte, short, or char
5	int	sbyte , byte, short, ushort, uint, ulong, or char
6	uint	sbyte , byte, short, ushort, int, or char
7	long	sbyte , byte, short, ushort, int, uint, ulong, or char
8	char	sbyte , byte, or short
9	float	sbyte , byte, short, ushort, int, uint, long, ulong, char, or decimal
10	ulong	sbyte , byte, short, ushort, int, uint, long, or char
	double	sbyte , byte, short, ushort, int, uint, long, ulong, char, float, or decimal
	decimal	sbyte , byte, short, ushort, int, uint, long, ulong, char, float, or double

پیوست ۲

الگوی نمایش اطلاعات

در جدول زیر لیستی از تعیین کننده‌های فرمت (format specifiers) که در الگوی نمایش {تعیین کننده فرمت : عدد} یا الگوی نمایش {تعیین کننده فرمت : عدد تراز، عدد} برای قالب بندی اعداد به کار می‌روند، آمده است :

Specifier	Format
C	مقدار پولی خاص یک محل
D	انواع صحیح (integer)
E	نماد علمی
F	نقطه اعشار ثابت
G	اعداد عمومی
N	نقطه اعشار ثابت با جدا کننده کاما
P	اعداد دارای درصد
X	هگزادسیمال

تأثیر دقت یک قالب خاص بستگی به تنظیمات منطقه‌ای دارد. به عنوان مثال قالب پولی C، به‌طور خودکار قالب پولی منطقه انتخاب شده را نشان می‌دهد. برای اکثر کاربران این اطلاعات به طور پیش‌فرض مربوط به منطقه و زبان آنها می‌باشد.

قالب بندی سفارشی اعداد

سی‌شارپ به شما اجازه قالب بندی اعداد به صورت سفارشی را نیز می‌دهد. برای این کار کاراکترهای خاصی در اختیار شما می‌گذارد که در جدول صفحه بعد لیست آنها را مشاهده می‌کنید :

کاراکتر	معنی
#	عدد
.	اعشار
,	جدا کننده سه رقمی
%	درصد
0	اضافه کردن صفر
;	اعداد مثبت، منفی و مقادیر صفر را با قالب‌های خاصی نمایش می‌دهد.
E0 E+0 E – 0 e0 e+0 e – 0	نماد علمی

نقطه مکان اعشار را مشخص می‌کند. علامت # می‌تواند در برگیرنده عددی باشد که قرار است در سمت چپ یا راست آن اعشار قرار بگیرد. اگر این علامت در سمت راست اعشار قرار بگیرد دقت اعشار را مشخص می‌کند و ممکن است در صورت لزوم عدد را رند کند و اگر در سمت چپ ممیز باشد نماینده قسمت صحیح عدد می‌باشد. اگر طول عدد از تعداد علامت‌های # که در سمت چپ علامت اعشار قرار دارند بیشتر باشد، کل عدد نشان داده می‌شود.

کاراکتر E و e برای نمایش اعداد به صورت نماد علمی به کار می‌روند. حداقل یک صفر و در صورت لزوم تعداد بیشتری صفر بعد از E و e می‌آیند. صفرها تعداد ارقام اعشار را نشان می‌دهند. استفاده از حرف E و یا e ممکن است باعث چاپ این حروف در خروجی شوند. در نتیجه برای تعریف توان مثبت و منفی می‌توان از علامت – و + بعد از این دو کاراکتر به صورت E+، e+ و یا E–، e– استفاده کرد. علامت ";" هم شما را قادر می‌سازد که اعداد مثبت، منفی و مقادیر صفر را با قالب‌های خاصی جدا سازید.

پیوست ۳

متدهای کلاس String

متد	شرح کار متد	نوع برگشتی
Clone()	ایجاد یک کپی از رشته	string
CompareTo()	مقایسه دو رشته	bool
Contains()	جستجوی یک کاراکتر یا یک رشته	bool
EndsWith()	آیا رشته به کاراکتر مورد نظر ختم می‌شود؟	bool
Equals()	مقایسه دو رشته	bool
IndexOf()	موقعیت وجود یک کاراکتر در رشته را بر می‌گرداند.	int
ToLower()	تبدیل تمام کاراکترهای یک رشته به حروف کوچک	string
ToUpper()	تبدیل تمام کاراکترهای یک رشته به حروف بزرگ	string
Insert()	درج یک کاراکتر یا یک رشته در درون یک رشته دیگر	string
LastIndexOf()	موقعیت آخرین تکرار یک کاراکتر در درون یک رشته	int
Length	تعداد کاراکترهای یک رشته را بر می‌گرداند.	int
Remove()	حذف کاراکترهای یک رشته تا یک مکان مشخص	string
Replace()	یک کاراکتر در یک رشته را با کاراکتر یا یک رشته دیگری عوض می‌کند.	string
Split()	یک رشته را به چند قسمت تقسیم می‌کند.	String[]
StartsWith()	آیا رشته با یک کاراکتر خاص شروع می‌شود؟	bool
Substring()	یک قسمت از یک رشته را جدا می‌کند و در اختیار قرار می‌دهد.	string
Trim()	فاصله و کاراکترهای Tab را از ابتدا و انتهای رشته حذف می‌کند.	string