خط دوم یک دستور اجرایی است :

```
System.Console.WriteLine();
```

در این خط از متد ( ) WriteLine استفاده شده است با این تفاوت که داخل پرانتز، خالی است. اجرای این دستور سبب می‌شود که روی صفحه نمایش یک سطر خالی ایجاد شود. دستور آخر، پیام .This is my first program را روی صفحه نمایش نشان می‌دهد.

با توجه به برنامه ۲ـ۲، پیامی را که می‌خواهید نمایش داده شود باید بین علامت های نقل قول " " قرار دهید. برای مثال اگر اسم شما Mohammad است و بخواهید روی صفحه نشان داده شود، باید به‌صورت زیر بنویسید :

```
System.Console.WriteLine ("Mohammad");
```

اگر بخواهید نام و نام خانوادگی خود را در دو سطر، نمایش دهید، در اینصورت می‌توانید دو بار از متد WriteLine() استفاده کنید. مثلا برای نمایش "Mohammad " و " Ghasemi" چنین می‌نویسیم :

```
System.Console.WriteLine("Mohammad");
System.Console.WriteLine("Ghasemi");
```

حروف، علامت‌ها و عبارتی که ما بین علامت‌های نقل قول نوشته می‌شود را **رشته**[1] می‌نامند. این حروف می‌تواند فارسی، انگلیسی یا به هر زبانی باشد. مانند نام "Mohammad" و یا یک رمز عبور "Mehran2014". هر یک از حروف و علامت‌ها را نیز یک **کاراکتر**[2] می‌نامند. برای مثال Mohammad از ۸ کاراکتر و رمز عبور Mehran2014 از ۱۰ کاراکتر تشکیل شده است. چنان‌چه فاصله در رشته وجود داشته باشد، فاصله نیز یک کاراکتر محسوب می‌شود.

نکته

برای درج توضیحات در برنامه، اگر یک خط باشد از علامت // و چنان‌چه چند خط باشد از علامت /* توضیحات */ استفاده کنید :

```
// Display a greeting message
/*
    FileName: welcome.cs ... Date : 05_07_2014
    Display a greeting message
*/
```

۱ـ Str ng ۲ـ Character

برای مطالعه

زبان برنامه‌ویسی C# از لایه نرم‌افزاری .NET. استفاده می‌کند و مایکروسافت .NET را برای ویندوز طراحی کرده و به این ترتیب روی ویندوز کار می‌کند و اگر کسی بخواهد بر روی سیستم عامل دیگری، برنامه C# را اجرا کند باید لایه نرم‌افزاری مطابق با .NET را بر روی آن سیستم داشته باشد. خوشبختانه برنامه‌هایی مانند Mono وجود دارند که بر روی سیستم عامل‌های دیگری غیر از ویندوز نیز نصب می‌شود و دارای مترجم C# می‌باشد و به این وسیله برنامه‌های نوشته شده به این زبان را بر روی سیستم‌های دیگر غیر از ویندوز می‌توان ترجمه و اجرا کرد. برنامه Mono برای سیستم عامل‌های زیر وجود دارد :

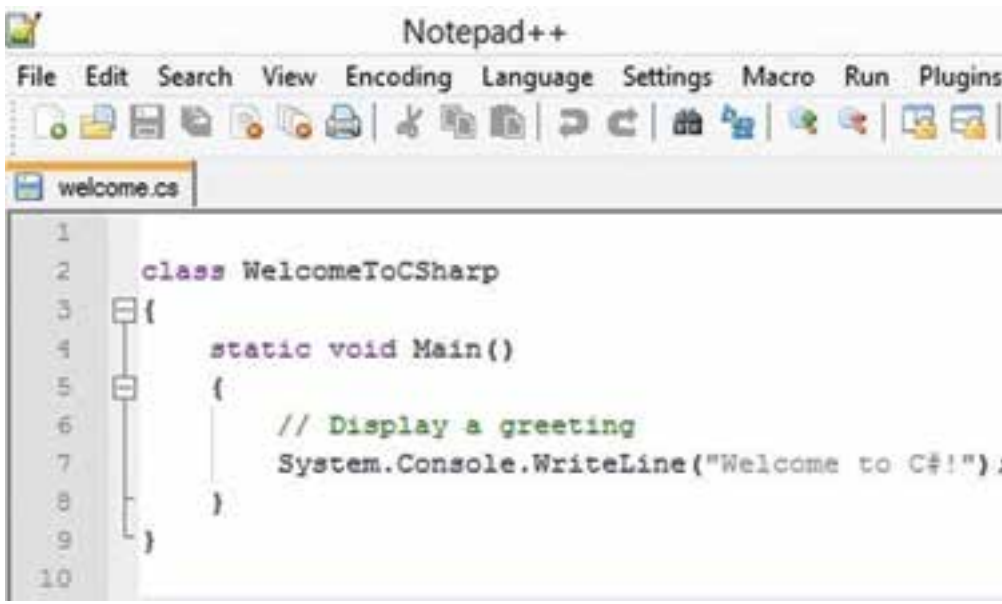
Android, BSD, iOS, Linux, OS X, Windows, Solaris and UNIX

سری به سایت Mono بزنید. www.go mono.com

شکل ۲ــ۲ــ برنامه نویسی C# بر روی سیستم عامل غیر از ویندوز

قدم اول : نوشتن و تایپ برنامه : برای نوشتن یک برنامه ساده مانند برنامه ۱ــ۲، که در این فصل مورد بررسی قرار گرفت، نیاز به یک ویرایشگر متنی[۱] است. یک ویرایشگر متنی قادر است حروف، کلمات و آنچه را که تایپ می‌کنید بدون در نظر گرفتن اطلاعات نوع فونت، اندازه حروف و رنگ در یک فایل ذخیره کند. برنامه Notepad یک ویرایشگر ساده است که همراه با سیستم عامل ویندوز در روی کامپیوتر نصب می‌شود.

علاوه بر برنامه Notepad، می‌توانیم از ویرایشگر دیگری مانند برنامه Notepad که به‌صورت رایگان از طریق سایت[۲] آن قابل دانلود است استفاده نماییم. این ویرایشگر به‌منظور برنامه‌نویسی به زبان‌های مختلف طراحی شده است به‌طوری که کلمات رزرو شده، رشته‌ها و توضیحات در این ویرایشگر با رنگ های مختلف نشان داده می‌شود (شکل ۳ــ۲). البته برای استفاده از این ویژگی باید ابتدا از منوی Language، زبان برنامه‌نویسی موردنظر خود را C# انتخاب کنید.



شکل ۳ــ۲ــ تصویری از محیط ویرایشگر Notepad

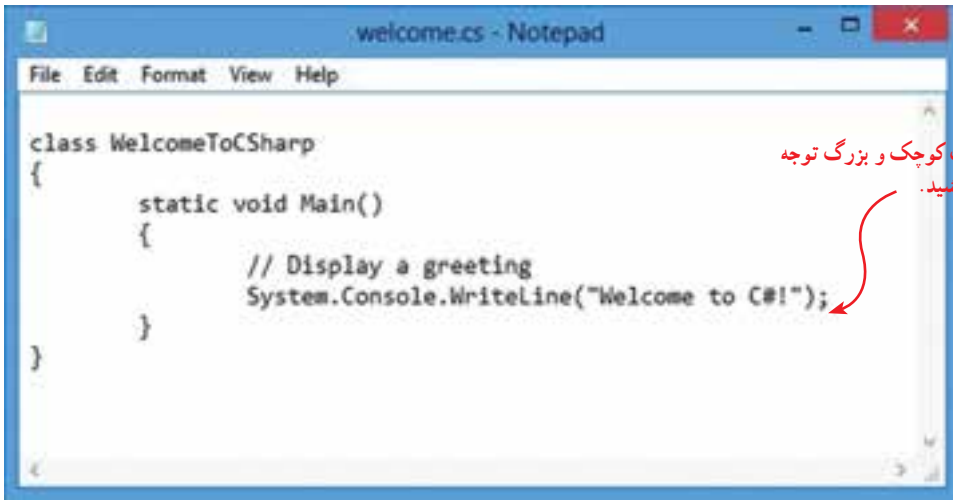
۱ــ Text Ed tor

۲ــ http: //notepad p us p us.org

توجه داشته باشید که از **برنامه Word** استفاده نکنید که در این صورت، کدهای اضافی مربوط به صفحه‌بندی، رنگ و فونت را نیز به فایل شما اضافه می‌کند که مترجم در هنگام ترجمه برنامه، انتظار آنها را ندارد و دچار مشکل می‌شود.

● در هنگام تایپ دستورات، صرفنظر از این که از چه ویرایشگری (Notepad , Notepad) استفاده می‌کنید، باید دقت داشته باشید که زبان C# نسبت به حروف کوچک و بزرگ حساس است؛ بنابراین برنامه را دقیقا مانند کتاب تایپ کنید (شکل ۴ــ۲).

پس از آن که برنامه ۱ــ۲ را تایپ کردیم، باید آن را ذخیره نماییم. برای این منظور به منوی File بروید و گزینه Save As... را انتخاب کنید. در هنگام ذخیره فایل دقت کنید که فایل را در کجا و در چه مسیری ذخیره می‌نمایید و نام فایل مناسبی را برای آن انتخاب کنید و پسوند آن را cs. قرار دهید (شاید بهتر باشد یک پوشه در یکی از درایوها به نام خودتان بسازید و فایل را در آن ذخیره کنید). در این تمرین نام فایل را welcome.cs قراردهید.

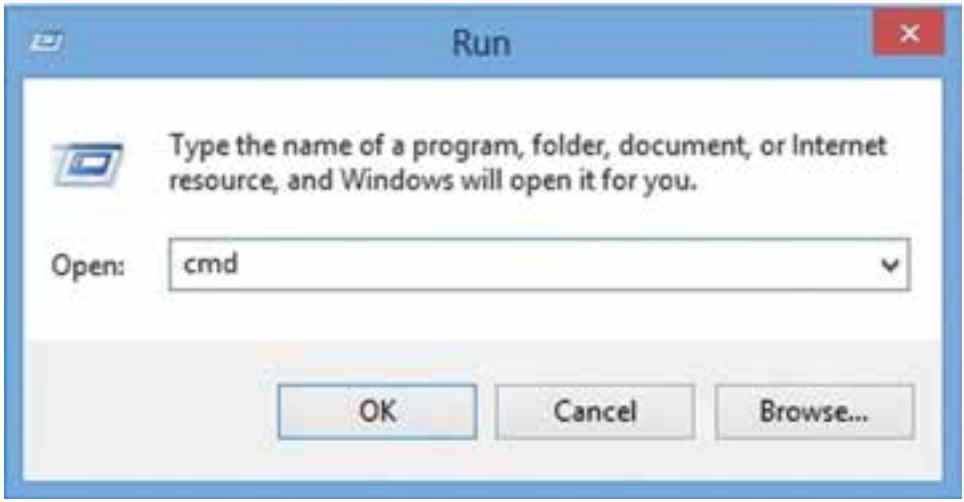


شکل۴ــ۲ــ تصویری از محیط ویرایشگر Notepad

قدم دوم : ترجمه و اجرای برنامه : پس از نوشتن و ذخیره کردن برنامه، لازم است ابتدا برنامه را ترجمه کنیم و اگر اشکالی در تایپ آن وجود دارد آن را برطرف کنیم.

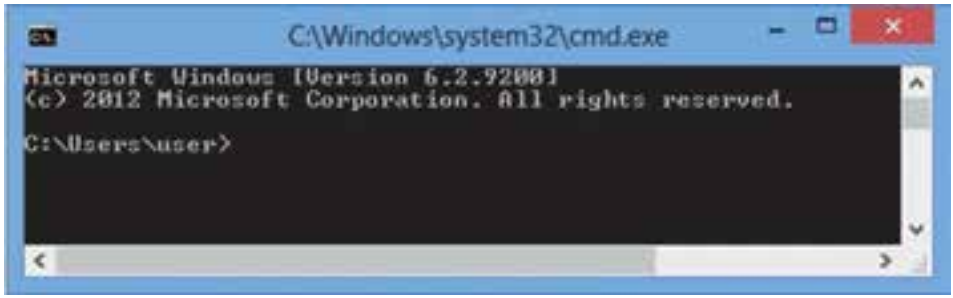
برای ترجمه برنامه مراحل زیر را دنبال کنید.

۱ــ از طریق گزینه Run فرمان cmd را اجرا کنید، تا وارد پنجره فرمان شویم (شکل ۵ــ۲).



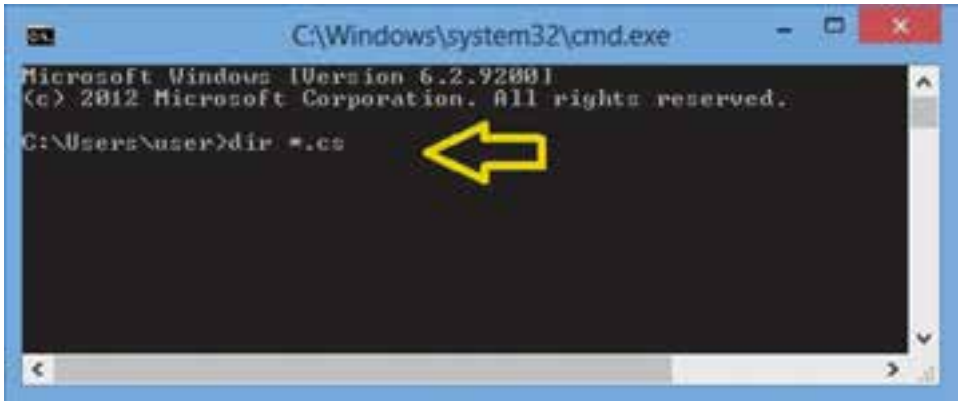
شکل ۵ــ۲ــ فرمان cmd

۲ــ پنجره فرمان ظاهر می‌شود (نوشته‌ها در شکل، ممکن است با آنچه در پنجره command prompt کامپیوتر شما دیده می‌شود متفاوت باشد) (شکل ۶ــ۲).



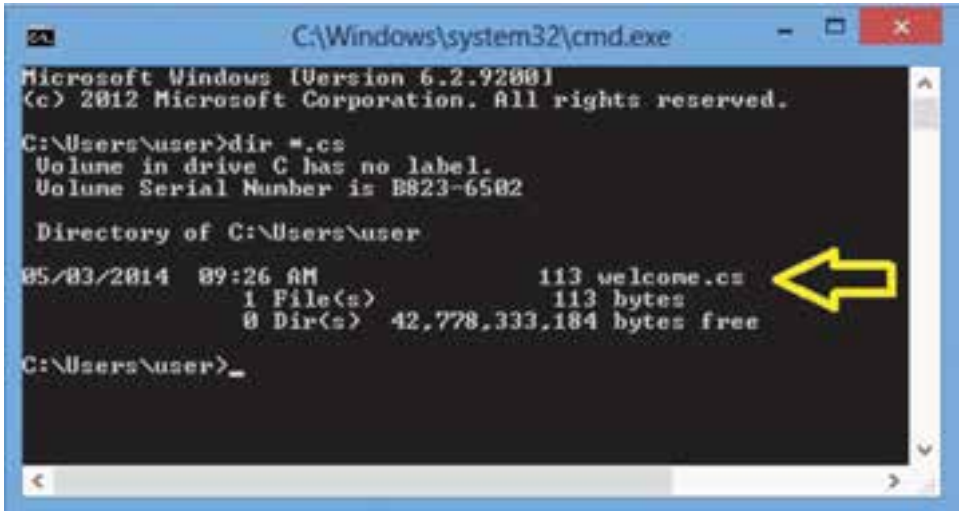
شکل ۶ــ۲ــ پنجره command prompt

۳ــ در پنجره Command Prompt از فرمان Dir استفاده می‌کنیم و با توجه به این که پسوند فایل cs می‌باشد، با تایپ فرمان زیر از وجود فایل برنامه مطمئن می‌شویم (شکل ۷ــ۲). اگر فایل را پیدا نکردید باید وارد پوشه‌ای شوید که برنامه را در آنجا ذخیره کرده‌اید. به این ترتیب از دستور cd برای وارد شدن به پوشه موردنظر خود استفاده کنید. شاید دستور cd.. نیز برای وقتی که می‌خواهید به یک پوشه بالاتر بروید مفید باشد.



شکل ۷ــ۲ــ فرمان Dir

۴ــ اگر مرحله قبل را به درستی انجام دهید، باید شکل ۸ــ۲ را مشاهده کنید.

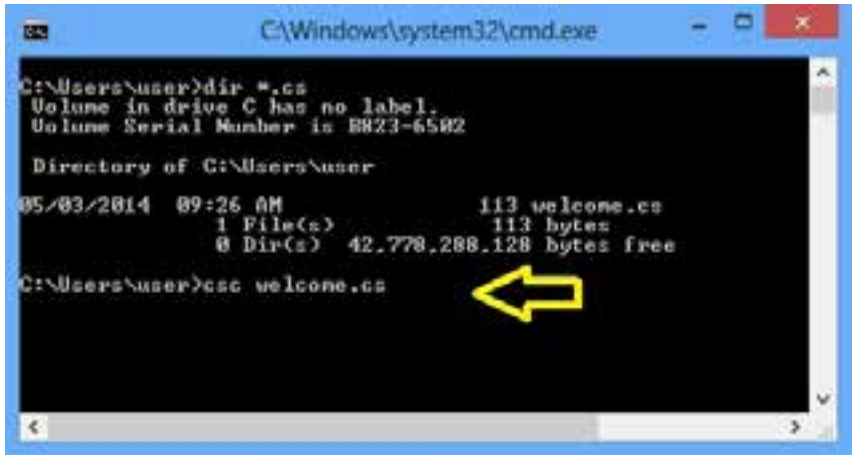


شکل ۸ــ۲ــ مشخصات فایل برنامه

۵ــ توجه داشته باشید اگر برنامه Net Framework. بر روی کامپیوتر شما قبلاً نصب شده باشد برنامه ای به نام csc.exe[۱] برای ترجمه برنامه های زبان C# در اختیار دارید. پس از یافتن فایل خود، با استفاده از این مترجم، برنامه خود را ترجمه نمایید. در پنجره فرمان از دستور زیر استفاده کنید :

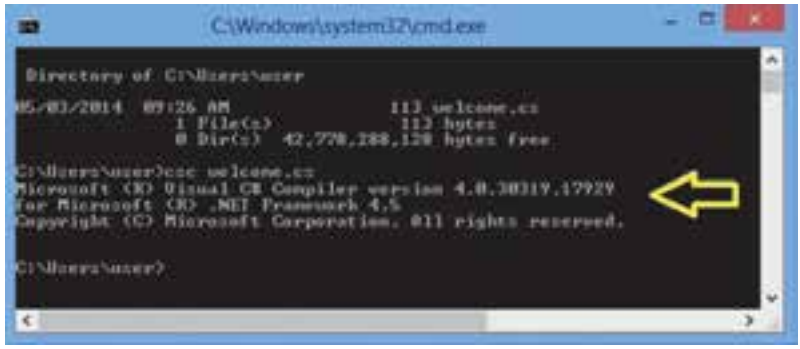
```
csc filename.cs
```

منظور از filename.cs نام فایل مورد نظر شما است.



شکل ۹ــ۲ــ فرمان ترجمه برنامه

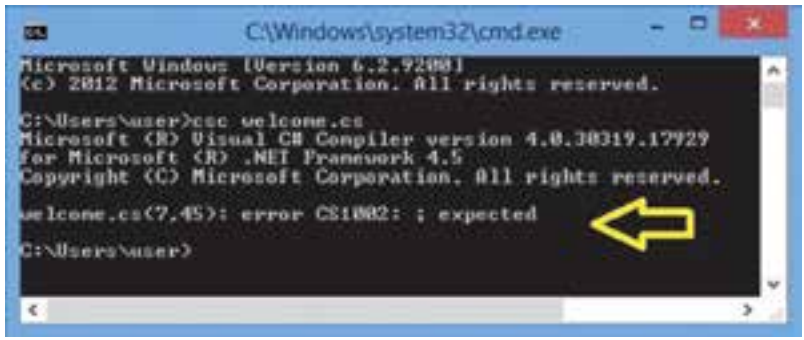
۶ــ با اجرای دستور بالا مترجم شروع به ترجمه برنامه می کند و اگر همه کارها به درستی انجام شده باشد (برنامه NET. قبلاً نصب شده باشد، تایپ دستور csc را درست انجام داده باشید، فایل csc به درستی در مسیر جستجوی سیستم عامل (path) معرفی شده باشد و برنامه هیچ خطایی نداشته باشد.) شکل ۱۰ــ۲ را مشاهده خواهید کرد.



شکل ۱۰ــ۲ــ نتیجه ترجمه برنامه

۱ــ C Sharp Comp er

۷ــ اگر در تایپ برنامه اشتباهی انجام داده باشید، مترجم نمی‌تواند برنامه را ترجمه کند و در این‌صورت خطا یا خطاهای برنامه را گزارش می‌دهد. برای مثال اگر فراموش کرده باشید علامت نقطه ویرگول را در انتهای دستور System.Console.WriteLine() بنویسید، با ترجمه برنامه خطای شکل ۷ــ۲ ظاهر می‌شود.



شکل ۱۱ــ۲ــ پیام خطای مترجم برای ننوشتن علامت ; در خط ۷ برنامه

در خط آخر این شکل مشاهده می‌کنید که از سمت چپ، ابتدا نام فایل برنامه یعنی welcome.cs و سپس در جلوی آن دو عدد ۷ و ۴۵ نوشته شده است که مکان خطا را در برنامه نشان می‌دهد. عدد اول (۷) شماره سطر و عدد دوم (۴۵) شماره ستون محل خطا در برنامه است. بعد از این دو عدد، کد خطا (error CS1002) و سپس توضیح آن (; expected) ذکر شده است. شما باید توضیح خطا را با دقت بخوانید و معنی آن را درک کنید تا بتوانید اشکال برنامه را برطرف کنید. در این مثال توضیح خطا چنین است : علامت ; فراموش شده

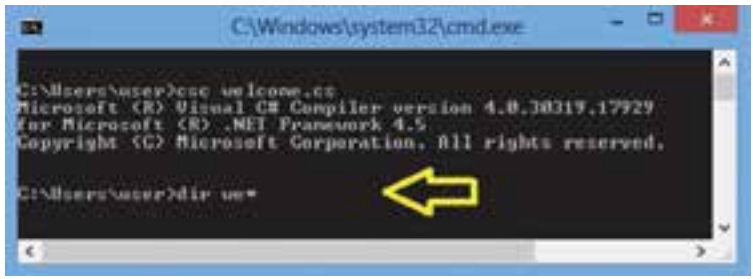
برای برطرف کردن خطای بالا لازم است برنامه را به وسیلهٔ ویرایشگر باز کنید و به سطر ۷ و ستون ۴۵ بروید و علامت نقطه ویرگول را اضافه کنید و سپس برنامه را تحت همان نام قبلی ذخیره کنید و دوباره به پنجره فرمان بازگردید و عمل ترجمه را انجام دهید. اگر خطایی رخ نداد به مرحله بعدی بروید وگرنه باید دوباره عملیات رفع اشکال را تکرار کنید.

نکته

اشتباه معمول برنامه نویسان در هنگام نوشتن برنامه، فراموش کردن علامت نقطه ویرگول در انتهای دستورات است. در این صورت خطای صادره از طرف مترجم چنین است :

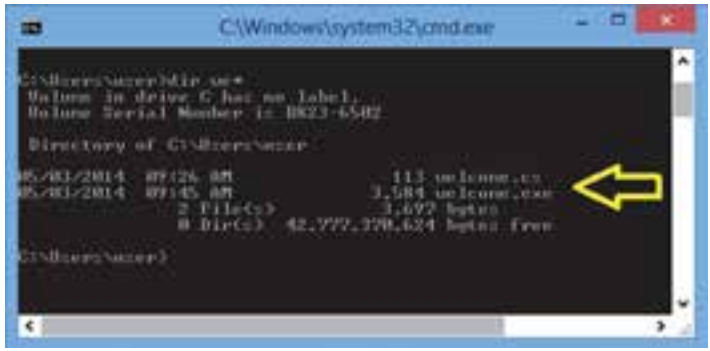
error CS1002: ; expected

۸ــ در صورت ترجمه موفق برنامه، فایل جدیدی ساخته می‌شود. با دستور Dir از وجود آن مطمئن می‌شویم (شکل ۱۲ــ۲).



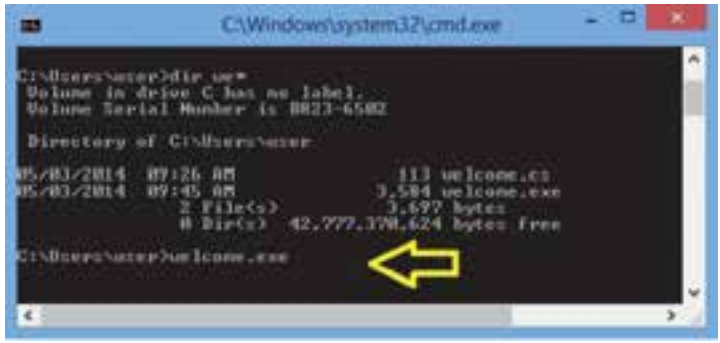
شکل۱۲ــ۲ــ جستجوی فایل ترجمه شده

۹ــ همان طور که در شکل ۱۳ــ۲ مشاهده می‌کنید، فایل جدیدی با همان نام welcome ولی با پسوند EXE ساخته شده است.



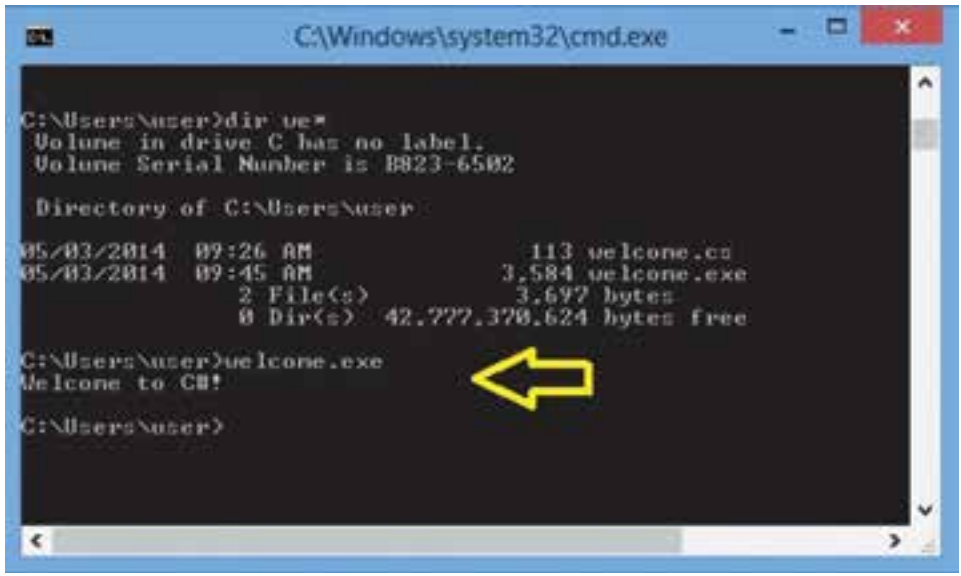
شکل۱۳ــ۲ــ لیست فایل های برنامه و ترجمه شده

۱۰ــ بعد از اینکه فایل اجرایی ساخته شد، می‌توانید برنامه را اجرا نمایید. کافی است نام آن را در پنجره فرمان بنویسید و کلید Enter را بزنید (شکل ۱۴ــ۲).



شکل۱۴ــ۲ــ اجرای فایل ترجمه شده یا برنامه اجرایی

۱۱ــ نتیجه اجرای برنامه به‌صورت شکل ۱۵ــ۲ خواهد بود.

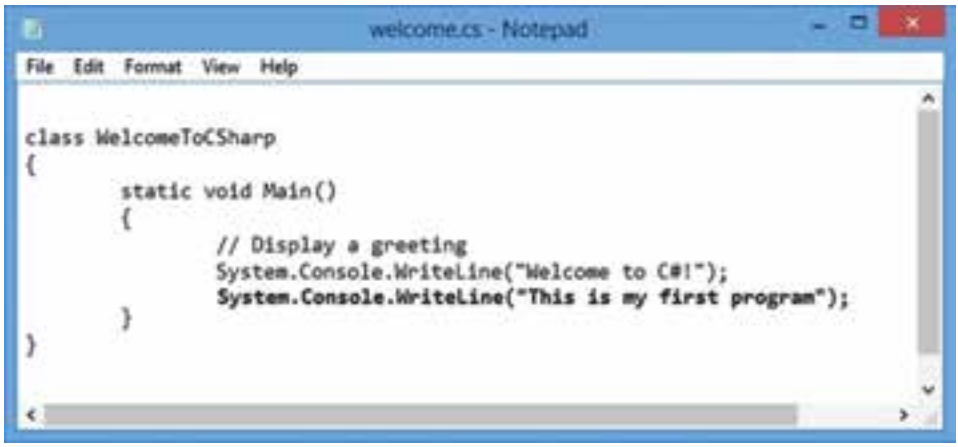


شکل ۱۵ــ۲ــ نتیجه اجرای برنامه

تبریک می‌گوییم که توانستید اولین برنامه خود را به زبان C# بنویسید و آن را ترجمه و اجرا نمایید.

۱۲ــ برنامه را با ویرایشگر باز کنید و یک دستور مانند شکل ۱۶ــ۲ به آن اضافه کنید. (تغییرات با رنگ تیره مشخص شده است.)

برنامه را تحت همان نام قبلی ذخیره کنید و مراحل ترجمه و اجرا (مرحله ۵ به بعد) را تکرار کنید.



شکل۱۶ــ۲ــ اضافه کردن یک دستور به برنامه

۱۳ــ دوباره برنامه را با ویرایشگر باز کنید و در اولین دستور به جای متد ()WriteLine از متد ()Write مانند شکل ۱۷ــ۲ استفاده کنید.

شکل۱۷ــ۲ــ استفاده از متد ()Write

چه تفاوتی در اجرای برنامه حاصل می‌شود؟ پیام‌ها چگونه نشان داده شدند؟

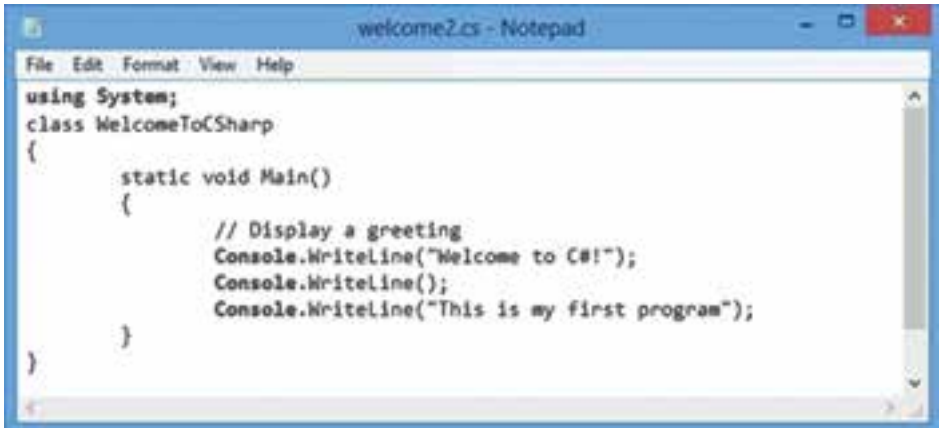
۱۴ــ برنامه را به حالت اولیه خود بازگردانید و یک دستور نیز به‌صورت شکل ۱۸ــ۲ به آن اضافه کنید. برنامه را ذخیره کنید و سپس ترجمه و اجرا نمایید. چه تغییری در خروجی ایجاد می‌شود؟

شکل ۱۸ــ۲ــ استفاده از متد ()WriteLine برای ایجاد یک خط خالی

۱۵ــ همان‌طور که در شکل ۱۸ــ۲ مشاهده می‌کنید، برای هر بار استفاده ازمتد WriteLine() مجبور هستیم کلمات System و Console را ذکر کنیم. برای کوتاه کردن این دستورات می‌توانیم در ابتدای برنامه، فضای نامی[1] System را معرفی کنیم که در آن کلاس Console تعریف شده است. در این‌صورت می‌توانیم در داخل برنامه، کلمه System را از ابتدای دستورات حذف کنیم. برای معرفی فضای نامی از دستور using به‌صورت زیر استفاده می‌کنیم :

```
using فضای نامی ;
```

برنامه قبلی را باز کرده و تغییرات زیر را اعمال کنید و آن را اجرا نمایید (شکل ۱۹ــ۲).



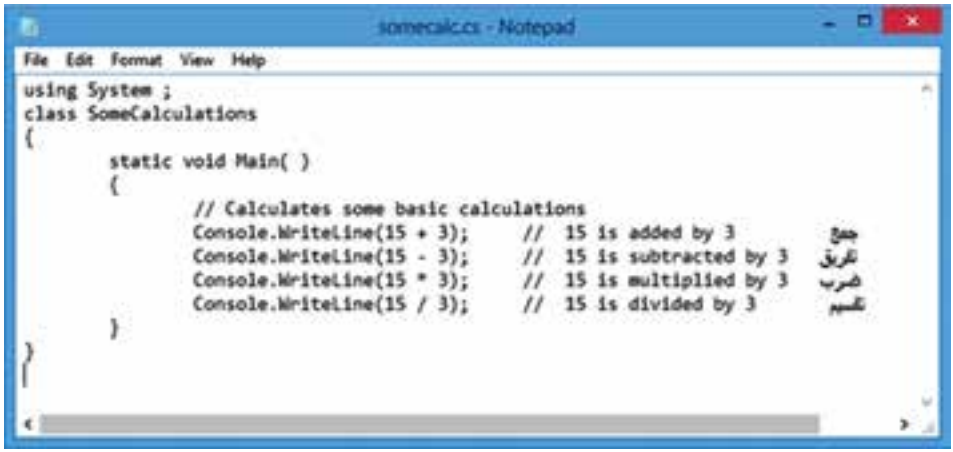
شکل۱۹ــ۲ــ استفاده از فضای نام

۱۶ــ از متد WriteLine() علاوه بر نمایش یک پیام می‌توان نتیجه یک عبارت ریاضی را نیز نمایش داد. برنامه‌ای برای انجام چهار عمل اصلی بر روی دو عدد نوشته شده است (شکل ۲۰ــ۲). این برنامه را با توجه به نکته زیر تایپ نموده و سپس ترجمه و اجرا نمایید. و به اعداد نشان داده شده برروی صفحه توجه کنید. آیا محاسبات درست انجام شده است؟

نکته

برای این‌که سریع‌تر بتوانید این برنامه را تایپ کنید، یکی از برنامه‌های قبلی خود را با ویرایشگر باز کنید و فقط نام کلاس و دستورات داخل متد Main را تغییر دهید و سپس تحت نام جدیدی آن را ذخیره (Save As...) کنید. توضیحات برنامه می‌تواند به زبان فارسی نیز باشد.

۱ــ برای توضیح بیشتر به ضمیمه ۴ در انتهای کتاب مراجعه کنید.



شکل ۲۰ــ۲ــ برنامهٔ چهار عمل اصلی

۱۷ــ برنامه شکل ۲۰ــ۲ را با اعداد دیگری آزمایش کنید یعنی به جای ۳ و ۱۵ اعداد دلخواه دیگری مانند ۲۵ و ۴۰ قرار دهید و سپس برنامه را ترجمه و اجرا نمایید. اعدادی که روی صفحه نشان داده می‌شوند را یادداشت کنید، آیا محاسبات درست انجام شده است؟

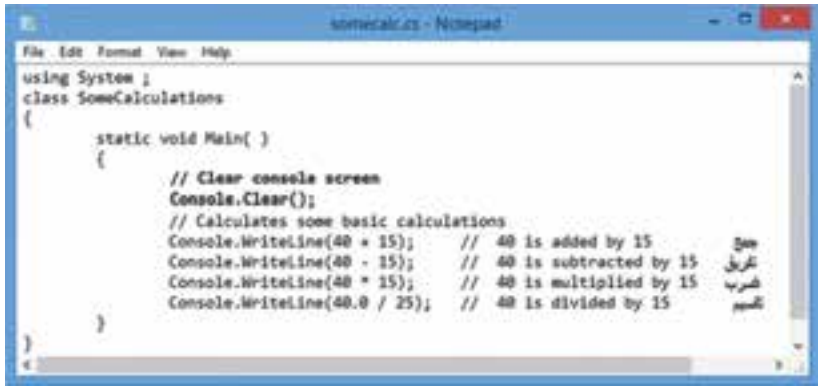
۱۸ــ اگر به خروجیِ دستور زیر توجه کنید

```
Console.WriteLine(40 / 25);
```

مشاهده خواهید کرد که عدد ۱ چاپ می‌شود چون تقسیم صحیح و بدون اعشار انجام می‌شود. در صورتی‌که جواب صحیح و دقیق عدد ۱.۶ است. برای این که کامپیوتر را مجبور به انجام عمل تقسیم اعشاری کنیم، لازم است دست کم یکی از اعداد را به صورت اعشاری بنویسیم. یعنی عدد ۴۰ را به صورت ۴۰.۰ یا عدد ۲۵ را به صورت ۲۵.۰ بنویسید. حال برنامه را در ویرایشگر باز کرده ویکی از اعداد در دستور تقسیم را به‌صورت اعشاری بنویسید و سپس برنامه را با همان نام قبلی ذخیره و ترجمه و اجرا نمایید.

```
Console.WriteLine(40.0 / 25);
```

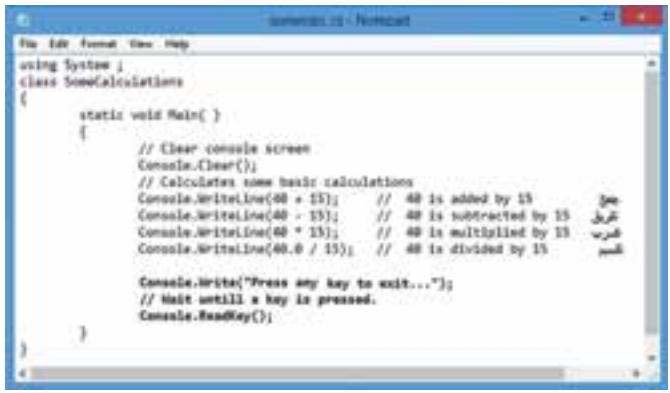
۱۹ــ برای این که پیام ها بهتر دیده شوند، می‌توانیم ابتدا با استفاده از متد Clear() صفحه نمایش را پاک کنیم. به این منظور برنامه را با ویرایشگر باز کنید و متد مذکور را به برنامه طبق شکل ۲۱ــ۲ اضافه کنید. برنامه را تحت نام فعلی ذخیره کنید و سپس ترجمه و اجرا نمایید.



شکل ۲۱ــ۲ــ استفاده از متد ()Clear برای پاک کردن صفحه

۲۰ــ پنجره فرمان را ببندید وارد محیط میزکار[۱] شوید و با استفاده از Computer وارد پوشه‌ای شوید که فایل‌های خود را در آن ذخیره کرده‌اید. فایل‌های اجرایی (EXE) تولید شده را پیدا کنید. با دوبار کلیک بر روی آیکن آن، برنامه را اجرا کنید. چه اتفاقی می‌افتد؟

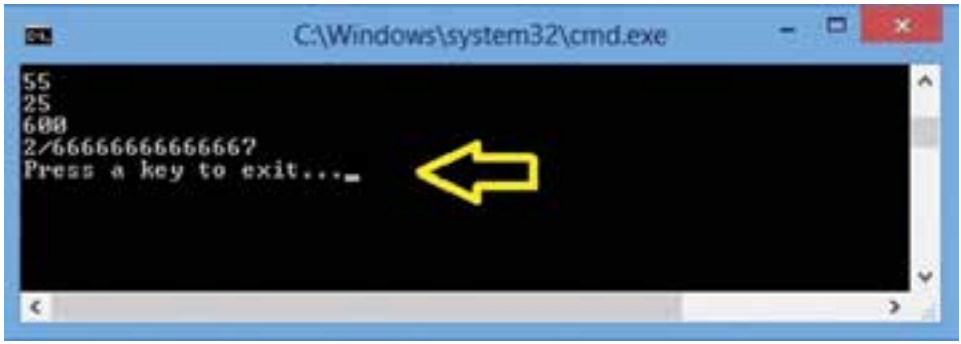
در یک لحظه صفحه کنسول باز شده و به سرعت بسته می‌شود و شما فرصت نمی‌کنید نتایج برنامه را مشاهده کنید. لازم است در انتهای برنامه، دستوری بنویسید که کامپیوتر را به‌صورت موقتی متوقف کند تا شما بتوانید نتایج محاسبات را ببینید. به این منظور از متد ()ReadKey استفاده می‌کنیم. با اجرای این متد، کامپیوتر منتظر زدن کلیدی باقی می‌ماند. البته برای این که کاربر بداند که چرا اجرای برنامه متوقف شده است و کامپیوتر منتظر دریافت چه چیزی است، یک پیام با استفاده از متد ()Write روی صفحه چاپ می‌کنیم. شکل ۲۲ــ۲ استفاده از این متدها را نشان می‌دهد. برنامه چهار عمل اصلی را باز کنید و تغییرات زیر را در آن اعمال و برنامه را ترجمه و اجرا کنید.



شکل ۲۲ــ۲ــ استفاده از متد ()Clear برای پاک کردن صفحه

۱ــ Desktop

با اجرای برنامه، مشاهده خواهید کرد که نتایج محاسبات و پیام ...Press any key to exit روی صفحه نشان داده می‌شود و سپس کامپیوتر متوقف می‌شود و با زدن یک کلید برنامه پایان می‌یابد (شکل ۲۳ــ۲).



شکل ۲۳ــ۲ــ استفاده از متد ReadKey() برای دریافت یک کلید و توقف موقتی برنامه

کاربا رنگ‌ها : ConsoleColor جعبهٔ رنگ ۱۶تایی در C# است. نام این رنگ‌ها در جدول ۱ــ۲ آمده است.

۲۱ــ ویژگی BackgroundColor رنگ زمینه کنسول را نشان می‌دهد.

مثال : با استفاده از جعبه رنگ ConsoleColor رنگ زمینه را آبی کنید :

```
Console.BackgroundColor   ConsoleColor.Blue ;
```

با اجرای دستور بالا تغییری در رنگ زمینه مشاهده نمی‌شود. برای تغییر رنگ زمینه لازم است از متد Clear کلاس Console استفاده کنیم. دستور زیر را در خط بعدی بنویسید.

```
Console.Clear( ) ;
```

۲۲ــ ویژگی ForegroundColor رنگ قلم کنسول را نشان می‌دهد.

مثال : کلمه IRAN را با رنگ قرمز روی صفحه بنویسید :

```
Console.ForegroundColor   ConsoleColor.Red ;
Console.WriteLine("IRAN") ;
```

جدول ۱ــ۲ ــ جعبهٔ رنگ ConsoleColor

نمونه رنگ	نام رنگ	نام رنگ در ConsoleColor
Black	مشکی	Black
DarkBlue	سورمه ای	DarkBlue
DarkGreen	سبز تیره	DarkGreen
DarkCyan	فیروزه ای تیره	DarkCyan
DarkRed	قرمزتیره	DarkRed
DarkMagenta	بنفش	DarkMagenta
DarkYellow	زرد تیره	DarkYellow
DarkGray	خاکستری تیره	DarkGray
Blue	آبی	Blue
Green	سبز	Green
Cyan	فیروزه‌ای	Cyan
Red	قرمز	Red
Magenta	صورتی	Magenta
Yellow	زرد	Yellow
White	سفید	White
Gray	خاکستری	Gray

خودآزمایی فصل دوم

۱ــ قالب کلی یک برنامه به زبان C# چگونه است؟

۲ــ درهر برنامه به زبان C#، دست‌کم یک ..... تعریف می‌شود که در داخل آن، یک متد به نام ..... وجود دارد که اجرای برنامه از آن نقطه آغاز می‌گردد.

۳ــ در زبان C#، انتهای یک دستور با چه علامتی مشخص می‌شود؟

۴ــ تحقیق کنید که در زبان‌های C، C و Java علامت پایان دستور چیست؟

۵ــ علامت توضیحات در زبان C# کدام است؟

۶ــ روش قراردادی پاسکال برای نوشتن نام یک کلاس چیست؟

۷ــ برای نشان دادن یک پیام یا یک عبارت بر روی صفحه نمایش از چه دستوری استفاده می‌کنید؟

۸ــ تفاوت بین دو متد ()WriteLine و ()Write را بیان کنید.

۹ــ با استفاده از متد ()WriteLine، دستور مناسبی برای ایجاد یک سطر خالی بین نوشته‌ها در خروجی بنویسید.

۱۰ــ مثالی از کاربرد دستور using بیاورید.

۱۱ــ برای هر یک از خواسته‌های زیر، دستور یا دستورات مربوطه را بنویسید :

الف) نام شما در خروجی بارنگ زرد نمایش داده شود.

ب) صفحه کنسول نمایش را پاک کنید.

پ) اجرای برنامه تا فشردن یک کلید متوقف شود.

ت) صفحه کنسول نمایش با رنگ آبی پاک شود.

تمرینات برنامه‌نویسی فصل دوم

۱ــ برنامه‌ای بنویسید که نام، نام خانوادگی و نام مدرسه شما را روی صفحه نمایش دهد.

۲ــ برنامه تمرین ۱ را تغییری دهید تا اطراف نام شما یک کادر مانند شکل زیر رسم نماید مثلاً اگر اسم شما محمد است. روی صفحه چنین نمایش داده شود :

MOHAMMAD

۳ــ برنامه‌ای بنویسید که حرف انگلیسی نام شما را با استفاده از علامت * نشان دهد. مثلاً اگر اسم شما حمید است حرف انگلیسی H را با استفاده از علامت * به‌صورت زیر نشان دهد.

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

یادآوری : هنگامی‌که می‌خواهید برنامه‌ای بنویسید سعی کنید از برنامه‌ای که قبلاً ذخیره کرده‌اید استفاده کنید. برنامه را در یک ویرایشگر باز و سپس تغییرات لازم را ایجاد و ذخیره نمایید.

۴ــ برنامه‌ای بنویسید که تعداد روزهای عمر شما را با استفاده از ضرب سن شما در عدد ۳۶۵ محاسبه روی صفحه نشان دهد. مثلا اگر سن شما ۱۶ است عبارت ۱۶ × ۳۶۵ را محاسبه و نمایش دهد.

```
System.Console.WriteLine (16 * 365);
```

۵ــ با توجه به تمرین ۴ تعداد سال‌های کبیسه را نیز حساب کنید و برنامه را تغییر دهید.

(**راهنمایی :** برای محاسبه تعداد سال کبیسه، خارج قسمت سن بر عدد ۴ را حساب کنید.)

۶ــ با اجرای برنامه زیر چه عبارتی بر روی صفحه نشان داده می‌شود؟ به نظر شما چه ارتباطی بین علایم {۰} و {۱} و {۲} با اعداد ۱۸ و۱۵ و ۱۵ ۱۸ وجود دارد؟

```
Class SomeCalculations
{
Static void Main( )
{
  System.Console.WriteLine("{0}   {1}   {2}", 18 , 15 , 18  15);
}
}
```

۷ــ تمرین زیر به زبان انگلیسی است. آن را با دقت بخوانید و برنامه خواسته شده را بنویسید. برای زیبایی خروجی از جعبه رنگ ConsoleColor استفاده نمایید.

Write a program that prints a face, using text characters, hopefully better looking than this one.

```
  ///////
 | o o |
(|  ^  |)
 |  ‾  |
```

واژگان و اصطلاحات انگلیسی فصل دوم

ردیف	واژه انگلیسی	معنی واژه به فارسی
۱	C Sharp Compiler	
۲	Calculate	
۳	Calculation	
۴	Character	
۵	Class	
۶	Command Prompt	
۷	Comment	
۸	Common Language Runtime	
۹	Console	
۱	Display a greeting message	
۱۱	Expected	
۱۲	Main	
۱۳	Method	
۱۴	Namespace	
۱۵	Object Oriented Language	
۱۶	Open Source	
۱۷	Pascal case	
۱۸	Path	
۱۹	Press any key to exit	
۲	Reserved words	
۲۱	String	
۲۲	Text Editor	

فصل سوم

آشنایی با ویژوال استودیو

در فصل قبلی با ساختار کلی یک برنامه C# آشنا شدید و برنامه‌های ساده را با استفاده از یک ویرایشگر، نوشته و ذخیره کردیم. سپس فایل برنامه را از طریق پنجره فرمان[1] با استفاده از مترجم زبان C# (برنامه csc.exe) ترجمه کرده و فایل اجرایی EXE. تولید شد. در آخر، فایل EXE را اجرا کردیم. این روش برای نوشتن و تولید برنامه‌های کوچک، خوب است اما اگر برنامه‌ای که می‌نویسیم بزرگ باشد، به کارگیری این روش کمی‌دشوار و زمان بر خواهد بود. به خصوص عیب‌یابی و اشکال‌زدایی آن بسیار وقت گیر است. در این فصل با برنامه‌ای آشنا خواهیم شد که همه ابزارها و لوازم مورد نیاز یک برنامه نویس در آن گردآوری شده است و کار برنامه‌نویسی را آسان می‌کند.

پس از پایان این فصل انتظار می‌رود که فراگیر بتواند :

۱ــ IDE را تعریف کند و برای آن مثال بیاورد.

۲ــ مزایای استفاده از IDE و کاربرد VS را بیان نماید.

۳ــ یک برنامهٔ جدید در محیط VS از نوع کنسول ایجاد کند، آن را ذخیره، ترجمه و اجرا نماید.

۱ــ۳ــ آشنایی با ویژوال استودیو[2]

تایپ برنامه در یک ویرایشگر، ورود به پنجره فرمان، ترجمه کردن، عیب یابی و اشکال زدایی برنامه، همگی عملیاتی بودند که در فصل قبلی انجام دادید وکمی‌وقت گیر و پر زحمت بود، چون از یک محیط باید وارد محیط دیگری می‌شدید. برای این که راحت‌تر بتوانیم برنامه‌نویسی کنیم لازم است

۱ــ Command Prompt

۲ــ V sua Stud o

از محیطی استفاده کنیم که همه ابزارها و لوازم مورد نیاز برنامه‌نویسی در آن گردآوری و متمرکز شده باشد. به چنین محیط برنامه‌نویسی که در آن می‌توان تمام مراحل برنامه‌نویسی، ترجمه، اشکال یابی و سرانجام اجرا را انجام داد، IDE[1] گفته می‌شود که به معنای محیط تولید برنامهٔ متمرکز می‌باشد. یعنی همه ابزارها و امکانات لازم برای تولید برنامه در یک جا گردآوری شده است.

شرکت مایکروسافت یک IDE بسیار پیشرفته برای برنامه‌نویسی فراهم کرده است که با کمک آن می توان راحت‌تر برنامه بنویسیم و ترجمه و اجرا کنیم. نام این نرم افزار ویژوال استودیو است.

ویژوال استودیو یک محیط برنامه‌نویسی بسیار قوی برای تولید برنامه‌های کاربردی تحت ویندوز و بر پایه Net Framework. می‌باشد. ویژوال استودیو از چند زبان برنامه‌نویسی نظیر C#، C و VB پشتیبانی می‌کند. در این محیط علاوه بر تایپ برنامه، می‌توان برنامه را ترجمه، عیب‌یابی و سرانجام اجرا کرد. لایه نرم افزاری NET Framework 4.5. به همراه ویژوال استودیو ۲۰۱۲ عرضه شد است. در حال حاضر[2] VS 2013 عرضه شده است. خوشبختانه شرکت مایکروسافت همراه با عرضه ویژوال استودیوی تجاری، یک نسخه رایگان از این نرم افزار را تحت عنوان ویژوال استودیو اکسپرس[3] نیز عرضه می‌کند که می‌توانید آن را از روی سایت شرکت مایکروسافت[4] دانلود نمایید.

شکل ۱ــ۳ــ اجرای ویژوال استودیو اکسپرس ۲۰۱۲

---

۱ ــ Integrated Deve opment Env ronment

۲ ــ اردیبهشت ۹۳

۳ ــ V sua Stud o Express

۴ ــ http: //m crosoft.com/v sua stud o/down oads