برنامه‌نویسی برای نوجوانان:

اسمال بیسیک Small Basic

نوشته‌ی مایکروسافت

مترجم و معلّم : حدیث ملکی

ناشر: گراف ابریشم

www.silkgraph.ir

به نام خداوند بخشنده و مهربان

سرشناسه : ملکی، حدیث
عنوان و پدید آور : برنامه‌نویسی برای نوجوانان: اسمال بیسیک / مایکروسافت؛ برگردان حدیث ملکی.
مشخصات نشر : ایران: گراف ابریشم، ۱۳۹۷.
مشخصات ظاهری : ۸۵ص. نسخه دیجیتال
موضوع : کلیات ، دانش و کامپیوتر
شناسه افزوده : ملکی، حدیث، ۱۳۶۳– ، مترجم
رده بندی دیویی :

نشر : گراف ابریشم
http://silkgraph.ir hadis.design@gmail.com http://hmaleki.ir

برنامه‌نویسی برای نوجوانان: اسمال بیسیک
نویسنده: مایکروسافت
برگردان: حدیث ملکی
طراح جلد و صفحه‌آرا: حدیث ملکی
ناشر: گراف ابریشم
نوبت نشر اینترنتی : اوّل، زمستان ۹۷
قیمت نسخه ی ترجمه پارسی دیجیتال: ۱۱۰۰۰ تومان
خرید از فروشگاه اینترنتی گیلاگراف http://guilagraph.ir
دانلود رایگان بخش‌هایی از کتاب در وبسایت گراف ابریشم http://silkgraph.ir

---

اطلاعات بیشتر در silkgraph.ir
هماهنگی و خرید اجازه برای تکثیر و چاپ برای مدارس با ایمیل یا وب سایت

اگر از خواندن این کتاب لذّت می برید آن را به دوستان خود معرفی کنید.

نسخه ی انگلیسی این کتابچه به طور رایگان توسط
مایکروسافت در اختیار عموم قرار داده شده

اسمال بیسیک

Microsoft Small Basic

مقدّمه ای بر برنامه نویسی

فهرست منابع

پیشگفتار مترجم

سلام به همه‌ی خوانندگان عزیزی که دل توی دلتان نیست که هنرجوی هنرِ برنامه نویسی بشوید و یا بزرگ‌ترهایی که به دنبال پیدا کردن راه مناسبی برای آشنا کردن فرزندانتان، دوستانتان یا شاگردانتان با برنامه‌نویسی هستید.

یک سلام بلند و گرم و لطیف به تو خواننده عزیز و شریف [1] **به همراه خوش‌آمدگویی گرم.**

چرا برنامه‌نویسی؟ برنامه‌نویسی اگرچه که دومین پلّه از پلّه‌های مختلف دنیای مهندسی کامپیوتر هست، اما انقدر لذّت بخش است و انقدر فایده دارد که هرکسی حتی شده مدت کوتاهی باید با آن آشنا بشود. هرکس در هر شغلی در هر رشته‌ای حتماً با لحظه‌هایی مواجه می‌شود که به خودش می‌گوید: « حالا چه کار کنم؟ کدام را انتخاب کنم؟ چه طور مدیریت کنم؟ از بین این همه کار، کدام را اوّل انجام بدهد؟ چه طور با کمترین زمان یا کمترین هزینه بیشترین فایده را ببرم؟» و اینجاست که آن روزهایی که با دنیای الگوریتم سر وکار داشتید به کارتان می‌آید، آن روزها که به کامپیوتر می‌گفتید چه طور کاری را و به چه ترتیبی انجام بدهد، آن روزها به کارتان می‌آید. حتی اگر آنقدرها نیز سرتان شلوغ نباشد حداقل آن است که همه مشغول حلّ یک مسأله ایم که *از کجا آمده ام آمدنم بهر چه بود* [2].

و اما چرا اسمال بیسیک؟ چون اسمال بیسیک در واقع یک چراغ جادو است که وقتی به آن دست می‌کشیم غولی از آن بیرون می‌آید و به ما در برنامه‌نویسی کمک می‌کند. کاملاً جدی می‌گویم، از سال ۱۳۸۳ که تدریس را شروع کردم، با دانش‌آموزان با نرم‌افزارهای مختلفی کار کردیم و در زمینه‌ی برنامه‌نویسی، با زبان‌های مختلفی شروع کردیم. موقع برنامه‌نویسی هرکدام از زبان‌های برنامه‌نویسی، مزیت‌های خودشان را داشتند اما وقت یادگیری دردسرهایشان نیز مشخص می‌شد، بعضی محیط‌های برنامه‌نویسی، برای گروه سنی سال‌های آخر دبستان مناسب بود و برای دانش‌آموزان پایه ی دوم راهنمایی و هفتم به بالا بعد از چند جلسه کودکانه به نظر می‌رسید، برخی دیگر وقتی دانش آموز در آن‌ها تایپ می‌کرد، مشکل دیگری داشتند: با وجود این‌که دانش‌آموز می‌توانست راه حل را پیدا کند و برنامه‌نویسی را بلد بود اما اشتباهات تایپی پیش می‌آمد یا اینکه شیوه‌ی دقیق استفاده از دستور [3] را فراموش می‌کرد، و این باعث می‌شد که دقیقاً همان جا که با هیجان زیادی آماده بود تا نتیجه‌ی زحماتش را ببیند، کارش نتیجه نمی‌داد و با پیغام خطا روبرو می‌شد. یا

---

[1] بخشی از شعری از ابوالفضل زرویی نصرآباد (۱۳۹۷–۱۳۴۸)

[2] مولانا

[3] فراموش کردن نحو یا همان syntax به طور کاملاً طبیعی برای دانش‌آموزان خیلی پیش می‌آید خصوصاً که زبان انگلیسی زبان اوّل ما نیست.

حتی بدتر از آن خیلی سال‌های قبل، وقتی تازه محیط‌های جذاب ویندوز و سیستم عامل‌های جدید آمده بودند، شروع به یادگیری برنامه‌نویسی در آن محیط‌های ساده با پس‌زمینه‌ی آبی یا مشکی واقعاً کسالت آور بود.

امّا اسمال بیسیک با چند ویژگی وارد میدان شد تا علاقه‌مندان از برنامه‌نویسی و یادگیری آن لذّت ببرند.

- **اوّل اینکه محیطی جذاب، با پس زمینه‌ی مناسب دارد و کلمات را با رنگ‌های متنوّع می‌نویسد همین محیط رنگی، دل خیلی‌ها را می‌برَد.**

- **دوم اینکه در سمت راست، نوار ابزاری هست که وقتی دستوری نوشته می‌شود، شیوه‌ی درست استفاده از آن را در آن جا نمایش می‌دهد و دانش‌آموز به جای آنکه برای جستجو در جزوه‌هایش سردرگم شود یا در کتاب و اینترنت بگردد تا به یادش بیاید که مثلاً از پرانتز استفاده کند یا نه، خیلی راحت به نوار سمت راست نگاه می‌کند.**

- **و سوم غول چراغ جادو یا همان گردونه‌ی پیشنهادات[4]، کافی است برنامه نویس تایپ کند t و یا در جایی نقطه بگذارد و غول چراغ جادو یک گردونه از انواع پیشنهادات که با t شروع می‌شود یا می‌تواند بعد از نقطه بیاید را به او نشان می‌دهد و می‌گوید: « جانم! قربان امری داشتید؟ فرمودید t، خوب بفرمایید text می‌خواهید یا textwindow یا turtle یا timer ، امرتان کدام است؟»**

مطالب گفته شده را که روی هم بگذاریم، از این بهتر می‌شود؟ برای نصب اسمال بیسیک می‌توانید آن را به راحتی از وبسایت smallbasic.com دریافت و نصب کنید و یا حتی راحت‌تر از آن، بدون نصب برنامه در همان وب‌سایت در بخش start coding online شروع به برنامه‌نویسی کنید.

این کتاب مباحث ابتدایی و متوسط اسمال بیسیک را به خوبی توضیح می‌دهد. اگر فرصتی باشد امیدوارم در کتاب بعدی به شکل پروژه محور، به امکانات بیشتر و پیشرفته‌ی اسمال بیسیک، و معرفی افزونه‌های و امکانات گرافیکی برای تولید نرم‌افزارهای فارسی جذاب بپردازم.

-حدیث ملکی

دی ۱۳۹۷

[4] autocomplete

فصل اوّل

مقدّمه

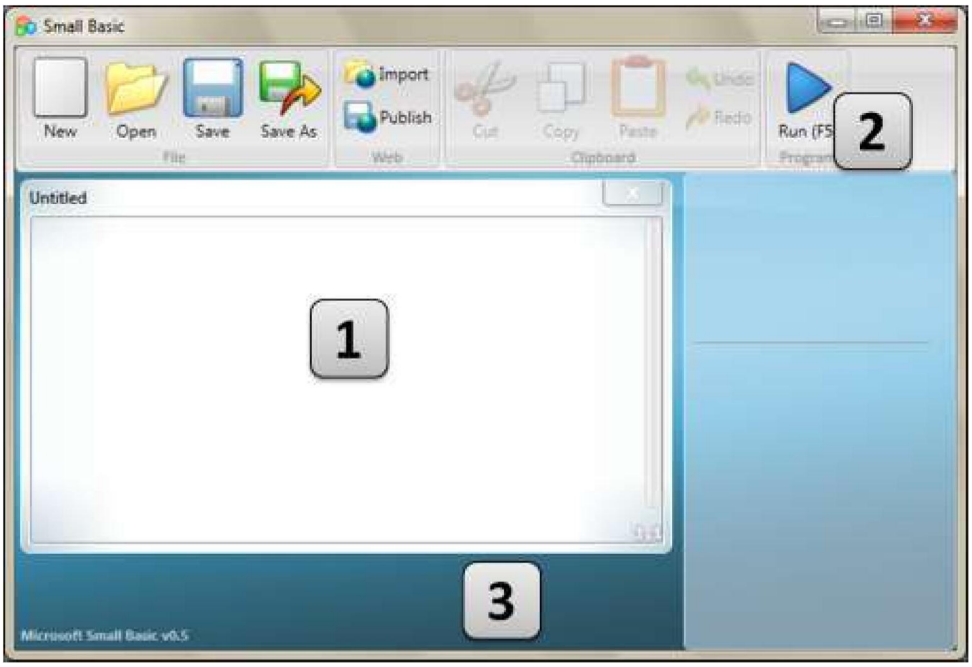
اِسمال بِیسیک و برنامه‌نویسی

منظور از برنامه نویسی کامپیوتری، نوشتن نرم‌افزار کامپیوتری با کمک یک زبان برنامه نویسی است. همان طور که ما انسان‌ها با زبان‌های فارسی، عربی، انگلیسی، اسپانیایی، فرانسوی و ... صحبت می‌کنیم تا منظور یکدیگر را متوجّه شویم، کامپیوترها نیز برنامه‌هایی را متوجّه می‌شوند که به زبان‌های مخصوص نوشته شده باشند. این زبان‌ها، زبان برنامه‌نویسی نام دارند. اوایل، تنها چند زبان برنامه‌نویسی وجود داشت که درک و یادگیری بیشتر آن‌ها ساده بود. اما همانطور که کامپیوترها و نرم‌افزارها پیچیده‌تر و پیچیده‌تر شدند، زبان‌های برنامه‌نویسی نیز به سرعت رشد کردند، و مفاهیم پیچیده‌ای را در برگرفتند. بنابراین زبان‌های برنامه‌نویسی مدرن و مفاهیم آن‌ها، برای افرادی که در ابتدای راه هستند بسیار پیچیده هستند. همین موضوع باعث شده تا برخی مردم از تلاش برای یادگیری برنامه‌نویسی کامپیوتر دلسرد شوند.

اسمال بیسیک یک زبان برنامه‌نویسی است و به شکلی طراحی شده تا برنامه‌نویسی را برای افراد تازه‌کار، بسیار ساده، قابل انجام و لذّت بخش کند. اسمال بیسیک قصد دارد تا حصارهای بلند ورود به دنیای برنامه‌نویسی را کوتاه کند و سکویی باشد تا شما با عبور از آن وارد دنیای شگفت‌انگیز برنامه‌نویسی بشوید.

محیط برنامه‌نویسی اسمال بیسیک

بیایید با آشنایی کوتاهی با محیط برنامه‌نویسی اسمال‌بیسیک شروع کنیم. وقتی اسمال بیسیک را باز می‌کنید، اوّلین موردی که خواهید دید پنجره‌ای شبیه شکل زیر است.



شکل۱- محیط برنامه‌نویسی اسمال بیسیک

این محیط برنامه‌نویسی اسمال بیسیک است، جایی که در آن برنامه‌های اسمال بیسیک را نوشته و اجرا می‌کنید. این محیط ویژگی‌هایی دارد که در شکل با اعداد مشخص شده‌اند.

شماره‌ی [۱] *ادیتور* یا *ویرایشگر* است که برنامه‌های اسمال بیسیک را در آن می‌نویسیم. وقتی شما شروع به نوشتن برنامه‌ای ساده می‌کنید و یا به تکمیل کردن برنامه‌ای می‌پردازید که از قبل ذخیره کرده اید، برنامه‌ی شما در این بخش نشان داده می‌شود. شما می‌توانید آن را تغییر دهید و یا ذخیره کنید تا بعداً روی تکمیل آن کار کنید.

همچنین شما می‌توانید بیشتر از یک برنامه را به طور هم‌زمان در اسمال بیسیک باز کرده و ببینید که هر برنامه در یک ویرایشگر جداگانه نمایش داده می‌شود. ویرایشگری که در حال حاضر مشغول نوشتن برنامه در آن هستید، *ویرایشگرِ فعال* نامیده میشود.

شماره‌ی [۲]، *نوار ابزار* است و در آن توضیحاتی درباره‌ی دستورات ویرایشگر یا موارد محیط برنامه‌نویسی نمایش داده می‌شود. در این کتاب، به مرور بیشتر درباره‌ی دستورات مختلف که در این نوار ابزار نمایش داده می‌شود، یاد خواهیم گرفت.

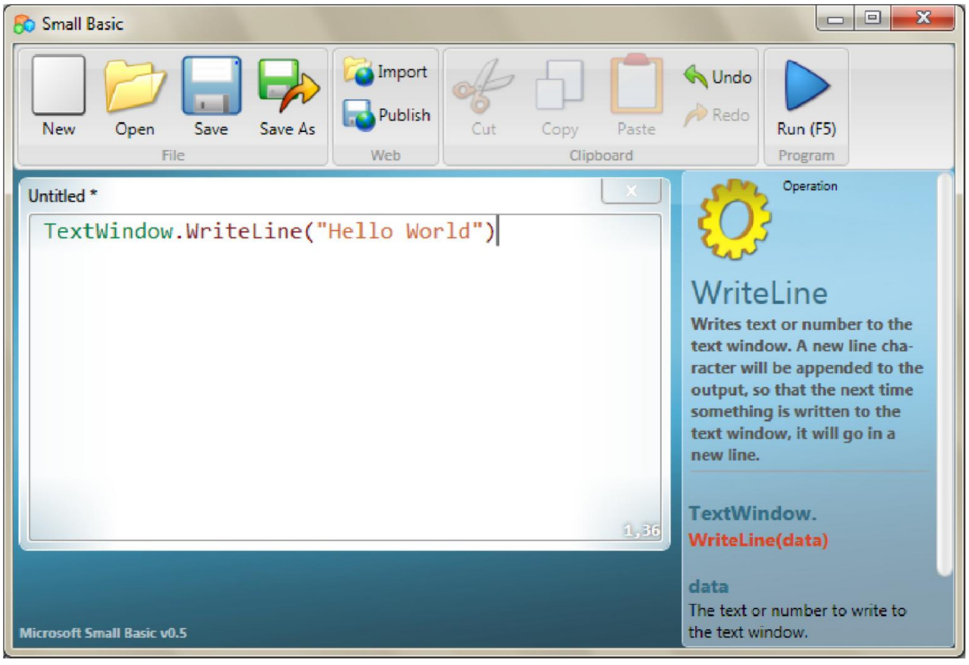
شماره‌ی[۳]، پوسته است که تمام ویرایشگرهای باز شده در آن قرار می‌گیرد.

اوّلین برنامه مان

حالا که با محیط اسمال بیسیک آشنا شدید، می‌خواهیم جلوتر برویم و شروع به برنامه‌نویسی کنیم. همان‌طور که پیش‌تر گفتیم، ویرایشگر جایی است که ما در آن برنامه‌مان را می‌نویسیم. بنابراین دست به کار شویم، این خط را در ویرایشگر بنویسید.

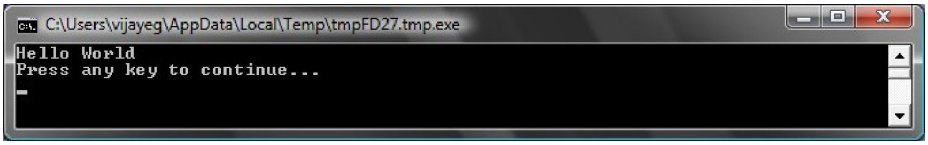
```
TextWindow.WriteLine("Hello World")
```

این اوّلین برنامه‌ی اسمال بیسیک ما است و اگر متن را درست تایپ کرده باشید آنچه می‌بینید باید شبیه شکل زیر باشد.



شکل ۲- اوّلین برنامه

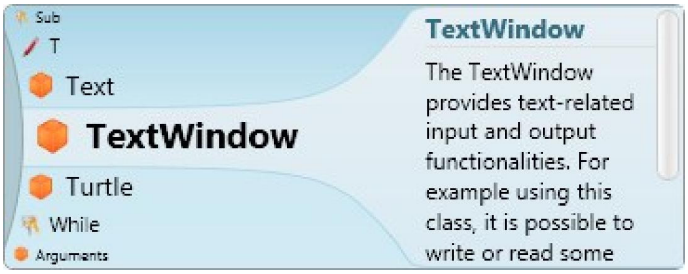
حالا که برنامه‌ی جدیدمان را نوشتیم، بیایید یک قدم جلوتر برویم، برنامه را اجرا کنیم و ببینیم چه اتفاقی رخ می‌دهد. برای اجرای برنامه می‌توانیم روی دکمه‌ی Run کلیک کنیم و یا از یک میانبر استفاده کنیم یعنی دکمه ی F5 را در صفحه کلید فشار دهیم. اگر همه چیز طبق روال پیش برود، نتیجه‌ی اجرای برنامه باید مشابه شکل زیر باشد.



شکل ۳– اوّلین اجرا گرفتن از برنامه‌مان

زنده باد! شما همین الان اوّلین برنامه‌ی امسال بیسیک را نوشتید و اجرا کردید. یک برنامه‌ی ساده و کوچک اما قدمی بزرگ به سمت تبدیل شدن به یک برنامه‌نویس واقعی! قبل از شروع به نوشتن برنامه‌های بزرگ‌تر باید کاری انجام دهیم و آن این‌که بررسی کنیم دقیقاً چه اتفاقی افتاد _ ما دقیقاً به کامپیوتر چه گفتیم و او چه طور فهمید که چه کار کند؟ در فصل بعد، ما به تجزیه و تحلیل اوّلین برنامه‌ای که نوشتیم می‌پردازیم، تا بتوانیم به درک بهتری دست پیدا کنیم.

درحالی که داشتید اوّلین برنامه را می‌نوشتید احتمالاً متوجّه نوشته‌ای به همراه فهرستی در کنار آن شُدید (شکل ۴). این «دستیار هوشمند» است و به شما کمک می‌کند تا برنامه‌تان را سریع‌تر تایپ کنید. شما می‌توانید با کمک دکمه‌های بالا و پایین در صفحه‌کلید موارد مختلف فهرست را ببینید و وقتی موردی را که دنبالش بودید پیدا کردید با زدن دکمه‌ی Enter آن دستور در متن برنامه‌ی شما نوشته می‌شود.



شکل ۴ – دستیار هوشمند

ذخیره کردن[5] برنامه‌مان

اگر بخواهید از اسمال بیسیک خارج شوید و بعداً در فرصتی دیگر برای نوشتن بقیه‌ی برنامه‌تان یا تغییر آن برگردید، نیاز هست تا برنامه را ذخیره کنید. جدای از آن، عادت کردن به ذخیره کردن مداوم برنامه، عادت خوبی است چرا که باعث می‌شود اگر ناگهان مشکلی در برق سیستم پیش آمد و یا سیستم به طور اتّفاقی خاموش شد، اطّلاعات برنامه‌های شما پاک نشود. برای ذخیره کردن برنامه می‌توانید دکمه‌ی Save را بزنید و یا از میانبر crtl+S در صفحه کلید استفاده کنید (درحالی که دکمه ی Ctrl را فشار داده اید، دکمه‌ی S را فشار دهید).

[5] save

فصل دوم

بررسی اوّلین برنامه‌مان

واقعاً برنامه‌ی کامپیوتری چیست؟

برنامه مجموعه‌ای از دستورات نوشته شده برای کامپیوتر است. این دستورات به کامپیوتر می‌گوید که دقیقاً چه کاری انجام دهد، و کامپیوتر همیشه این دستورات را اجرا می‌کند. کامپیوترها نیز مثل انسان‌ها می‌توانند دستورات را انجام دهند به شرط آن‌که دستورها به زبانی نوشته شده باشد که برای کامپیوتر قابل فهم باشد. به این‌ها زبان برنامه‌نویسی گفته می‌شود. زبان‌های قابل فهم برای کامپیوتر خیلی زیاد هستند و اسمال بیسیک یکی از آن‌هاست.

فرض کنید شما و دوستتان مشغول صحبت هستید. هر دوی شما برای انتقال اطلاعات از کلمات که آن ها را در قالب جملات درآورده‌اید استفاده می‌کنید. زبان‌های برنامه‌نویسی نیز به شکل مشابهی مجموعه‌ای از کلمات هستند که در قالب جملات چیده شده‌اند. هر برنامه مجموعه‌ای از جملات است (گاهی تنها چند جمله و گاهی حتی هزاران جمله) که این جمله‌ها در کنار یکدیگر برای برنامه‌نویس و برای کامپیوتر یک معنا پیدا می‌کنند.

برنامه‌های اسمال بیسیک

یک برنامه‌ی اسمال بیسیک از تعدادی *عبارت*[6] تشکیل می‌شود. هر خط برنامه یک عبارت را در خود دارد و هر عبارت دستورالعملی برای کامپیوتر است. وقتی از کامپیوتر می‌خواهیم که برنامه‌ی اسمال بیسیک را اجرا کند، برنامه را گرفته و اوّلین عبارت آن را می‌خواند و متوجّه می‌شود که ما می‌خواهیم به او چه بگوییم و دستورات ما را اجرا می‌کند. وقتی اجرای عبارت اوّل تمام شد، دوباره به برنامه برگشته و عبارت دوم را خوانده و اجرا می‌کند. آنقدر این کار را انجام می‌دهد تا به انتهای برنامه برسد. آن وقت است که برنامه تمام می‌شود.

[6] statement

سراغ برنامه‌ی اوّلمان برویم

این اوّلین برنامه‌ای بود که نوشتیم،

```
TextWindow.WriteLine("Hello World")
```
[7]

این برنامه‌ی خیلی ساده‌ای است که تنها از یک عبارت تشکیل شده است. این عبارت به کامپیوتر می‌گوید که در یک پنجره‌ی ساده‌ی مشکی یک خط نوشته یعنی Hello World را نمایش دهد.

انگار که در ذهنش به کامپیوتر می‌گوید

```
بنویس "Hello World"
```

احتمالاً تا الآن دقت کرده‌اید که همان‌طور که جملات می‌توانند به کلمات خرد شوند، عبارات برنامه نیز به بخش‌های کوچکتری تقسیم شده است. مثلاً در همین اوّلین عبارت ما سه بخش جدا داشتیم:

الف) TextWindow

ب) WriteLine

ج) "Hello World"

علامت‌های نگارشی مثل نقطه، پرانتز یا نقل قول[8] (")، مواردی هستند که باید در عبارت در جای درستی استفاده شود تا کامپیوتر منظور ما را متوجّه شود.

حتماً پنجره‌ی مشکی را که بعد از اجرای اوّلین برنامه‌مان ظاهر شد[9] یادتان می‌آید. به این پنجره Textwindow یا *پنجره‌ی متنی* یا گاهی حتی *کنسول* گفته می‌شود و جایی هست که نتیجه‌ی این برنامه در آن نمایش داده می‌شود.

---

[7] اگر دوست داشته باشید می‌توانید به جای Hello World بنویسید Salaam Bacheha

[8] گاهی برنامه‌نویسان در فارسی، معادل انگلیسی آن یعنی کوتِیشن را به کار می‌برند

[9] دوستان نوجوان من، مطالب این پاراگراف مربوط به معرفی اسامی مختلف در عبارات است، و به نظر من اگر تازه شروع به برنامه نویسی کرده اید به هیچ وجه نیازی به تمرکز دقیق یا حفظ کردن نیست، انقدر این نام‌ها را به کار خواهیم برد که کم کم با آن‌ها آشنا خواهید شد. – مترجم (حدیث ملکی http://hmaleki.ir)

در برنامه‌ای که نوشتیم در اصطلاحات برنامه نویسی به کلمه‌ی textwindow شیء[10] می‌گویند. در برنامه‌نویسی تعداد زیادی از شیء ها برای استفاده توسط برنامه‌نویسان وجود دارد. هر شیء مثل یک بسته است که در دل خود تعدادی *عملگر*[11] را دارد. و ما می‌توانیم از عملگرهای مختلف برای برنامه نویسی استفاده کنیم. مثلاً در برنامه‌ی ما از عملگر Writeline **به همراه متن** Hello World **داخل نقل قول استفاده کردیم. این متن به عنوان یک ورودی[12] برای عملگرِ** writeline **است، که در نهایت آن را بعد از اجرا در خروجی نشان می‌دهد. برخی عملگرها یک ورودی یا حتی بیشتر از یک ورودی می‌گیرند.**

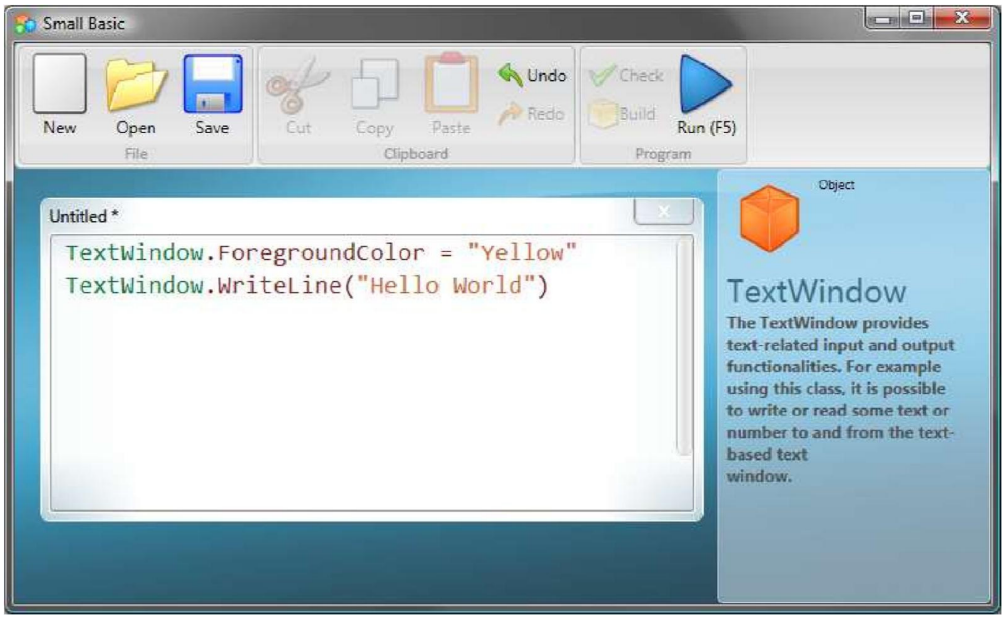
دومین برنامه‌مان

حال که اوّلین برنامه‌مان را نوشتیم و آن را متوجّه شدیم، به سراغ نوشتن یک برنامه‌ی جالب‌تر برویم و کمی رنگ به کارمان اضافه کنیم.

```
TextWindow.ForegroundColor = "Yellow"
TextWindow.WriteLine("Hello World")
```

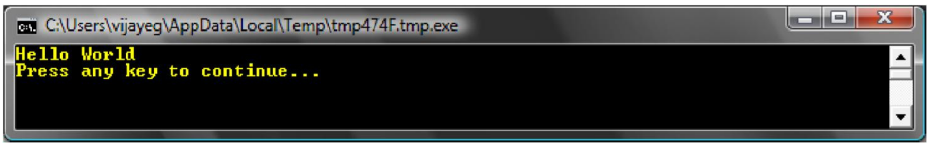
دقت کنید که علامت‌های نگارشی مثل نقل قول، فاصله و پرانتز در برنامه‌نویسی بسیار مهم هستند. مواردی مثل جای استفاده و تعداد استفاده از آن‌ها، می‌تواند معنی آن چه را که بیان می‌شود به کلی عوض کند.

[10] Object
[11] operation
[12] ورودی و خروجی دو اصطلاح رایج در کامپیوتر هستند، در یک مثال ساده یک دستگاه آب انارگیری را تصور کنید، ورودی آن، برش های انار و خروجی آن آب انار است. خروجی برنامه نیز آن چیزی است که برنامه بعد از اجرا، به ما می‌دهد. –مترجم (حدیث ملکی http://hmaleki.ir)



شکل ۵- اضافه کردن رنگ‌ها

وقتی برنامه‌ی بالا را اجرا می‌کنید، خواهید دید که در خروجی و در پنجره‌ی متنی مشکی، متن Hello World را نشان می‌دهد، اما این بار متن متفاوت با اجرای برنامه‌ی اوّل و به جای خاکستری، زرد رنگ است.



شکل ۶- سلام دنیا **Hellow World** به رنگ زرد

به عبارت جدیدی که در برنامه‌ی اصلی‌مان به کار بردیم دقت کنید. این برنامه از کلمه‌ی جدیدی یعنی ForegroundColor (رنگ متن) استفاده کرده و آن را برابر با Yellow (زرد) قرار داده است. این به این معنی است که ما رنگ متن را زرد کرده‌ایم. فرق بین دو عملگر Writeline و ForegroundColor در این است که ForegroudColor هیچ ورودی‌ای نمی‌گیرد[۱۳]. شما می توانید به جای Yellow از هرکدام از کلمات زیر استفاده کنید. به کاربردن علامت نقل قول را فراموش نکنید، استفاده از آن در ابتدای و انتهای رنگ لازم است.

[۱۳] یعنی جلوی آن نیاز به استفاده از پرانتز باز و بسته و قرار دادن ورودی بین آن ها نیست. – مترجم (حدیث ملکی http://hmaleki.ir)

Black	سیاه
Blue	آبی
Cyan	آبی فیروزه
Gray	سیاه
Green	سبز
Magenta	ارغوانی
Red	قرمز
White	سفید
Yellow	زرد
DarkBlue	آبی تیره
DarkCyan	آبی فیروزه‌ای تیره
DarkGray	خاکستری تیره
DarkGreen	سبز تیره
DarkMagenta	ارغوانی تیره
DarkRed	قرمز تیره
DarkYellow	زرد تیره

فصل سوم

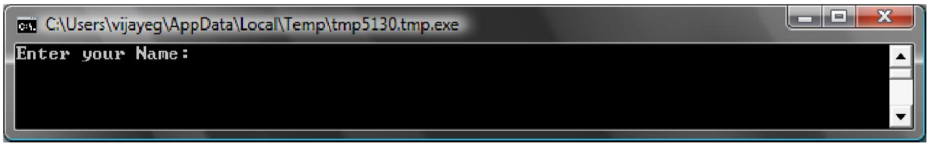
متغیّر چیست؟

استفاده از متغیّرها در برنامه‌مان

به نظرتان برنامه‌مان خیلی جذاب‌تر نخواهد شد اگر به جای آن که بگوید Hello World! اسم کسی را که برنامه را اجرا می‌کند[۱۴] بپرسد و اسمش را در یک جا نگه دارد تا بتواند بعد یک سلام ویژه مخصوص او و با به کار بردن اسم هر فرد در خروجی بنویسد؟ این، راهِ انجام این کار است:

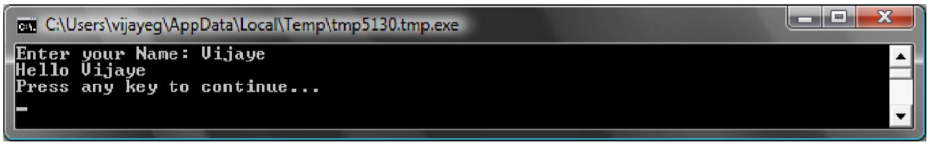
```
TextWindow.Write("Enter your Name: ")
name = TextWindow.Read()
TextWindow.WriteLine("Hello " + name)
```

وقتی این برنامه را اجرا می‌کنید چنین چیزی خواهید دید:



شکل ۷ – اسم کاربر را می‌پرسد

و وقتی شما اسم خودتان را تایپ می‌کنید و دکمه‌ی Enter را فشار می‌دهید:



شکل ۸ – سلامی گرم

حالا اگر بار دیگر برنامه را اجرا کنید و اسم دیگری را به کامپیوتر بدهید، کامپیوتر به اسم جدید سلام می‌کند.

[۱۴] در اصطلاحات کامپیوتر به کسی که با کامپیوتر کار می کند ، کاربَر یا user می‌گویند. –مترجم (حدیث ملکی http://hmaleki.ir)

بررسی برنامه

در برنامه‌ای که نوشتیم احتمالاً این خط توجه شما را به خود جلب کرده :

```
name = TextWindow.Read()
```

Read() همانند writeline() است، اما ورودی نمی‌گیرد. یک عملگر است که خیلی ساده به کامپیوتر می‌گوید، صبر کن تا کاربر چیزی تایپ کند و Enter را بزند. وقتی کاربر Enter را زد، این عملگر، آنچه را که کاربرد تایپ کرده می‌گیرد و به برنامه می‌دهد. مورد جالب این است که هرچه که کاربر تایپ کرده حالا در *متغیّری* به نام name نگه داشته شده است. متغیّر فضایی است که برای نگه‌داشتن موقت مقادیر و استفاده از آن‌ها در آینده تعریف می‌شود.

خطّ بعد هم جالب توجه است:

```
TextWindow.WriteLine("Hello " + name)
```

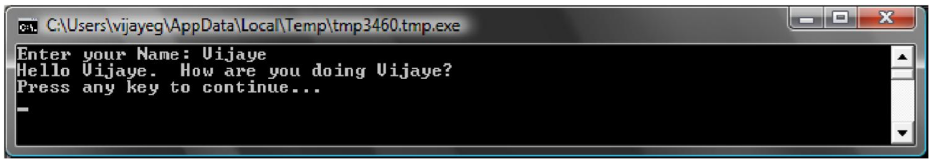
این جا خطی است که ما از مقدار نگه‌داشته شده در متغیّر، name، استفاده می‌کنیم. ما مقدار درون name را برداشته و کنار کلمه ی Hello می‌گذاریم و آن را در یک پنجره‌ی متنی نمایش می‌دهیم.[۱۵]

Write نیز همانند WriteLine عملگری دیگر برای پنجره‌ی کنسول است. با Write می‌توان متنی را در خروجی نشان داد تنها با این تفاوت که در آن متن بعدی در ادامه ی همین خط نوشته می‌شود و این‌طور نیست که از سر خطّ بعد نوشته شود.

```
TextWindow.Write("Enter your Name: ")
name = TextWindow.Read()
TextWindow.Write("Hello " + name + ".  ")
TextWindow.WriteLine("How are you doing " + name + "?")
```

[۱۵] یا به اصلاحِ رایج در کتاب‌های کامپیوتر: آن را در خروجی چاپ می‌کنیم –مترجم (حدیث ملکی http://hmaleki.ir)

سپس شما خروجی‌ای مانند شکل زیر خواهید دید:



شکل ۹– استفاده‌ی مجدد از متغییر

قوانین نام‌گذاری متغیّرها

متغیّرها نام‌گذاری می‌شوند و با کمک همین اسامی است که از هم تشخیص داده می‌شوند. قوانین ساده‌ای در مورد نام‌گذاری وجود دارند و برخی راهنمایی بسیار خوبی برای انتخاب اسم برای متغیّرها هستند.

۱ـ اسم متغیّر باید با حرف شروع شود و نباید هیچ کدام از کلمات دستورهای دیگر مثل if، for، then و ... باشد.

۲ـ نام متغّیر می‌تواند شامل ترکیبی از حروف، رقم‌ها و زیر خط[16] (_) باشد.

۳ـ بهتر است در نام‌های طولانی از _ بین کلمات استفاده کنیم و اسامی با معنا انتخاب کنیم.

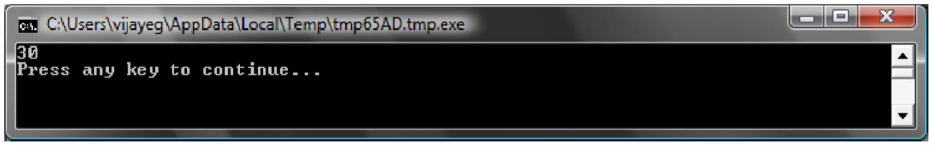
بازی با اعداد

تا این‌جا دیدیم که چه‌طور از متغیّر برای نگه‌داشتن اسم کاربر استفاده کنیم. در برنامه‌های بعدی، نحوه‌ی استفاده از متغیّر برای نگه‌داشتن اعداد و محاسبات روی آن را خواهیم دید. بیایید با یک برنامه‌ی واقعاً ساده شروع کنیم:

```
number1 = 10
number2 = 20
number3 = number1 + number2
TextWindow.WriteLine(number3)
```

[16] underscore

وقتی برنامه را اجرا می‌کنید خروجی زیر نشان داده می‌شود.



شکل ۱۰ – جمع کردن دو عدد

توجه کنید که اطراف اعداد نیازی به استفاده از علامت نقل قول نیست. علامت نقل قول " " تنها اطراف متن لازم است.

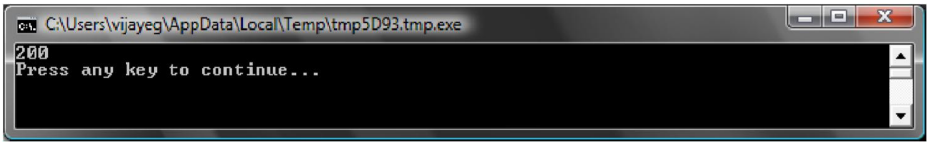
در اوّلین خط از برنامه، شما در متغیّری با نام number1 مقدار ۱۰ را نگه می‌دارید. در خط بعدی شما در متغیّری به نام number2 ، عدد ۲۰ را می‌ریزید. در خط سوم شما مقادیر درون متغیّرهای number1 و number2 را با هم جمع می‌زنید و نتیجه را در متغیّری به نام number3 می‌ریزید. بنابراین در این برنامه در متغیّر number3 مقدار ۳۰ وارد می‌شود. و این همان چیزی است که در خروجی متنی برنامه‌مان نشان داده می‌شود.

حالا بیایید برنامه را کمی تغییر دهیم و ببینیم نتیجه چه می‌شود:

```
number1 = 10
number2 = 20
number3 = number1 * number2 TextWindow.WriteLine(number3)
```

برنامه‌ی بالا مقدارهای درون number1 و number2 را در هم ضرب می‌کند[۱۷] و نتیجه را در number3 می‌ریزد.

شما خروجی آن را مشابه شکل زیر خواهید دید:



شکل ۱۱– ضرب کردن دو عدد

مشابه همین کار را می‌توانید برای تفریق استفاده کنید. خط زیر مربوط به تفریق است:

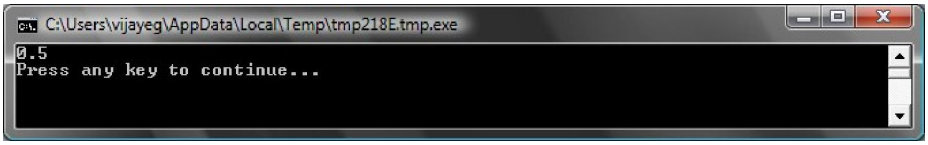
```
number3 = number1 - number2
```

[۱۷] در برنامه نویسی اسمال بیسیک از علامت ستاره (shift+ 8) برای ضرب استفاده کنید – مترجم (حدیث ملکی http://hmaleki.ir)

علامت تقسیم در برنامه نویسی اسمال بیسیک / است و دستورش مانند شکل زیر است:

```
number3 = number1 / number2
```

و نتیجه در خروجی به شکل زیر خواهد بود:



شکل ۱۲- تقسیم دو عدد

برنامه‌ی تبدیل دما

در برنامه بعدی می‌خواهیم دمای فارنهایت را به سانتی‌گراد تبدیل کنید و برای این‌کار از فرمول آن در فیزیک یعنی $℃ = \frac{5(°F-32)}{9}$ **استفاده می‌کنیم.**

ابتدا دمای فارنهایت را از کاربر می‌پرسیم و آن را در متغیّری نگه می‌داریم. TextWindow.ReadNumber عملگری است که از آن برای گرفتن عدد از کاربر استفاده می‌شود.

```
TextWindow.Write("Enter temperature in Fahrenheit: ")
fahr = TextWindow.ReadNumber()
```

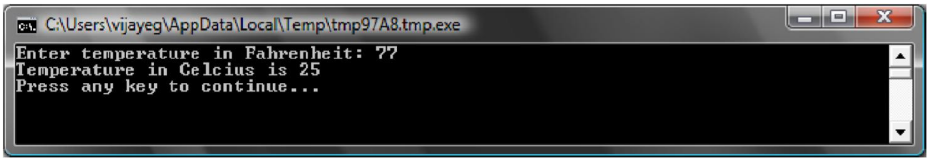
به محض این‌که دما را در واحد فارنهایت از کاربر گرفتیم، می‌توانیم با کمک فرمول زیر آن را به سانتیگراد (سلسیوس) تبدیل کنیم.

```
celsius = 5 * (fahr - 32) / 9
```

استفاده از پرانتز به کامپیوتر می‌گوید که اوّل تفریقِ fahr-32 را انجام دهد و بعد از آن سراغ سایر عملیات ضرب و تقسیم برود. روی هم رفته برنامه‌ی نهایی ما به این شکل خواهد بود:

```
TextWindow.Write("Enter temperature in Fahrenheit: ")
fahr = TextWindow.ReadNumber()
celsius = 5 * (fahr - 32) / 9
TextWindow.WriteLine("Temperature in Celsius is " + celsius)
```

و خروجی برنامه به شکل زیر خواهد بود:



شکل ۱۳ – تبدیل دما

فصل چهارم

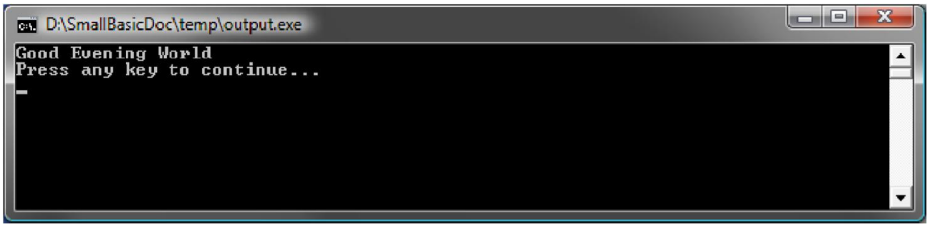
دستورهای شرطی و پرش ها

برنامه‌ی اوّلی که نوشتیم را در نظر بگیرید، به نظر شما خیلی جالب‌تر نخواهد شد اگر به جای اینکه بگوید HelloWorld! با توجه به ساعت، صبح به خیر، یا عصر به خیر بگوید؟ در برنامه‌ی بعدی می‌خواهیم کاری کنیم تا کامپیوتر اگر ساعت قبل از ۱۲ است بگوید Good Morning World و اگر ساعت از ۱۲ گذشته، بگوید Good Evening World

```
If (Clock.Hour < 12) Then
  TextWindow.WriteLine("Good Morning World")
EndIf
If (Clock.Hour >= 12) Then
  TextWindow.WriteLine("Good Evening World")
EndIf
```

براساس اینکه چه زمانی برنامه را اجرا کنید یکی از دو خروجی زیر را خواهید دید

شکل ۱۴- صبح به خیر دنیا!



شکل ۱۵- عصر به خیر دنیا!

بیایید سه خط اوّل برنامه را بررسی کنیم. احتمالاً متوجّه شده‌اید که این خط به کامپیوتر می گوید که اگر Clock.Hour کمتر از ۱۲ بود، در خروجی بنویسید Good Morning World. کلمات If، Then و EndIf کلمات خاصی هستند که موقع اجرا برای کامپیوتر معنی دارند. If به معنی «اگر» است و بعد از آن یک شرط نوشته می‌شود که در اینجا شرط (Clock.Hour < 12) است، دقت کنید که استفاده از پرانتز در اطراف شرط لازم است بدون آن‌ها کامپیوتر متوجّه منظور ما نمی‌شود. بعد از شرط باید از کلمه‌ی **then** به معنی «آنگاه» استفاده کنیم و بعد از آن بگوییم دوست داریم اگر آن شرط برقرار بود آنگاه کامپیوتر چه کاری را انجام دهد. سپس، وقتی کار مورد نظر را نیز نوشتیم باید از کلمه‌ی **Endif** استفاده کنیم. این به کامپیوتر می گوید که دستور شرطی ما تمام شد.

در اسمال بیسیک، شما می‌توانید از Clock برای داشتن زمان و تاریخ استفاده کنید. Clock یک شیء است که تعداد زیادی عملگر را برای استفاده‌ی شما همراه دارد، می توانید با کمک آن روز، ماه، سال، ساعت، دقیقه و ثانیه را در برنامه‌هایتان به دست آورید.

می‌توانید بین **then** و **Endif** از بیش از یک عملگر استفاده کنید و اگر شرط برقرار باشد کامپیوتر همه آن‌ها را برای شما انجام می‌دهد.

```
If (Clock.Hour < 12) Then
  TextWindow.Write("Good Morning. ")
  TextWindow.WriteLine("How was breakfast?")
EndIf
```

Else

شاید دقت کرده باشید در برنامه‌ای که ابتدای این فصل نوشتیم، شرط دوم به نوعی اضافه بود، وقتی ساعت قبل از ۱۲ نباشد، بعد از ۱۲ خواهد بود، چه نیازی به بیان آن است. ساعت یا قبل از ۱۲ هست و یا نیست و نیازی نیست که برای حالت دوم دقیق بگوییم که ساعت بعد از ۱۲ باشد. در چنین مواردی می‌توانید if . . then . . endif دوم را با یک کلمه کوتاه تر بنویسیم: else که در فارسی به معنی «وگرنه» است.

چنانچه برنامه را مجدداً اما این بار با دستور «وگرنه» else بنویسیم مانند زیر خواهد شد:

```
If (Clock.Hour < 12) Then
  TextWindow.WriteLine("Good Morning World")
 Else
  TextWindow.WriteLine("Good Evening World")
EndIf
```

این برنامه نیز مانند برنامه‌های قبل درس مهمی در مورد برنامه‌نویسی برایمان به همراه دارد:

> **در برنامه‌نویسی معمولاً بیش‌تر از یک راه برای انجام یک کار هست. گاهی یک راه سرراست‌تر است. انتخاب راه بر عهده‌ی برنامه‌نویس است. به مرور که بیشتر برنامه بنویسید و تجربه کسب کنید متوجّه تفاوت‌ها و مزیت‌ها یا نقاط ضعف هر راه می‌شوید.**

جلوبردن خطوط

در همه‌ی مثال‌ها دیدید که عبارت‌های بین if، else و Endif کمی جلوتر از سرخط نوشته شده‌اند و اصطلاحاً دارای تورفتگی هستند. اگر این کار را انجام ندهیم کامپیوتر همچنان برنامه را متوجّه می‌شود و انجام می‌دهد. اما، این طور مرتب نوشتن باعث می‌شود تا خود ما برنامه‌نویس‌ها خیلی راحت‌تر با یک نگاه به برنامه متوجّه ساختار آن شویم. بنابراین عادت به ایجاد تورفتگی برای دستورات بین این کلمات از عادت‌های خوب برنامه‌نویسی است.

زوج یا فرد

حالا که دستور If..Then..Else..EndIf را در خورجین داریم، بیایید یک برنامه بنویسیم که یک عدد بپرسد و بعد بگوید عددی که کاربر به او داده زوج (Even) است یا فرد(Odd).

```
TextWindow.Write("Enter a number: ")
num = TextWindow.ReadNumber()
remainder = Math.Remainder(num, 2)
If (remainder = 0) Then
  TextWindow.WriteLine("The number is Even")
Else
  TextWindow.WriteLine("The number is Odd")
EndIf
```

خروجی برنامه به این شکل خواهد بود:

شکل ۱۶- زوج یا فرد

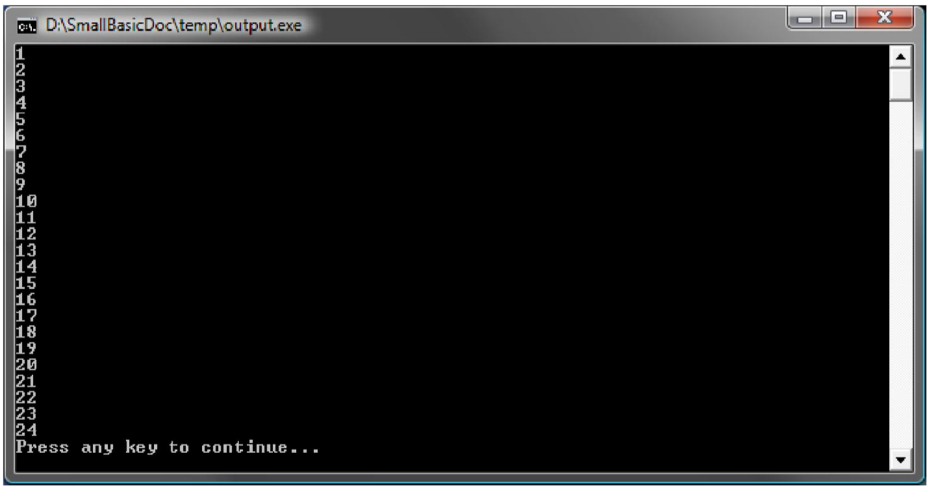
در این برنامه، از یک عملگر جدید استفاده کردیم، Math.Remainder. این عملگر عدد اوّل را بر عدد دوم تقسیم می‌کند و باقیمانده را می‌دهد.

پرش‌ها

در فصل دوم دیدیم که کامپیوتر در یک برنامه دستورها را یک به یک و به نوبت از بالا به پایین اجرا می‌کند. با این‌حال، عبارت مخصوصی هست که با کمک آن کامپیوتر می‌تواند از روی چند دستور بپرد و بدون اجرا آن ها به سراغ خط دیگری برود[۱۸]. بیایید به برنامه‌ی بعدی نگاه کنیم:

```
i = 1
start:
TextWindow.WriteLine(i)
i = i + 1
If (i < 25) Then
  Goto start
EndIf
```

[۱۸] دوستان عزیز، من معمولاً این شیوه از پرش را که شیوه‌ای قدیمی است تدریس نمی کنم، استفاده از پرش‌ها در برنامه نویسی اگر به درستی انجام نشود می‌تواند باعث مساله ای به نام «برنامه نویسی اسپاگتی» شود یعنی برنامه پر از پرش، درهم و برهم و گنگ شود. توصیه‌ی من این هست که در شرایطی که تازه شروع به برنامه نویسی کرده اید از این دستور استفاده نکنید، در دهه‌ی اخیر برنامه نویسان حرفه‌ای به جای استفاده از این دستور هر پرشی را که بخواهند با ساختارهای if.. else... و یا دستورهای حلقه که خواهیم دید، ایجاد می‌کنند. -مترجم (حدیث ملکی http://hmaleki.ir)



شکل ۱۷ – استفاده از **Goto**

در برنامه‌ی بالا، ما به متغیّر i، مقدار ۱ را دادیم و سپس عبارت دیگر را که با عبارت دو نقطه : خاتمه یافته است اضافه کردیم.

```
start:
```

به این کار *برچسب‌گذاری* گفته می‌شود. برچسب‌ها نوعی نشان هستند که کامپیوتر آن‌ها را متوجه می‌شود. شما می‌توانید نام برچسب و تعداد آن ها را هرچه می‌خواهید انتخاب کنید فقط نباید از یک اسم برای دو برچسب استفاده کنید.

عبارت جالب دیگر این است:

```
i = i + 1
```

این دستور به کامپیوتر می‌گوید که به آن چه که درون متغیّر i است یک عدد اضافه کند و سپس مقدار جدید را درون i نگه دارد. می‌بینید در این عبارت، کامپیوتر اوّل محاسبات سمت راست تساوی را انجام میکند و بعد به سراغ سمت چپ می‌رود.[۱۹]

[۱۹] هنرجویان عزیزِ هنر برنامه نویسی، در مورد علامتِ = در اسمال بیسیک، اگر آن را در یک خط جداگانه دیدید، آن را به شکل تساوی ریاضی نخوانید بلکه به شکل یک فلش ← که مقدار راست را در سمت چپ می‌ریزد تصور کنید، اما اگر این علامت را جلوی دستورهای شرطی if و یا جلوی حلقه‌های تکرار (فصل‌های بعدی) دیدید، آن را به همان معنای برابری در ریاضیات بخوانید. –مترجم (حدیث ملکی http://hmaleki.ir)

و در نهایت:

```
If (i < 25) Then
  Goto start
EndIf
```

این بخش به کامپیوتر می‌گوید که اگر مقدار درون i از ۲۵ کمتر بود، اجرای برنامه را از برچسب start شروع کند.

اجرای بی‌پایان

با کـمک دستور Goto شما می‌توانید کاری کنید که کامپیوتر بخشی از برنامه را چندین بار اجرا کند، برای مثال شما می‌توانید برنامه‌ی زوج یا فرد را به شکل زیر تغییر دهید، این برنامه تا ابد اجرا می‌شود. شما برای پایان دادن بـرنامه می‌توانید از دکمه‌ی بستن (ضربدر قرمز) در گوشه بالا سمت راست پنجره‌ی خروجی استفاده کنید.

```
begin:
TextWindow.Write("Enter a number: ")
num = TextWindow.ReadNumber()
remainder = Math.Remainder(num, 2)
If (remainder = 0) Then
  TextWindow.WriteLine("The number is Even")
Else
  TextWindow.WriteLine("The number is Odd")
EndIf
Goto begin
```

شکل ۱۸– اجرای بی‌پایان از تشخیص زوج یا فرد بودن

سرفصل‌های بعدی در بخش فهرست مطالب آورده شده است.

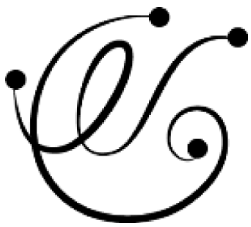
اگر تمایل به ادامه‌ی مطالعه و یادگیری برنامه نویسی اسمال بیسیک دارید،

برای دریافت نسخه‌ی کامل کتاب

می‌توانید به وب سایت گیلاگراف مراجعه کرده

و به راحتی بعد از انجام عملیات، فایل کامل کتاب را دریافت بفرمایید.

http://guilagraph.ir




www.silkgraph.ir