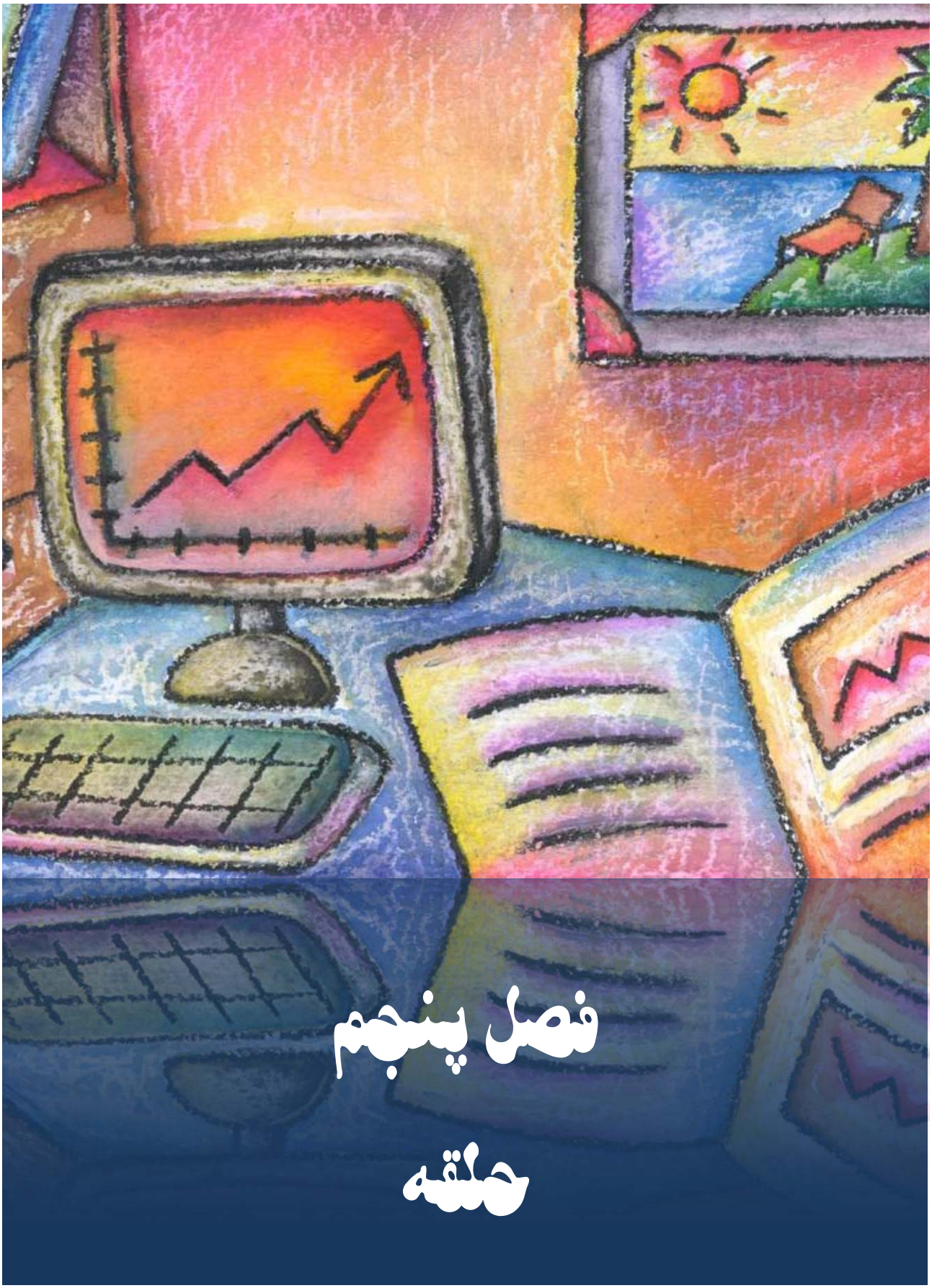

فصل پنجم

حلقه

۵. ۱. مقدمه

یکی از دلایل استفاده از رایانه‌ها، انجام اعمال تکراری با دقّت و سرعت بالا است. بسیار پیش می‌آید که ما برنامه‌ای برای انجام کارهای تکراری نیاز داشته باشیم. مثلاً فرض کنید می‌خواهیم برنامه‌ای بنویسیم که تمام اعداد اول کوچکتر از ۱۰۰۰ را نمایش دهد. باید این اعداد را بررسی کنیم و هر کدام را که اول هستند نمایش دهیم. پس باید عمل بررسی اول یا مرکب بودن اعداد را ۱۰۰۰ بار تکرار کنیم. در زبان بیسیک برای انجام کارهای تکراری از حلقه‌ها استفاده می‌کنیم. در این فصل دو نمونه از حلقه‌های پر کاربرد زبان بیسیک و طرز ساخت آن‌ها را یاد می‌گیریم.

۵. ۲. حلقه‌ی FOR

می‌خواهیم برنامه‌ای بنویسیم که اعداد ۱ تا ۱۰۰۰ را نمایش دهد. با توجه به دستوراتی که در فصل‌های گذشته خوانده‌ایم، می‌توانیم یک متغیر به اسم A انتخاب کنیم و عدد ۱ را در آن بگذاریم. سپس آن را چاپ کرده و یکی به مقدارش اضافه کنیم و این کار تکراری را ۱۰۰۰ بار انجام دهیم. با این کار به برنامه زیر می‌رسیم:

```
LET  A = 1
PRINT  A
LET  A = A + 1
IF  A <= 1000  THEN  GOTO  10
```

به خط سوم برنامه بالا دقّت کنید. این خط به محتوای خانه حافظه A یک واحد اضافه می‌کند. هم چنین به علامت => در خط چهارم دقّت کنید. اگر به جای آن از علامت > استفاده کنیم، عدد 1000 نمایش داده نمی‌شود.

توجه: برای افزودن X واحد به محتوای خانه حافظه A می‌توانیم از این دستور استفاده کنیم:

```
LET A = A + X
```

چون تکرار یک کار در زبان بیسیک زیاد مورد استفاده دارد، در زبان بیسیک دستور خاصی هم برای تکرار داریم. این دستور به شکل زیر است:

```
FOR نام متغیر = مقدار اولیه TO مقدار نهایی
        دستورات
NEXT نام متغیر
```

رایانه به این شکل با سه خط بالا رفتار می‌کند: ابتدا مقدار اولیه را درون متغیر ذخیره می‌کند. سپس دستوراتی که بین خط FOR و خط NEXT نوشته شده را انجام می‌دهد. بعد به خط NEXT می‌رسد. کار این خط این است که یک واحد به محتوای متغیر مورد استفاده اضافه می‌کند. سپس رایانه بررسی می‌کند که آیا محتوای فعلی متغیر از مقدار نهایی‌اش که در خط FOR نوشته‌ایم، بیش‌تر شده یا نه. اگر بیش‌تر نبود، دوباره تمام دستورات بین FOR و NEXT را انجام می‌دهد. بعد دوباره به خط NEXT می‌رسد و یکی به محتوای متغیر اضافه می‌کند. سپس بررسی می‌کند که آیا محتوای متغیر از مقدار نهایی بیش‌تر شده یا نه. اگر بیش‌تر نشده بود، دوباره این کارها را انجام می‌دهد و آن قدر این رویه تکرار می‌شود تا در خط NEXT، محتوای متغیر از مقدار نهایی که برایش مشخص کرده‌ایم، بیش‌تر شود. هر موقع چنین اتفاقی افتاد دیگر رایانه به خط‌های بالای NEXT برنمی‌گردد و به سراغ خط بعد از NEXT می‌رود و ادامه‌ی برنامه را اجرا می‌کند.

نمونه : برنامه‌ای بنویسید که طول و عرض یک مستطیل را گرفته، محیط و مساحت آن را نمایش دهد و این کار را ۱۰۰ بار تکرار کند.

پاسخ :

```
FOR  A = 1  TO  100
  INPUT  X , Y
  PRINT  (X +Y) * 2 , X * Y
NEXT A
```

مقدار اولیه متغیر A برابر 1 و مقدار نهایی‌اش برابر 100 است. پس دستورات بین FOR و NEXT، ۱۰۰ بار تکرار می‌شوند.

نمونه : برنامه‌ای بنویسید که اعداد ۱ تا ۱۰۰۰ را نمایش می‌دهد.

پاسخ :

```
FOR  A = 1  TO  1000
  PRINT  A
NEXT A
```

سعی کنید با توجه به توضیحاتی که در مورد روش کار دستور FOR گفتیم، این برنامه را خط به خط دنبال و اجرا کنید.

نمونه : برنامه‌ای بنویسید که اعداد زیر را بنویسد:

$1^2, 2^2, 3^2, \dots, 20^2$

پاسخ :

```
FOR  A = 1  TO  20
  PRINT  A ^ 2
NEXT  A
```

نمونه : برنامه‌ای بنویسید که یک عدد گرفته، تمام مقسوم علیه‌های آن را بنویسد.

پاسخ :

```
INPUT  A
FOR  X = 1  TO  A
  IF  A MOD X = 0  THEN  PRINT  X
NEXT  X
```

در این برنامه از عدد ۱ تا عدد گرفته شده (یعنی A) را بررسی می‌کنیم. عدد گرفته شده بر هر کدام از این اعداد بخش‌پذیر بود، آن عدد مقسوم علیه‌اش است. برای بخش‌پذیر بودن یک عدد بر یک عدد دیگر کافی است باقی‌مانده‌ی تقسیم اولی بر دومی صفر شود.

نمونه : برنامه‌ای بنویسید که یک عدد گرفته، تعداد مقسوم علیه‌های آن را بنویسد.

پاسخ : از همان ابتدای برنامه مثال قبل استفاده می‌کنیم. تنها فرق برنامه جدید با قبلی این است که به جای نمایش مقسوم علیه‌ها، باید تعدادشان را بشماریم. برای شمردن تعداد مقسوم علیه‌ها از این روش استفاده می‌کنیم: یک متغیر (مثلاً به اسم C) انتخاب می‌کنیم. هر وقت یک مقسوم علیه از عددمان پیدا کردیم، یک واحد به محتوای C اضافه می‌کنیم. در این صورت کار متغیر C می‌شود شمردن تعداد مقسوم علیه‌ها. به این روش، تکنیک شمارش می‌گویند و معمولاً برای شمردن تعداد یک چیز از آن استفاده می‌کنیم. پس برنامه به صورت زیر در می‌آید:

```
INPUT A
FOR X = 1 TO A
  IF A MOD X = 0 THEN LET C = C + 1
NEXT X
PRINT C
```

توجه کنید که بعد از خاتمه‌ی کار حلقه و شمارش کل مقسوم علیه‌ها، C را نمایش داده‌ایم. (در خط آخر)

نمونه : برنامه‌ای بنویسید که ۱۰۰ عدد گرفته، در پایان حاصل جمع تمام آن‌ها را بنویسد.

پاسخ : برای حل این مثال، یک خانه حافظه برای ذخیره حاصل جمع (مثلاً به اسم Bank) انتخاب می‌کنیم. هر کدام از ۱۰۰ عدد را که گرفتیم، آن را به محتوای این خانه حافظه اضافه می‌کنیم. در مثال‌های قبلی دیدیم که اگر بخواهیم x واحد به محتوای خانه حافظه Bank اضافه کنیم، کافی است بنویسیم:

```
LET Bank = Bank + x
```

برنامه مورد نظر به صورت زیر در می‌آید:

```
FOR A = 1 TO 100
  INPUT x
  LET Bank = Bank + x
NEXT A
PRINT Bank
```

نکته عملی :
اگر بخواهید درستی برنامه نوشته شده در بالا را روی رایانه امتحان کنید، به مشکل بر می‌خورید. چون وارد کردن ۱۰۰ عدد و بررسی صحت حاصل جمع آن‌ها کار سختی است. برای امتحان برنامه بالا بهترین کار این است که در خط اول به جای ۱۰۰، مثلاً ۵ بنویسید و برای ۵ عدد برنامه را امتحان کنید.

. . . . . .

۵. ۳. گام حلقه

یک نکته دیگر در مورد دستور FOR باقی می‌ماند و آن این است که فرض کنید ما بخواهیم در خط NEXT به جای این که هر دفعه ۱ واحد به محتوای متغیرمان اضافه شود، یک مقدار دیگر (مثلاً ۲ واحد یا ۳ واحد یا ۱.۵ واحد یا حتی ۲- واحد) اضافه شود. در زبان بیسیک قابلیت این کار پیش بینی شده است. برای عوض کردن گام حلقه به عددی غیر از ۱، کافی است جلوی خط FOR، کلمه‌ی STEP و سپس گام مورد نظر را بنویسید.

نمونه : برنامه‌ای بنویسید که اعداد زوج ۲ تا ۱۰۰ را بنویسد.

پاسخ :

```
FOR A = 2 TO 100 STEP 2
  PRINT A
NEXT A
```

مشاهده می‌کنید که برای ساختن اعداد زوج و نمایش آن‌ها، از یک حلقه با گام ۲ استفاده کرده‌ایم.

نمونه : برنامه‌ای بنویسید که اعداد زیر را به ترتیب از سمت چپ بنویسد:

۳۰۰ , ۲۹۷ , ۲۹۴ , ... , ۳ , ۰

پاسخ :

```
FOR A = 300 TO 0 STEP -3
  PRINT A
NEXT A
```

توجه:

۱. موقعی که گام حلقه منفی باشد، زمانی کار حلقه خاتمه می‌یابد که مقدار متغیر از مقدار نهایی که برای آن مشخص کرده‌ایم، کمتر شود.

۲. گام حلقه، مقدار نهایی و مقدار اولیه می‌توانند هر عدد مثبت یا منفی یا اعشاری باشند.

۳. اگر در حلقه‌ای با گام مثبت (گام منفی) مقدار اولیه از مقدار نهایی بزرگ‌تر (کوچکتر) باشد، دستورات داخل حلقه اصلاً اجرا نمی‌شوند و رایانه به سراغ دستورات بعد از خط NEXT می‌رود.

۴. اگر مقدار اولیه و نهایی یک حلقه با هم مساوی باشند، دستورات داخل حلقه فقط یکبار اجرا می‌شوند.

نمونه : برنامه زیر چه کار می‌کند؟

```
FOR A = 1 TO 5 STEP 0
  PRINT A
NEXT A
```

پاسخ : چون گام حلقه صفر است، هر بار در خط NEXT، صفر واحد به محتوای A اضافه می‌شود، یعنی محتوای آن در خط NEXT اصلاً تغییری نمی‌کند. پس مقدار آن هم هیچ‌گاه از مقدار نهایی بیش‌تر نمی‌شود. پس این حلقه اصلاً تمام نمی‌شود، یعنی یک حلقه پایان ناپذیر داریم که بی‌نهایت 1 می-نویسد.

نکته :

اگر خواستید در حین اجرای برنامه، آن را متوقف کنید، کلیدهای Ctrl و Break را با هم فشار دهید.
در برنامه بالا اگر این کلیدها را بگیرید، تا موقعی که برق نرود، رایانه به نوشتن عدد 1 ادامه می‌دهد!

نمونه : برنامه‌ای بنویسید که ۱۰۰ عدد گرفته، بزرگ‌ترین آن‌ها را بنویسد.

پاسخ :روش کار بدین صورت است که یک خانه حافظه (مثلاً به اسم Max) برای ذخیره بزرگ‌ترین عدد انتخاب می‌کنیم. ابتدا اولین عدد را در آن قرار می‌دهیم. سپس اعداد بعدی را دانه دانه با محتوای این خانه حافظه مقایسه می‌کنیم. هر وقت یک عدد بزرگ‌تر پیدا کردیم، همان را درون خانه حافظه Max ذخیره می‌کنیم. با انجام این مقایسه برای هر ۱۰۰ عدد، نهایتاً بزرگ‌ترین عدد از بین اعداد گرفته شده درون خانه حافظه Max ذخیره شده است که آن را نمایش می‌دهیم.

پس برنامه به این شکل در می‌آید:

```
INPUT  A
LET  Max = A
FOR  X = 1  TO  99
  INPUT  A
  IF  A > Max  THEN  Max = A
NEXT  X
PRINT  Max
```

. . . . . .

تمرین‌های برنامه نویسی

۱. برنامه‌ای بنویسید که شعاع ۱۰۰ دایره را گرفته، محیط و مساحت آن‌ها را بنویسد.

۲. برنامه‌ای بنویسید که ۷۰ عدد صحیح اتفاقی بین ۲ و ۲۷ بنویسد. (خود ۲ و ۲۷ هم امکان تولید داشته باشند)

۳. برنامه‌ای بنویسید که اعداد زیر را بنویسد:

۳ , ۶, ۹ , ... , ۳۰

۴. برنامه‌ای بنویسید که حاصل کسرهای زیر را بنویسد: (هر کدام در یک خط)

$$1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}, \cdots, \frac{1}{100}$$

۵. برنامه‌ای بنویسید که یک عدد گرفته، بگوید اول است یا مرکب.

۶. برنامه‌ای بنویسید که ۱۰۰ عدد بپرسد و در آخر تعداد اعداد بزرگ‌تر از ۵ را بنویسد.

۷. برنامه‌ای بنویسید که خروجی آن به شکل زیر باشد:

1	2
2	4
3	6
4	8
⋮	⋮
⋮	⋮
95	190

۸. برنامه‌ای بنویسید که N را بگیرد و !N را حساب کرده و نمایش دهد.

راهنمایی:

$$N! = 1 \times 2 \times 3 \times \cdots \times N$$

۹. برنامه‌ای بنویسید که حاصل a ^ b را بدون استفاده از توان و تنها با استفاده از ضرب‌های متوالی حساب کند.

$$a^b = \overbrace{a \times a \times \cdots \times a}^{b \text{ بار}}$$

۱۰. برنامه‌ای بنویسید که با گرفتن n (که یک عدد فرد بزرگ‌تر از ۷ است)، حاصل جمع دنباله‌های زیر را حساب کند:

$$1+\frac{1}{3}+\frac{1}{5}+\frac{1}{7}+\ldots+\frac{1}{n}$$

۱۱. در ریاضیات، عددی را که مساوی مجموع همه مقسوم علیه‌هایش (به جز خودش) باشد، عدد کامل می‌گویند. مثلاً ۲۸ عدد کامل است، چون ۲۸ = ۱۴ + ۷ + ۴ + ۲+ ۱. برنامه‌ای بنویسید که یک عدد از ما گرفته، بگوید کامل است یا نه.

۱۲. در ریاضیات ثابت می‌شود که :

$$1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\frac{1}{9}-\ldots=\frac{\pi}{4}$$

حال شما برنامه‌ای بنویسید که با استفاده از دنباله‌ی فوق و با گرفتن مخرج آخرین کسر مورد نظر، عدد π را حساب کند و نمایش دهد.

۱۳. در ریاضیات دو عدد را که حاصل جمع مقسوم علیه‌های هر یک برابر دیگری باشد، «هم‌دوست» می‌نامند. مثلاً ۲۲۰ و ۲۸۴ هم‌دوست هستند، زیرا :

حاصل جمع مقسوم‌علیه‌های ۲۲۰ = ۱+۲+۴+۵+۱۰+۱۱+۲۰+۲۲+۴۴+۵۵+۱۱۰ = ۲۸۴

حاصل جمع مقسوم‌علیه‌های ۲۸۴ = ۱+۲+۴+۷۱+۱۴۲ = ۲۲۰

حال شما برنامه‌ای بنویسید که دو عدد گرفته، بگوید هم دوست هستند یا نه.

۱۴. برنامه‌ای بنویسید که تمام اعداد دنباله فیبوناچی را که کمتر از ۱۰۰۰ هستند، نمایش دهد.

دنباله‌ی فیبوناچی = ۱ ، ۱ ، ۲ ، ۳ ، ۵ ، ۸ ، ۱۳ ، ۲۱ ، ۳۴ ، ...

۱۵. برنامه‌ای بنویسید که دو عدد گرفته، و ب.م.م آن‌ها را به روش نردبانی حساب کرده و نمایش دهد.

تمرین‌های مخصوص دانش‌آموزان علاقه‌مند:

۱. برنامه‌ای بنویسید که یک عدد گرفته، ارقام آن را برعکس بنویسد. مثلاً اگر عدد ۶۷۴۱ را وارد کنیم، بنویسد:

1
4
7
6

آیا می‌توانید خروجی را به صورت یک عدد بنویسید؟

۲. برنامه‌ای بنویسید که اعداد ۱ تا ۱۰۰ را ۵۰ بار بنویسد.

۳. برنامه‌ای بنویسید که تمام اعداد اول بین ۲ تا ۱۰۰۰ را بنویسد.

۴. برنامه‌ای بنویسید که جدول ضرب اعداد ۱ تا ۱۰ را به طور مرتب زیر هم بنویسد.

۵. برنامه‌ای بنویسید که تمام اعداد ۳ رقمی را که می‌توان با ارقام ۱ تا ۵ نوشت بنویسد.