فصل ۴

برنامه‌نویسی

- انواع کتابخانه در زبان++C را نام برده و کاربرد هر یک را توضیح دهید.
- تابع اصلی برنامه به زبان ++C چگونه است؟ با نوشتن آن هر خط را تبیین کنید.
- انواع متغیر را نام برده و برای هریک مثالی طرح کنید.
- مقدار دهی به متغیرها و چاپ متغیرها را توضیح دهید.

مقدمهٔ برنامه‌نویسی به زبان C++

یک رایانه با توجه به ساختاری که دارد، تنها می‌تواند دستورهایی از جنس اعداد صفر و یک (باینری[1]) را اجرا کند. در گذشته برنامه‌هایی که برای رایانه‌ها نوشته می‌شدند، تنها به زبان باینری بودند. به برنامه‌هایی که به زبان ماشین (باینری) نوشته می‌شوند، برنامه‌های سطح پایین[2] می‌گویند. با پیشرفت سخت‌افزارها و نرم‌افزارها و افزایش قابلیت رایانه‌ها، نوشتن برنامه‌ها به زبان ماشین، مشکلات زیادی به‌وجود آورد. این امر منشأ پیدایش زبان‌های سطح بالا[3] است. در زبان‌های سطح بالا دستورات به شکلی ساده و قابل فهم بیان می‌شوند و هرکدام از این دستورات، ساده شدهٔ چند دستور سطح پایین است. پس برای اجرای آن توسط رایانه، لازم است از مترجمی برای ترجمه زبان سطح بالا به زبان ماشین استفاده کرد. به این مترجم، کامپایلر[4] زبان C++ و به عملیات ترجمه، کامپایل کردن گفته می‌شود.

کتابخانه‌ها

کتابخانه‌ها برنامه‌هایی هستند که از قبل نوشته شده‌اند و ما می‌توانیم در برنامهٔ خود از آنها استفاده کنیم. جدول ۱ـ۴ برخی از کتابخانه‌های پرکاربرد در زبان C++ را نشان می‌دهد.

جدول ۱ـ۴ـ برخی از کتابخانه‌های پرکاربرد در زبان C++

نام کتابخانه		کاربرد
ورودی / خروجی	iostream	برای خواندن از صفحه کلید و نمایش دادن روی صفحه نمایش
ورودی / خروجی	stdio	برای خواندن از صفحه کلید و نمایش دادن روی صفحه نمایش
ریاضی	math	برای استفاده از توابع ریاضی مانند sin, cos, log
زمان	time	برای به‌دست آوردن زمان سیستم و انجام عملیات زمانی
رشته‌ها (عبارات متنی)	string	برای نگهداری و انجام محاسبات متنی

۱ـ Binary ۲ـ Low-level ۳ـ High-level ۴ـ Compiler

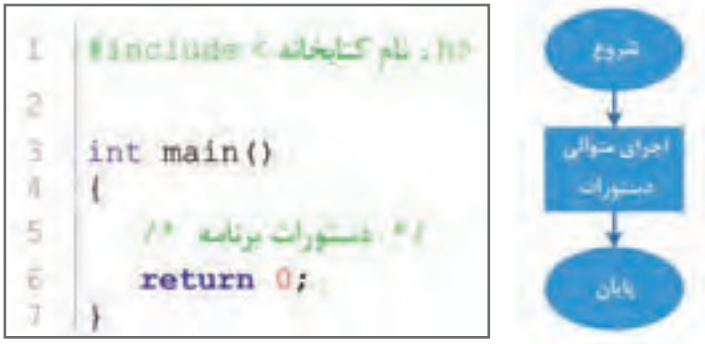
برای استفاده از هر کتابخانه لازم است در ابتدای برنامه نام آن را بنویسیم؛ به‌عنوان مثال اگر بخواهیم از کتابخانه ورودی / خروجی (iostream) استفاده کنیم، کافی است دستور زیر را در ابتدای برنامه بنویسیم:

```
#include <iostream>
```

تابع اصلی برنامه

هر برنامه شامل مجموعه‌ای از دستورات است و منظور از اجرای برنامه، اجرای متوالی دستورات آن است. در زبان برنامه‌نویسی C++، هر دستور در یک خط نوشته شده و در پایان آن دستور، علامت نقطه ویرگول[1] ';' نشانگر ختم دستور است.

هنگامی که برنامه‌ای اجرا می‌شود، در واقع تابع اصلی آن فراخوانی شده و دستورات آن به ترتیب اجرا می‌شوند. اگر اجرای این دستورات با موفقیت به پایان برسد، تابع اصلی مقدار عددی صفر را به عنوان خروجی موفقیت‌آمیز برمی‌گرداند. درصورتی‌که اجرای برنامه با خطایی مواجه شود، مقدار خروجی، دیگر صفر نخواهد بود و با توجه به مقدار خروجی می‌توان دربارهٔ صحت اجرای برنامه صحبت کرد. به‌طور کلی نمودار برنامهٔ ۱ـ۴ ساختار اصلی یک برنامه را نشان می‌دهد.



نمودار برنامه ۱ـ۴ـ ساختار اصلی یک برنامه

متغیرها

تعریف متغیرها

هر برنامه می‌تواند اطلاعات مورد نیاز خود را در حافظه ذخیره کند و در مواقع موردنیاز، مقدار آنها را بازیابی کرده یا تغییر دهد. به همین دلیل به آن متغیر می‌گوییم. از آنجایی که اطلاعات ممکن است از انواع گوناگون باشند، لازم است هنگام تعریف هر متغیر، نوع داده‌ای که قرار است در آن نگهداری شود مشخص گردد. اگر در بخشی از برنامه به اعداد صحیح (... و ۲ و ۱ و ۰ و ۱- و ۲- و...) احتیاج بود، کافی است متغیر را از نوع اعداد صحیح تعریف کرد. درحالی‌که برای کار با اعداد گویا، از نمایش اعشاری استفاده شده و متغیر از نوع اعداد اعشاری تعریف می‌شود. برای نگهداری عبارت (مانند نام یک دستگاه، آدرس دفتر نمایندگی و...) لازم است به ازای هر حرف از عبارت موردنظر، یک متغیر از نوع حرف (کاراکتر) تعریف کرده و مقدار لازم درون آن ذخیره گردد.

۱ـ Semicolon

بیشتر بدانیم

در رایانه، همهٔ اطلاعات درون حافظه نگهداری می‌شود. پس برای دسترسی به هر داده لازم است آدرس آن را روی حافظه بدانیم و با مراجعه به همان بخش از حافظه داده مورد نظر را بازیابی کنیم. همچنین می‌توانیم آن داده را تغییر داده یا حذف کنیم؛ اما اطلاعات ممکن است از جنس‌های گوناگون باشد. یک برنامه انبارداری را در نظر بگیرید. اگر هدف ثبت دستگاهی جدید در برنامه باشد، شماره سریال دستگاه داده‌ای عددی است. در حالی که نام دستگاه دنباله‌ای از حروف است؛ بنابراین برای ذخیره‌سازی اطلاعات علاوه‌بر آدرس آن روی حافظه، لازم است نوع آن را مشخص کنیم. در زبان‌های برنامه نویسی ابتدایی، تعیین مکان‌های داده روی حافظه به عهدهٔ خود برنامه‌نویس است؛ اما در رایانه‌های امروزی، به‌دلیل امکان اجرای هم‌زمان چند برنامه، این وظیفه به سیستم‌عامل محول گشته است و هنگام برنامه‌نویسی کافی است نوع داده را مشخص کنیم. از آنجایی که امکان تغییر برای این داده‌ها فراهم است، از این پس به آنها متغیر می‌گوییم.

انواع متغیرها

هر متغیر تنها می‌تواند نوع خاصی از مقادیر را در خود نگهداری کند و انتخاب این نوع با توجه به نیاز انجام می‌شود. جدول ۲ـ۴ برخی از متغیرهای پرکاربرد در زبان C++ را نشان می‌دهد.

جدول ۲ـ۴ـ برخی از متغیرهای پرکاربرد در زبان C++

نوع متغیر		مثال				
عدد صحیح	int	-12	1395	0	-5	27
عدد اعشاری	float	19.5	3.14	-500.0	12.333	-0.25
حرف (کاراکتر)	char	‘?’	‘+’	‘A’	‘h’	‘@’
صحیح/ غلط (بولی)	bool	true	1	false	.	

متغیرهای بولی تنها می‌توانند یکی از دو مقدار یک (true) یا صفر (false) را اختیار کنند.

نام متغیرها

هر متغیر با نامی مشخص معرفی می‌شود. این نام متشکل از حروف بزرگ و کوچک انگلیسی، کاراکتر زیرخط[۱] ‘_’ و اعداد است با این فرض که اولین حرف نمی‌تواند عدد باشد. اگرچه بهتر است هنگام تعیین نام متغیرها از اسامی با معنی استفاده کرد؛ استفاده از اسامی بی‌معنی خطایی در برنامه به وجود نمی‌آورد.

۱ـ Underline

جدول ۳ـ۴ مثال‌های درست و نادرست نام‌گذاری متغیرها را نشان می‌دهد.

جدول ۳ـ۴ـ مثال‌های درست و نادرست نام‌گذاری متغیرها

	مثال‌های درست	مثال‌های نادرست
متغیر عدد صحیح با نام num۱	int num۱;	int ۱num;
متغیر کاراکتری با نام _c	char _c;	char @c;
متغیر عدد اعشاری با نام num_۲	float num_۲;	float num#۲;
متغیر بولی با نام YesOrNo	bool YesOrNo;	bool yes~no;

مقداردهی به متغیرها

مقداردهی به متغیرها با استفاده از عملگر تساوی صورت می‌گیرد. بدین شکل که نام متغیر در سمت چپ علامت تساوی نوشته‌شده و سمت راست آن مقدار موردنظر قرار می‌گیرد. جدول ۴۴ مثال‌های درست و نادرست مقداردهی متغیرها را نمایش می‌دهد.

جدول ۴ـ۴ـ مثال‌های درست و نادرست مقداردهی متغیرها

	مثال‌های درست	مثال‌های نادرست
مقداردهی متغیر با نام num1 با مقدار 12	num1 = 12;	12 = num1;
مقداردهی متغیر با نام _c با کاراکتر ‘%’	_c = ‘%’;	_c=% ;
مقداردهی متغیر با نام num_2 با مقدار -3.6	num_2 = -3.6;	num_2 = -3.6;
مقداردهی متغیر با نام YesOrNo با مقدار true	YesOrNo = true;	false = YesOrNo;

ممکن است مقداردهی متغیرها هنگام تعریف انجام شود که به این کار مقداردهی اولیه می‌گویند. مقداردهی اولیه احتمال خطای برنامه‌نویسی را کاهش می‌دهد. جدول ۵ـ۴ مقداردهی اولیه به انواع متغیرها را نشان می‌دهد.

جدول ۵ـ۴ـ مقداردهی اولیه به انواع متغیرها

	مثال
مقداردهی اولیه متغیر عدد صحیح	int num1 = 12;
مقداردهی اولیه متغیر کاراکتری	char _c = ‘%’;
مقداردهی اولیه متغیر عدد اعشاری	float num_2 = -3.6;
مقداردهی اولیه متغیر بولی	bool YesOrNo = true;

بیشتر بدانیم

یک برنامه می‌تواند از قطعات متفاوتی تشکیل شده باشد. این قطعات با علامت {شروع و با علامت} خاتمه می‌یابند. تمام دستورهای درون یک قطعه به اندازهٔ یک باریکه (معمولاً به طول سه یا چهار کاراکتر) جلوتر از آکولادها نوشته می‌شود. اگر متغیری درون یک قطعه تعریف شود، پس از اتمام آن قطعه از بین می‌رود و دیگر نمی‌توان از آن استفاده کرد. اگر متغیری قبل و خارج از تابع اصلی برنامه تعریف شود، به عنوان متغیر جهانی شناخته شده و در همهٔ قطعات برنامه قابل استفاده است.

چاپ متغیرها در خروجی

برای چاپ مقدار یک متغیر در خروجی (صفحه نمایش) لازم است از دستور زیر استفاده کنیم:

cout << نام متغیر;

این دستور در کتابخانهٔ ورودی / خروجی (iostream) و در بخش استاندارد (std) قرار دارد پس لازم است، علاوه بر نوشتن نام کتابخانه در ابتدای برنامه، امکان استفاده از بخش استاندارد (std) را برای برنامه فراهم کنیم. این‌کار با نوشتن دستور زیر بعد از نام کتابخانه‌ها صورت می‌گیرد.

using namespace std;

برنامه ۱ـ۴ شامل چهار متغیر با جنس‌ها و مقادیر متفاوت است که پس از اجرای آن، محتویات هر متغیر در یک خط جداگانه چاپ می‌شود.

```
#include iostream                    فراخوانی کتابخانه iostream

using namespace std;                  معرفی بخش استاندارد جهت استفاده از cout و endl

int main()                            شروع تابع اصلی برنامه
{                                     
    int n = 12;                       تعریف متغیر عددی n با مقداردهی اولیه ۱۲
    float f = 3.14;                   تعریف متغیر اعشاری f با مقداردهی اولیه ۳.۱۴
    char c = 'H';                     تعریف متغیر کاراکتری c با مقداردهی اولیه 'H'
    bool s = true;                    تعریف متغیر بولی s با مقداردهی اولیه true

    cout << n << endl;                چاپ متغیر n و رفتن به ابتدای خط بعد
    cout << f << endl;                چاپ متغیر f و رفتن به ابتدای خط بعد
    cout << c << endl;                چاپ متغیر c و رفتن به ابتدای خط بعد
    cout << s << endl;                چاپ متغیر s و رفتن به ابتدای خط بعد

    return 0;                         برگرداندن مقدار صفر به عنوان خروجی تابع اصلی به
}                                     معنای پایان موفقیت آمیز تابع اصلی برنامه
```

خروجی برنامه

```
12
3.140000
H
1
```

برنامهٔ ۱ـ۴

متغیرها از ورودی

خواندن متغیرها از ورودی (صفحه کلید) به روشی مشابه نوشتن آنها در خروجی صورت می‌پذیرد.

cin >> نام متغیر ;

مثال: برنامه ۲ـ۴ نحوهٔ خواندن یک متغیر عددی از ورودی و نمایش دو برابر آن را در خروجی نشان می‌دهد.

```
#include <iostream>                    فراخوانی کتابخانه iostream

using namespace std;                   معرفی بخش استاندارد جهت استفاده از cout و endl

int main()                             شروع تابع اصلی برنامه
{                                      تعریف متغیر عددی n بدون مقدار اولیه
    int n;                             خواندن مقدار متغیر n از ورودی
    cin >> n;                          دوبرابر کردن متغیر n
    n = n * 2;                         چاپ متغیر n و رفتن به ابتدای خط بعد
    cout << n << endl;                 برگرداندن مقدار صفر به عنوان خروجی تابع اصلی به
    return 0;                          معنای پایان موفقیت آمیز تابع اصلی برنامه
}
```

```
ورودی برنامه
14
خروجی برنامه
28
```

برنامهٔ ۲ـ۴

عملگرها

برای انجام عملیات ریاضی مانند جمع و ضرب، می‌توان از علامت مربوط استفاده کرد و یک عبارت محاسباتی تولید کرد. جدول ۶ـ۴ برخی از پرکاربردترین عملگرهای زبان C++ را نشان می‌دهد.

جدول ۶ـ۴ـ برخی از پرکاربردترین عملگرهای زبان C++

نوع عملگر		مثال		
جمع	+	17 + (-3)	29 + 4.5	-5.75 + 8.8
تفریق	-	-7 – (-4)	19.99 – 4	6.1 – 0.007
ضرب	*	(-17) * 9	4.5 * 2	-1.5 * 33.6
تقسیم	/	15 / 5.5	17 / -4	-23 / 9
باقیمانده	%	12%3	23%7	-17% 6

به مقادیری که در طرفین عملگر قرار می‌گیرد، عملوند گفته می‌شود. اگر هر دو عملوند یک عملگر از جنس یکسان باشند (مثلاً هر دو عدد صحیح باشند) نتیجه از همان جنس خواهد بود. پس هنگامی‌که دو متغیر (یا مقدار ثابت) عدد صحیح، بر یکدیگر تقسیم می‌شوند، حاصل از جنس عدد صحیح (و نه اعشاری) خواهد بود؛ به به‌عنوان مثال حاصل عبارت 17/2 برابر 8 خواهد بود (و نه 8.5). به بیان کلی‌تر عملگر تقسیم برای اعداد صحیح، خارج قسمت تقسیم را محاسبه می‌کند. عملگر باقیمانده (%)، باقیمانده تقسیم اعداد صحیح را محاسبه می‌کند.

عبارات محاسباتی

گاهی می‌خواهیم مقدار یک معادله را به ازای مشخصی از پارامترها حساب کنیم. معادله ۸+x۴- را در نظر بگیرید. این معادله شامل یک پارامتر به نام x است. برنامه ۳ـ۴ مقدار عددی x را از ورودی می‌خواند. سپس مقدار معادله را در آن نقطه محاسبه کرده و حاصل را در خروجی چاپ می‌کند.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int x;
8      int y;
9
10     cin >> x;
11     y = -4 * x + 8;
12     cout << y << endl;
13
14     return 0;
15 }
```

ورودی برنامه	3
خروجی برنامه	-4

برنامه ۳ـ۴

مقایسه مقادیر ـ عبارات بولی

برای مقایسه مقدار دو متغیر (یا عبارت محاسباتی) از یکی از عملگرهای جدول ۷ـ۴ استفاده می‌کنیم. به هریک از مثال‌های جدول مقابل یک عبارت بولی گفته می‌شود که جواب این عبارت‌ها با توجه به پارامترهای آن می‌تواند صحیح (true) یا غلط (false) باشد.

جدول ۷ـ۴

نام عملگر	نماد عملگر	مثال
کوچک‌تر	<	x < (y + 3)
بزرگ‌تر	>	(x / 2) > y
مساوی	==	x == 5
کوچک‌تر یا مساوی	<=	x * y <= 0
بزرگ‌تر یا مساوی	>=	x >= y
نامساوی ـ مخالف	!=	x != y

دستورهای شرطی

دستورهای شرطی شامل یک عبارت بولی به‌عنوان شرط و یک قطعه برنامه به‌عنوان نتیجه هستند. هنگام اجرای دستور شرطی مقدار عبارت بولی آن محاسبه شده و اگر پاسخ صحیح (true) بود، قطعه نتیجه اجرا می‌شود. نمودار برنامه ۲ـ۴ نمونه استفاده از دستور شرطی را نشان می‌دهد و می‌توان در نقاط مختلف برنامه از آن استفاده کرد.


```
1  if ( شرط )
2  {
3      /* دستورات نتیجه */
4  }
```

نمودار برنامه ۲ـ۴ـ نمونه استفاده از دستور شرطی

مثال: برنامه ۴ـ۴ یک عدد صحیح از ورودی می‌خواند و مثبت بودن آن را در خروجی نمایش می‌دهد.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int n;
8      cin >> n;
9      if (n >= 0)
10     {
11         cout << "n is positive" << endl;
12     }
13     return 0;
14 }
```

ورودی برنامه

```
8
```

خروجی برنامه

```
n is positive
```

برنامه ۴ـ۴

در ادامهٔ یک دستور شرطی می‌توان قطعه برنامه دیگری نیز مشخص کرد تا در صورت عدم برقراری شرط مورد نظر، آن بخش اجرا شود. نمودار برنامه ۳ـ۴ نمونه استفاده از دستور شرطی به همراه دستورات جایگزین را نشان می‌دهد.


```
1  if ( شرط )
2  {
3      /* دستورات نتیجه */
4  }
5  else
6  {
7      /* دستورات جایگزین */
8  }
```

نمودار برنامه ۳ـ۴ـ نمونه استفاده از دستور شرطی به همراه دستورات جایگزین

برنامه ۵ـ۴ یک عدد اعشاری از ورودی می‌خواند و مثبت بودن یا نبودن آن را در خروجی نمایش می‌دهد.

```
using namespace std;

int main()
{
    float k;
    cin >> k;
    if (k >= 0)
    {
        cout << k;
        cout << [illegible] << endl;
    }
    else
    {
        cout << k;
        cout << [illegible] << endl;
    }
    return 0;
}
```

ورودی برنامه	ورودی برنامه
2.333	-8.57
خروجی برنامه	خروجی برنامه
n is positive	n is not positive

برنامه ۵ـ۴

حلقه‌های شرطی

حلقه‌های شرطی همانند دستورهای شرطی، شامل یک عبارت بولی به عنوان شرط و یک قطعه برنامه به عنوان نتیجه هستند. هنگام اجرای حلقه شرطی، ابتدا مقدار عبارت بولی آن محاسبه شده و اگر حاصل صحیح (true) بود، قطعه نتیجه اجرا می‌شود. در ادامه مجدداً مقدار عبارت بولی محاسبه شده و تا زمانی که مقدار آن صحیح (true) باشد، قطعه نتیجه اجرا می‌شود. اگر هنگام محاسبهٔ شرط حلقه، حاصل غلط (false) باشد دیگر قطعه نتیجه تکرار نمی‌شود و برنامه به سراغ دستورهای پس از حلقه می‌رود. نمودار برنامه ۴ـ۴ نمونه استفاده از حلقه تکرار را نشان می‌دهد.


```
1  while ( شرط )
2  {
3      /* دستورات نتیجه */
4  }
```

نمودار برنامه ۴ـ۴ـ نمونه استفاده از حلقه تکرار

مثال: برنامه ۶ـ۴ اعداد مثبت کوچک‌تر از ۵ را در خروجی چاپ می‌کند.

```
#include <iostream>                    فراخوانی کتابخانه iostream
using namespace std;                   معرفی بخش استاندارد جهت استفاده از cout

int main()                             شروع تابع اصلی برنامه
{
    int n = 0;                         تعریف متغیر عددی n با مقداردهی اولیه صفر
    while ( n < 5 )                    تازمانیکه n کوچکتر از ۵ است
    {
        cout << n << " ";              مقدار n را چاپ کن
        n = n + 1;                     مقدار n را یک واحد افزایش بده
    }
    return 0;                          برگرداندن صفر به معنای پایان موفقیت‌آمیز تابع اصلی برنامه
}
```

خروجی برنامه

```
0 1 2 3 4
```

برنامه ۶ـ۴

حلقه‌های شرطی بی‌پایان

اگر عبارت شرطی یک حلقه، همواره صحیح (true) باشد، آن حلقه همواره تکرار می‌شود. برنامه ۷ـ۴ به دفعات نامحدود از ورودی عدد می‌خواند و زوج یا فرد بودن آن را مشخص می‌کند.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int n;
8      while ( true )
9      {
10         cin >> n;
11         if (n % 2 == 0)
12         {
13             cout << n  << " is even." << endl;
14         }
15         else
16         {
17             cout << n  << " is odd." << endl;
18         }
19     }
20     return 0;
21 }
```

ورودی برنامه

```
5
-4
33
-7
0
```

خروجی برنامه

```
5 is odd.
-4 is even.
33 is odd.
-7 is odd.
0 is even.
```

برنامه ۷ـ۴

ترکیب چند عبارت بولی

برای ساخت برخی شرط‌ها در دستورهای شرطی و حلقه‌ها، می‌توان عبارات بولی را با یکدیگر ترکیب کرد و شرط‌های پیچیده‌تر ساخت. جدول ۸ـ۴ عملگرهای «و» و «یا» را به‌عنوان پرکاربردترین عملگرهای عبارات بولی نمایش می‌دهد.

جدول ۸ـ۴ـ عملگرهای «و» و «یا»

نام عملگر	نماد عملگر	مثال
و	&&	(x < ۳) && (y > ۷)
یا	||	(x > ۵) || (x < -۵)

الگوهای برنامه‌نویسی

تکرار عملی برای تعداد مشخص

اگر نیاز باشد قطعه برنامه‌ای به دفعات مشخصی تکرار شود، می‌توان با استفاده از یک متغیر که در آن «تعداد دفعات باقی‌مانده» نگهداری می‌شود، یک حلقه ایجاد کرد و تا زمانی که متغیر موردنظر به صفر نرسیده است، حلقه را تکرار کرد. لازم است در هر بار اجرای حلقه، یک واحد از مقدار متغیر کاسته شود که این امر نشانگر پایان اجرای آن مرحله است.

مثال: برنامه ۸ـ۴ عدد صحیح n را از کاربر می‌گیرد و همهٔ اعداد n تا یک را در خطوط جداگانه چاپ می‌کند.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int n;
8      cin >> n;
9      while ( n > 0 )
10     {
11         cout << n << endl;
12         n = n - 1;
13     }
14     return 0;
15 }
```

ورودی برنامه

```
5
```

خروجی برنامه

```
5
4
3
2
1
```

برنامه ۸ـ۴

مثال: برنامه ۹ـ۴ عدد صحیح n را از کاربر می‌گیرد و مجموع همهٔ اعداد n تا یک را محاسبه می‌کند. دربارهٔ نقش متغیر sum در کلاس بحث کنید.

```
#include <iostream>

using namespace std;

int main()
{
    int n;
    int sum = 0;
    cin >> n;
    while ( n > 0 )
    {
        sum = sum + n;
        n = n - 1;
    }
    cout << sum << endl;
    return 0;
}
```

ورودی برنامه

```
5
```

خروجی برنامه

```
10
```

برنامه ۹ـ۴

مثال: برنامه ۱۰ـ۴ عدد صحیح n را از کاربر می‌گیرد و همهٔ اعداد زوج کوچک‌تر یا مساوی n را چاپ می‌کند.

```
#include <iostream>

using namespace std;

int main()
{
    int n;
    int m = 0;
    cin >> n;
    while ( m <= n )
    {
        cout << m << endl;
        m = m + 2;
    }
    return 0;
}
```

ورودی برنامه

```
9
```

خروجی برنامه

```
0
2
4
6
8
```

برنامه ۱۰ـ۴

تمرین

برنامه‌ای بنویسید که:

- دو عدد صحیح از کاربر بگیرد و میانگین آن دو عدد را نمایش دهد.
- برنامه فوق را ۵ مرتبه تکرار کند.
- برنامه فوق را تا زمانی که هر دو عدد وارد شده مثبت هستند تکرار کند.

تکرار عملی برای تعداد دلخواه

در حلقه‌ها می‌توان شرط را به گونه‌ای تنظیم کرد که تکرار حلقه با اختیار کاربر هنگام اجرای برنامه مشخص گردد. فرض کنید برنامه قرار است نمرات کلاسی دانش‌آموزی را بگیرد و میانگین آنها را به عنوان نمرهٔ کلاسی گزارش کند. عملیات خواندن نمرات باید با استفاده از حلقه تکرار شود؛ ولی از آنجایی که معلوم نیست برای هر دانش‌آموز، چه تعداد نمره ثبت شده است، نمی‌توان عدد ثابتی را برای تکرار حلقه در نظر گرفت. با فرض مثبت بودن همهٔ نمرات، شرط حلقه را مثبت بودن نمرهٔ وارد شده در نظر می‌گیریم. پس زمانی که وارد کردن نمرات به پایان رسید، واردکردن یک عدد منفی باعث اتمام عملیات خواندن نمرات می‌شود. اگر در طول خواندن نمرات، تعداد و مجموع آنها نگهداری شده باشد، محاسبهٔ میانگین به‌سادگی انجام می‌شود. برنامه ۱۱ـ۴ نمونهٔ ساده‌ای از پیاده‌سازی این مسئله است.

```
using namespace std;

int main()
{
    float score = 0;
    int count = 0;
    float sum = 0;
    float average;

    cin >> score;
    while ( scoure >= 0 )
    {
        sum = sum + scoure;
        count = count + 1;
        cin >> scoure;
    }
    if (count == 0)
    {
        cout << "No score found";
    }
    else
    {
        average = sum / count;
        cout << "sum: " << sum << endl;
        cout << "count: " << count << endl;
        cout << "average: " << average << endl;
    }
    return 0;
}
```

ورودی برنامه

```
19
16
18
15
-1
```

خروجی برنامه

```
sum: 68.000000
count: 4
average: 17.000000
```

برنامه ۱۱ـ۴

ارزشیابی پایانی

۱ باتوجه به نمودارهای داده شده، تفاوت دستورهای شرطی با حلقه‌ها در چیست؟

۲ برنامه‌ای بنویسید که عدد صحیح n را از کاربر بگیرد و اعداد فرد بین یک تا n را به ترتیب چاپ کند.

۳ برنامه‌ای بنویسید که مجموعه‌ای (با طول نامشخص) از اعداد اعشاری مثبت از کاربر بگیرد و بزرگ‌ترین آنها را چاپ کند (راهنمایی: در هر مرحله بزرگ‌ترین عدد یافت شده را نگهداری کنید. وارد کردن یک عدد منفی به معنای پایان مجموعه داده‌های ورودی است.)

۴ برنامه‌ای مطابق نمودار داده شده بنویسید.

۵ با توجه به برنامه زیر، نمودار رسم کنید.

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main()
6  {
7      int n;
8      float power = 1;
9      cout << "Enter a positive number: ";
10     cin >> score;
11     while ( n > 0 )
12     {
13         power = power * 2;
14         n = n - 1;
15     }
16     cout << "2 power n: " << power;
17     return 0;
18 }
```

منابع و مآخذ

فصل ۱

۱ Isermann, R. (Rolf): Mechatronic systems: fundamentals, ISBN 1852339306 Springer-Verlag London Limited 2005

۲ Sabri Cetinkunt: Mechatronics, John Wiley and sons 2007

۳ Robert H. Bishop, The Mechatronics Handbook Mechatronic System Control, Logic, and data Acquisition (Second Edition), 2002 by CRC Press LLC

۴ Introduction to Mechatronics and Measurement systems: Fourth Edition, David G. Alciatore Michael B. Histand

فصل ۲

۱ طراحی اجزای ماشین جلد ۱ و ۲ـ ترجمه مهندس فرامرزی انتشارات طرح (سال ۱۳۸۵)

۲ پرسش‌های چهار گزینه‌ای رسم فنی و نقشه‌خوانی طبق ـ حمیدرضا غلامرضایی، محسن بیاتی مؤسسه فرهنگی هنری دیباگران تهران ـ (سال ۱۳۸۴)

۳ رسم فنی ـ ترجمه باقر رجالی ـ انتشارات خوارزمی (سال ۱۳۸۰)

۴ درس فنی سال دوم نظام قدیم هنرستان کد ۵۰۳ ـ تألیف: محمد خادمی اقدم ـ بهروز نصیری زنوزی (سال ۱۳۶۸) وزارت آموزش و پرورش

۵ درس فنی سال سوم نظام قدیم هنرستان کد ۶۰۳ ـ تألیف: محمد خادمی اقدم ـ بهروز نصیری زنوزی (سال ۱۳۶۹) وزارت آموزش و پرورش

۶ جداول و استانداردهای طراحی و ماشین‌سازی ـ ترجمه مهندس ولی‌نژاد انتشارات طرح ـ (سال ۱۳۸۲)

۷ راهنمای مهندسان و تکنسین‌ها، ترجمه مهندس وحیدی ـ انتشارات ساپکو (سال ۱۳۸۹)

۸ نقشه‌کشی صنعتی ـ جلد ـ ، مهندس محمود مرجانی ـ انتشارات دانشگاه یزد (۱۳۹۱)

۹ رسم فنی تخصصی رشتهٔ ساخت و تولید (کد ۶ـ۴۸۸) حمیدرضا غلامرضایی سال (۱۳۹۲) وزارت آموزش و پرورش
۱۰ جزوه دانشگاهی بادامک‌ها ـ دانشگاه صنعتی شریف ـ تألیف دکتر ایرج ابطحی
۱۱ Madhine Drawing by: N. SIDHESWAR Mcgraw Hill pudlinhed 1990
تصاویر از سایت‌های مختلف گرفته شده است

فصل ۳

الکترونیک کاربردی ـ تألیف: شهرام نصیری سوادکوهی، شهرام خدادادی ـ سال (۱۳۹۴) وزارت آموزش و پرورش
مبانی برق ـ تألیف: فریدون قیطرانی و دیگران ـ سال (۱۳۹۴) وزارت آموزش و پرورش
الکترونیک عمومی ـ تالیف: محمود همتایی و دیگران ـ سال (۱۳۹۴) وزارت آموزش و پرورش
Giorgio Rizzoni: Principles and Applications of Electrical Engineering, McGraw-Hill Higher Education (2003)

فصل ۴

۱ Practical Programming in C++ by Steve Oualline
۲ The C++ Cookbook by D. Ryan Stephens, Christopher Diggins, Jonathan Turkanis, and Jeff Cogswell
۳ C++ How to Program by Deitel and Deitel
۴ Design Patterns: Elements of Reusable Object-Oriented Software by Gamma et. All
۵ Introduction to Algorithms, Third Edition by cormen leiserson rivest and stein

هنر آموزان محترم، هنرجویان عزیز و اولیای آنان می توانند نظرهای اصلاحی خود را دربارهٔ مطالب این کتاب از طریق نامه به نشانی تهران - صندوق پستی ۴۸۷۴ / ۱۵۸۷۵ - گروه درسی مربوط و یا پیام نگار tvoccd@roshd.ir ارسال نمایند.

وب گاه: www.tvoccd.medu.ir

دفتر تألیف کتاب های درسی فنی و حرفه ای و کار دانش