برنامه نویسی مقدماتی

مدرس : هادی حبیبی
ایمیل : h.habibi@irib.ir

تارنما:rlecture.ir

https://piazza.com/uast.ac.ir/fall2013/ce1/resources

جلسه چهارم

فهرست

- محاسبات در C
- دستورات تصمیم گیری
- الگوریتم
- شبه کد
- ساختارهای کنترلی
  - `If`
  - `If…else`
  - `while`

محاسبات در C

- علامت asterisk(*) برای عملیات ضرب و علامت percent sign(%) برای محاسبه باقیمانده استفاده می شوند.
- در جبر ریاضی اگر بخواهیم دو عدد a و b را در هم ضرب کنیم از عبارت ab استفاده می کنیم. ولی در C این عملیات اشتباه می باشد و باید حتما از عبارت a*b استفاده کنیم.

C operation	Arithmetic operator	Algebraic expression	C expression
Addition	+	$f + 7$	`f + 7`
Subtraction	–	$p - c$	`p - c`
Multiplication	*	bm	`b * m`
Division	/	x / y or $\frac{x}{y}$ or $x \div y$	`x / y`
Remainder	%	$r \bmod s$	`r % s`

Fig. 2.9 | Arithmetic operators.

محاسبات در C ...

- تمام عملگرهای ریاضی عملگرهای دودویی هستند.
- برای مثال عبارت 7 + 3 دارای عملگر + و دو عملوند 3 و 7 می باشد.
- حاصل تقسیم دو عدد صحیح یک عدد صحیح می شود.
- برای مثال، تقسیم 7/4 مقدار 1 و تقسیم 17/5 مقدار 3 را نتیجه می دهد.
- تقسیم بر مقدار صفر موجب بروز خطای مهلک (`fatal error`) می شود و برنامه از ادامه اجرا خارج می شود.
- عملگر باقیمانده (%) برای بدست آوردن باقیمانده تقسیم استفاده می شود.
- عملگر باقیمانده یک عملگر صحیح است و تنها برای اعداد صحیح قابل استفاده است
- به عنوان مثال 7%4 باقیمانده ۳ را برمی گرداند و 17%5 باقیمانده ۲ را بر می گرداند.

محاسبات در C ...

- عبارات ریاضی در C باید در یک خط نوشته شوند. عملگرها و عملوندها باید در یک خط نوشته شوند. به عنوان مثال برای تقسیم a بر b باید نوشت a/b.
- استفاده از پرانتز همانند استفاده آن در عبارات جبری می باشد. همچنین استفاده از پرانتزهای اضافی برای خوانایی عبارت جبری بلامانع است.
- a * (b + c) : ابتدا b و c جمع می شوند و سپس در a ضرب می شوند.
- تقدم انجام عملگرها در زبان C مانند انجام آنها در جبر می باشد. پرانتز بیشترین اولویت اجرا را دارد. در مورد پرانتزهای تو در تو ، ابتدا عبارات پرانتزهای داخلی محاسبه می شود.
- بعد از پرانتزها اولویت اجرا با ضرب و تقسیم و باقیمانده می باشد.
- جمع و تفریق در اولویت بعدی قرار دارند.
- اگر چندین عملگر با اولویت یکسان باشند ابتدا سمت چپ ترین عملگر اجرا میشود.

محاسبات در C ...

Operator(s)	Operation(s)	Order of evaluation (precedence)
( )	Parentheses	Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses "on the same level" (i.e., not nested), they're evaluated left to right.
*	Multiplication	Evaluated second. If there are several, they're evaluated left to right.
/	Division	
%	Remainder	
+	Addition	Evaluated last. If there are several, they're evaluated left to right.
-	Subtraction	

Fig. 2.10 | Precedence of arithmetic operators.

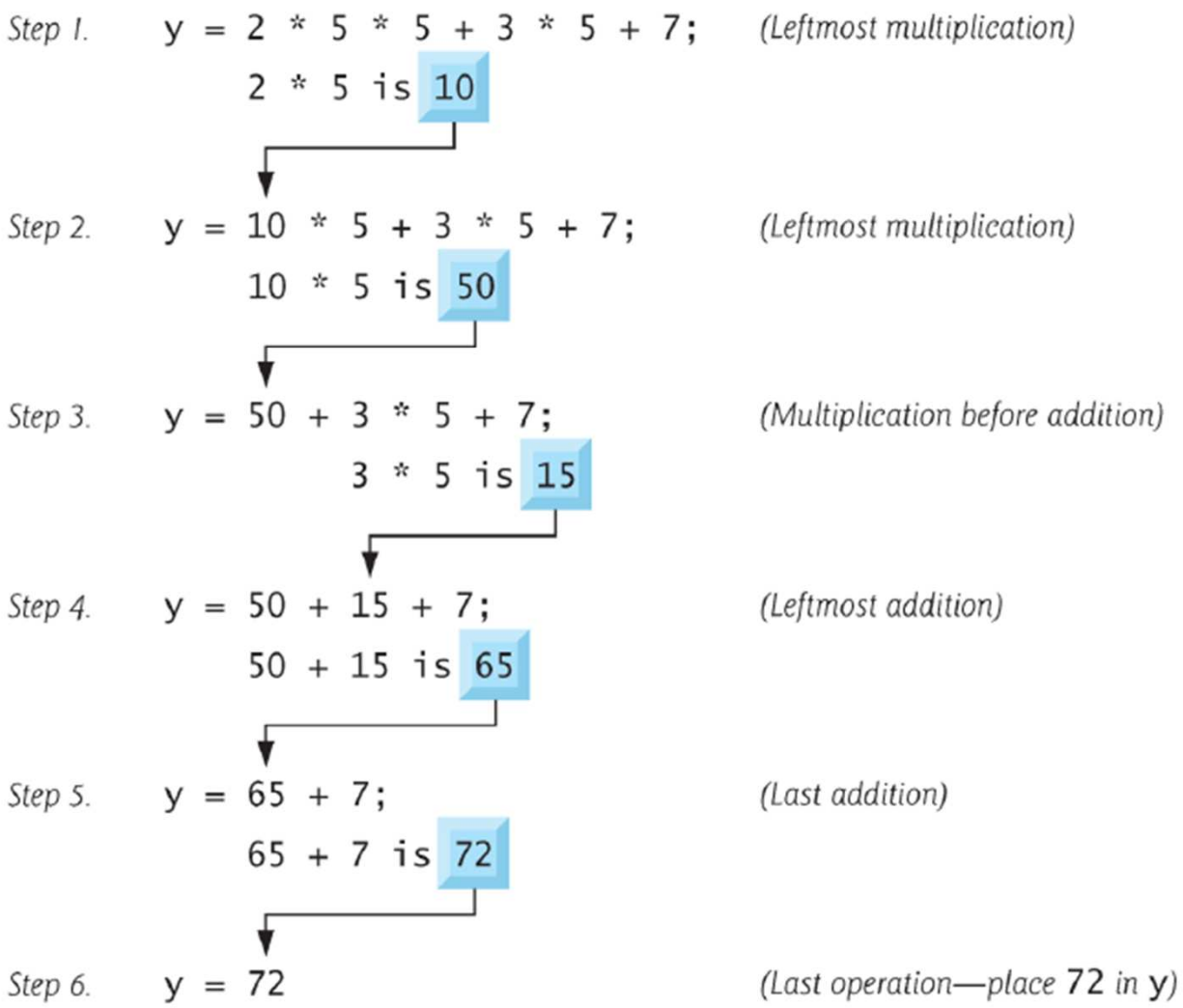

Fig. 2.11 | Order in which a second-degree polynomial is evaluated.

تصمیم گیری : عملگرهای مقایسه ای و رابطه ای

- خطوط دستوری زبان C
  - عملیاتی مانند محاسبات یا ورودی و خروجی
  - اتخاذ تصمیم
- در برنامه نویسی نیاز داریم در هنگام اجرای برنامه تصمیماتی را بگیریم. مثلا نمره دانشجویی را چک کنیم و در صورتی که بالای ۱۲ بود پیغام موفقیت در امتحان را برای او صادر کنیم.
- دستور `if` به برنامه اجازه می دهد بر مبنای درست و غلط بودن یک شرط یا condition تصمیمی را اتخاذ کند.
- اگر آن شرط درست بود دستورات موجود در بدنه `if` اجرا میشوند و اگر غلط بود دستورات بدنه `if` اجرا نمی شوند.
- شرط ها در دستور `if` بوسیله عملگرهای مقایسه ای یا رابطه ای تعریف می شوند.

عملگرهای مقایسه ای و رابطه ای ...

- عملگرهای رابطه های (relational) از اولویت اجرای یکسانی برخوردارند. بنابراین تنها اولویت حاکم بر آنها اجرا از چپ به راست می باشد.
- عملگرهای مقایسه ای (equality) اولویت پایینتری نسبت به عملگرهای رابطه ای دارند و همچنین ترتیب اجرای آنها از چپ به راست می باشد.
- در زبان C، هر عبارتی که نتیجه `0` یا غلط (`false`) و نتیجه غیر صفر یا درست (`true`) را داشته باشد یک شرط محسوب می شود.

عملگرهای مقایسه ای و رابطه ای ...

Algebraic equality or relational operator	C equality or relational operator	Example of C condition	Meaning of C condition
Equality operators			
=	==	x == y	x is equal to y
≠	!=	x != y	x is not equal to y
Relational operators			
>	>	x > y	x is greater than y
<	<	x < y	x is less than y
≥	>=	x >= y	x is greater than or equal to y
≤	<=	x <= y	x is less than or equal to y

Fig. 2.12 | Equality and relational operators.

الگوریتم ها

- قبل از اینکه برنامه ای بنویسید تا مشکل خاصی را حل کنید، این ضروری است که مساله را به خوبی درک کنید و روشی را برای حل مساله با دقت طراحی کنید.
- راه حل هر مساله محاسباتی شامل اجرای یک سری از دستورات با ترتیب مشخصی می باشد:
- یک procedure یا رویه برای حل مساله شامل :
  - عملیاتی (actions) برای اجرا شدن، و
  - ترتیب (order) خاصی برای اجرای این عملیات
- به این رویه الگوریتم می گویند.
- اجرای دستورات با نظم و ترتیب مشخص امر مهمی است. این عمل بر عهده کنترلر برنامه می باشد.

شبه کد (Pseudocode)

- شبه کدها زبان ها هوشمند و غیر رسمی هستند که به شما کمک می کنند تا الگوریتم را طراحی کنید.
- شبه کد ها شبیه زبان روزمره انگلیسی هستند. آنها راحت و قابل درک برای استفاده کنندگان از آن هستند ولی زبان های برنامه نویسی کامپیوتر نیستند.
- شبه کدها قابل اجرا در کامپیوتر نیستند.
- ولی به ما کمک می کنند تا درباره برنامه فکر کنیم قبل از اینکه آنها را توسط زبان برنامه نویسی مانند C پیاده کنیم.
- یک شبه کد به خوب، می تواند به سادگی تبدیل به یک برنامه C شود.
- شبه کد فقط از مجموعه عملیاتی که قرار است اجرا شود تشکیل شده است. عملیاتی که بعد از تبدیل به برنامه C اجرا می شوند.
- تعاریف (definitions) جزو عملیات اجرایی نیستند. بنابراین در شبه کد بیان نمی شوند.

شبه کد ...

- تعاریف تنها پیغام هایی را برای کامپایلر ارسال می کنند به عنوان مثال `int i` به کامپایلر می گوید فضایی را در حافظه برای متغیر `i` از نوع صحیح در نظر بگیرد.
- این دستور هیچ عملیاتی مانند ورودی و خروجی یا محاسبات را در پی ندارد.
- برخی برنامه نویسان ترجیح می دهند که لیستی از متغیرها و توضیحی در مورد هر کدام از آنها را در ابتدای شبه کد بیان کنند.

ساختار کنترل

- در حالت عادی دستورات یکی پس از دیگری اجرا می شوند. به این روش اجرای ترتیبی می گویند.
- دستورات مختلفی در C هستند که موجب می شوند تا این روند رعایت نشود و اجرای برنامه از جایی به جای دیگر منتقل شود. به این عمل انتقال کنترل می گویند.
- دستور go to به برنامه نویسان اجازه می دهد که کنترل برنامه را به هرجای برنامه منتقل کنند.
- استفاده از دستور go to با برنامه نویس ساختیافته در تقابل است.
- تحقیقات ثابت می کند که برنامه ها را می توان بدون استفاده از این دستور نوشت.

ساختار کنترل ...

- برنامه هایی که از تکنیک های ساختیافته استفاده می کنند، خواناتر ، راحتتر برای خطایابی (debug) و تغییر، و با خطا های کمتری همراه می باشد.
- تحقیقات نشان می دهد که تمام برنامه ها می تواند با سه ساختار کنترلی پیاده شوند:
  - Sequence structure ساختار ترتیبی
  - Selection structure ساختار انتخابی
  - Repetition structure ساختار تکرار
- ساختار ترتیبی در خود C نهفته است.

فلوچارت flowchart

- فلوچارت نمایش گرافیکی یک الگوریتم یا بخشی از می باشد.
- فلوچارت ها توسط اشکال گرافیکی خاصی رسم می شوند : مستطیل، لوزی، بیضی، دایره. این اشکال توسط خطوطی با نام flowline به هم متصل می شوند.
- همانند شبه کد، فلوچارت ابزاری مناسب برای پیاده سازی الگوریتم ها می باشد. ولی اکثر برنامه نویسان شبه کد ها را ترجیح می دهند.

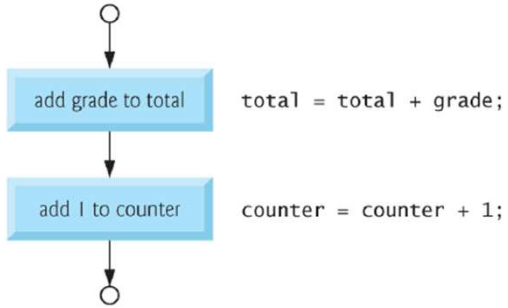

Fig. 3.1 | Flowcharting C's sequence structure.

دستورات ساختار کنترل

- زبان C سه دستور برای ساختار انتخاب دارد.
- دستور `if` یا دستور انتخاب تکی : اگر شرط آن درست باشد عملیات انتخاب انجام می شود و در غیر اینصورت از انجام آن عملیات صرفنظر می شود.
- دستور `if…else` یا دستور انتخاب دوتایی : اگر شرط آن درست باشد یک سری از عملیات را انجام می دهد و اگر درست نباشد عملیات دیگری را انجام می دهد.
- دستور `switch` یا دستور انتخاب چندتایی : با توجه به مقدار عبارت ورودی یکی از چندین عملیات ممکن را انجام میدهد.
- زبان C سه نوع دستور برای پیاده سازی ساختار تکرار دارد، `while` و `do…While` و `for`.

دستور انتخاب `if`

- شکل ۲.۱۳ از ۶ دستور `if` برای مقایسه دو عدد که از ورودی خوانده می شود استفاده می کند.
- اگر هر یک از شروط دستورات `if` درست باشد `printf` متناظر با آن اجرا شده و پیغامی را در خروجی چاپ می کند.

دستور انتخاب if

```
1  /* Fig. 2.13: fig02_13.c
2     Using if statements, relational
3     operators, and equality operators */
4  #include <stdio.h>
5
6  /* function main begins program execution */
7  int main( void )
8  {
9     int num1; /* first number to be read from user  */
10    int num2; /* second number to be read from user */
11
12    printf( "Enter two integers, and I will tell you\n" );
13    printf( "the relationships they satisfy: " );
14
15    scanf( "%d%d", &num1, &num2 ); /* read two integers */
16
17    if ( num1 == num2 ) {
18       printf( "%d is equal to %d\n", num1, num2 );
19    } /* end if */
20
21    if ( num1 != num2 ) {
22       printf( "%d is not equal to %d\n", num1, num2 );
23    } /* end if */
```

Fig. 2.13 | Using `if` statements, relational operators, and equality operators. (Part 1 of 3.)

دستور انتخاب if

```
24
25      if ( num1 < num2 ) {
26         printf( "%d is less than %d\n", num1, num2 );
27      } /* end if */
28
29      if ( num1 > num2 ) {
30         printf( "%d is greater than %d\n", num1, num2 );
31      } /* end if */
32
33      if ( num1 <= num2 ) {
34         printf( "%d is less than or equal to %d\n", num1, num2 );
35      } /* end if */
36
37      if ( num1 >= num2 ) {
38         printf( "%d is greater than or equal to %d\n", num1, num2 );
39      } /* end if */
40
41      return 0; /* indicate that program ended successfully */
42   } /* end function main */
```

Fig. 2.13 | Using `if` statements, relational operators, and equality operators. (Part 2 of 3.)

دستور انتخاب if

```
Enter two integers, and I will tell you
the relationships they satisfy: 3 7
3 is not equal to 7
3 is less than 7
3 is less than or equal to 7
```

```
Enter two integers, and I will tell you
the relationships they satisfy: 12 12
22 is not equal to 12
22 is greater than 12
22 is greater than or equal to 12
```

```
Enter two integers, and I will tell you
the relationships they satisfy: 7 7
7 is equal to 7
7 is less than or equal to 7
7 is greater than or equal to 7
```

Fig. 2.13 | Using `if` statements, relational operators, and equality operators. (Part 3 of 3.)

دستور انتخاب if ...

- این برنامه در خط ۱۵ از دستور scanf برای دریافت دو عدد از ورودی استفاده می کند.
- در این دستور : %d اول مقدار ورودی اول را تبدیل به عدد صحیح کرده و در متغیر num1 ذخیره می کند و %d دوم ورودی دوم را تبدیل به عدد صحیح کرده و در متغیر num2 ذخیره می کند.
- استفاده از خطوط خالی بین دستورات if برای خوانایی بیشتر می باشد.
- ; به تنهایی یک دستور محسوب می شود، بنابراین اگر بعد از دستور if به اشتباه ; قرار دهید کامپایلر آنرا به عنوان عملیات بعد از if در نظر می گیرد و خط بعدی به عنوان عملیات انتخاب محسوب نمی شود.

کلمات اصلی

- برخی کلمات از کلمات اصلی زبان C محسوب می شود و در برنامه نویسی از این کلمات به عنوان نام متغیر استفاده نمی شود. این کلمات که دستورات و تعاریف زبان C می باشند در جدول زیر آمده.

Keywords			
auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while

Keywords added in C99

`_Bool` `_Complex` `_Imaginary` `inline` `restrict`

Fig. 2.15 | C's keywords.

مثال انتخاب if

- فرض کنید حداقل نمره قبولی در یک امتحان ۶۰ باشد. شبه کدی که برای تشخیص قبولی استفاده می شود به شکل زیر می شود.

If student's grade is greater than or equal to 60
Print "Passed"

- اگر شرط درست باشد، واژه paased چاپ می شود.
- این شبه کد در زبان C به شکل زیر نوشته می شود.

```
if ( grade >= 60 ) {
   printf( "Passed\n" );
} /* end if */
```

- همانطور که می بینید کد C همانند شبه کد است.

فلوچارت مثال if

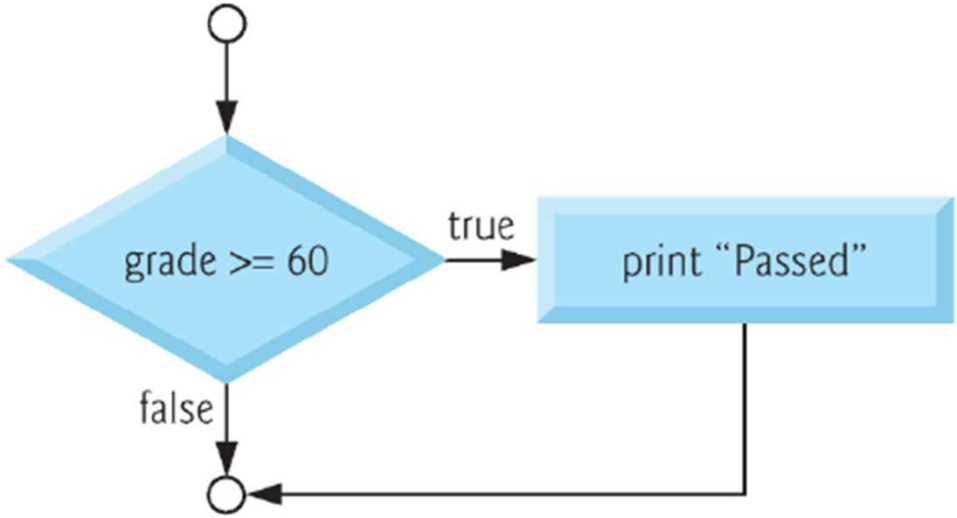

Fig. 3.2 | Flowcharting the single-selection `if` statement.

دستور انتخاب if…else

- از چندین دستور تو در تو if...else می توان برای انتخاب چندین گزینه استفاده کرد. برای مثال در شبه کد زیر، اگر نمره بیشتر یا مساوی ۹۰ باشد حرف A چاپ می شود، اگر بیشتر مساوی ۸۰ باشد حرف B چاپ می شود و به همین ترتیب حروف دیگر :

```
If student's grade is greater than or equal to 90
Print "A"
else
If student's grade is greater than or equal to 80
Print "B"
else
If student's grade is greater than or equal to 70
Print "C"
else
If student's grade is greater than or equal to 60
Print "D"
else
Print "F"
```

دستور انتخاب if…else ...

- این شبه کد به صورت زیر در C نوشته می شود.

```
if ( grade >= 90 )
   printf( "A\n" );
else
   if ( grade >= 80 )
      printf("B\n");
   else
      if ( grade >= 70 )
         printf("C\n");
      else
         if ( grade >= 60 )
            printf( "D\n" );
         else
            printf( "F\n" );
```

دستور انتخاب if…else ...

- بسیاری از برنامه نویسان ترجیح می دهند کد قبل را به این شکل بنویسند.

```
if ( grade >= 90 )
   printf( "A\n" );
else if ( grade >= 80 )
   printf( "B\n" );
else if ( grade >= 70 )
   printf( "C\n" );
else if ( grade >= 60 )
   printf( "D\n" );
else
   printf( "F\n" );
```

دستور انتخاب if…else ...

- در این مثال، در قسمت else از چند دستور استفاده شده است.

```
if ( grade >= 60 ) {
   printf( "Passed.\n" );
} /* end if */
else {
   printf( "Failed.\n" );
   printf( "You must take this course
again.\n" );
} /* end else */
```

اپراتور شرطی

- در زبانC به جای استفاده از `if...else` می توان از اپراتور شرطی `?:` استفاده کرد به عنوان مثال هر دو برنامه زیر یک کار انجام می دهند.

```
if ( grade >= 60 ) {
   printf( "Passed\n" );
} /* end if */
else {
   printf( "Failed\n" );
} /* end else */
```

- و

```
printf( "%s\n", grade >= 60 ?
   "Passed" : "Failed" );
```

ساختار تکرار while

- ساختار تکرار این امکان را میدهد تا عملیاتی تا وقتی شرطی درست است تکرار شود.
- بدنه دستور while می تواند یک دستور یا مجموعه ای از دستورات باشد.
- هر گاه شرط دستور while نادرست باشد دیگر دستورات بدنه آن اجرا نمی شود و کنترل برنامه به اولین دستور بعد از بدنه while منتقل می شود.
- بع عنوان مثالی از while : فرض کنید می خواهیم اولین عدد بزرگتر از ۱۰۰ که مضربی از ۳ باشد را پیدا کنیم.
- متغیر product با مقدار اولیه 3 در نظر بگیرید.
- وقتی حلقه زیر به پایان برسد متغیر product حاوی مقدار مورد جستجو خواهد بود.

```
product = 3;
while ( product <= 100 ) {
    product = 3 * product;
} /* end while */
```

ساختار تکرار while ...

- فلوچارت

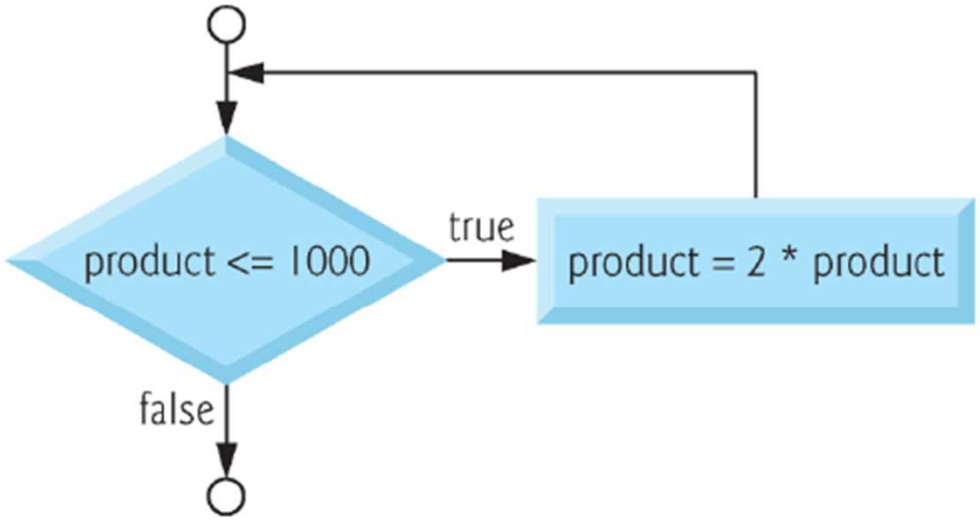

Fig. 3.4 | Flowcharting the `while` repetition statement.