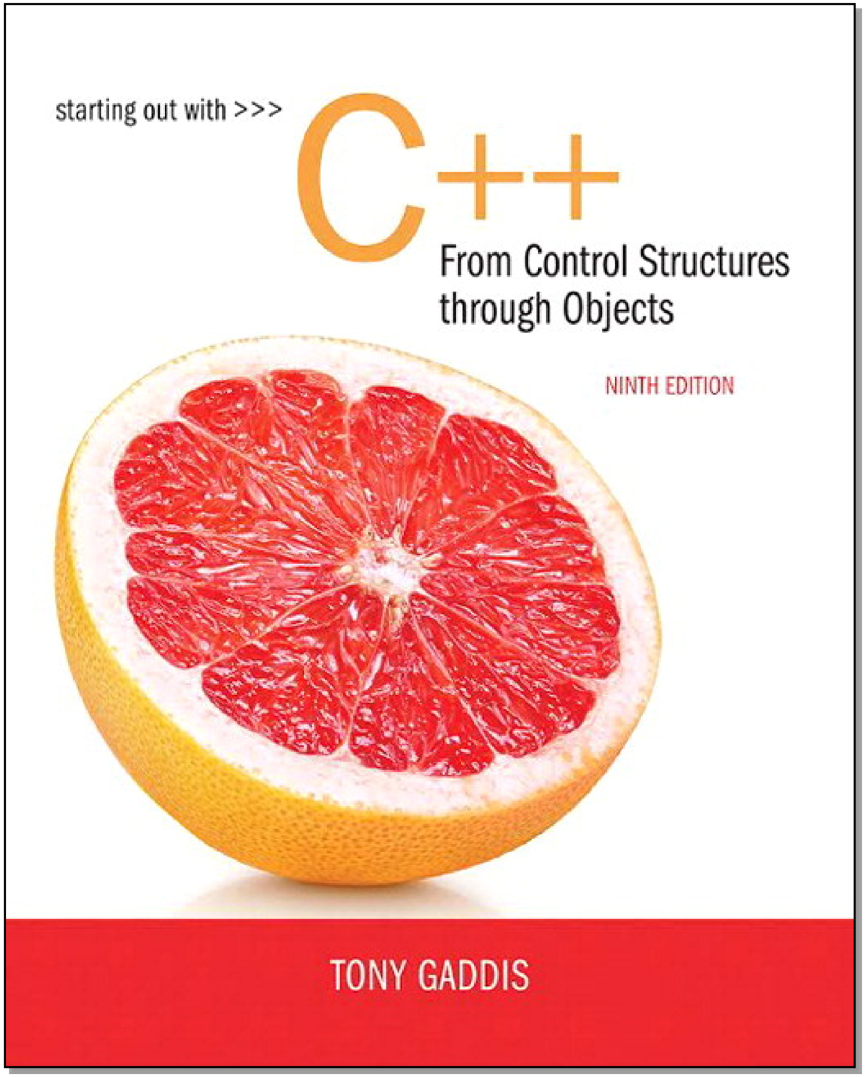
STARTING OUT WITH C++

9th Edition



Chapter 4

اتخاذ القرارات

Making Decisions

د. حسين طياوي بحبوح

4.1

العوامل العلائقية

Relational Operators

Relational Operators

- تستخدم لمقارنة الأعداد كي تحدد ترتيبها النسبي
- Operators: العوامل

>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
!=	Not equal to

Relational Expressions

- التعابير النسبية true أو false

- Examples:

  `12 > 5` is `true`

  `7 <= 5` is `false`

  if `x` is 10, then

  `x == 10` is `true`,

  `x != 8` is `true`, and

  `x == 8` is `false`

Relational Expressions

- يمكن أن تسند إلى متغير ما:

```
result = x <= y;
```

- Assigns `0` for `false`, `1` for `true`
- Do not confuse = and ==

4.2

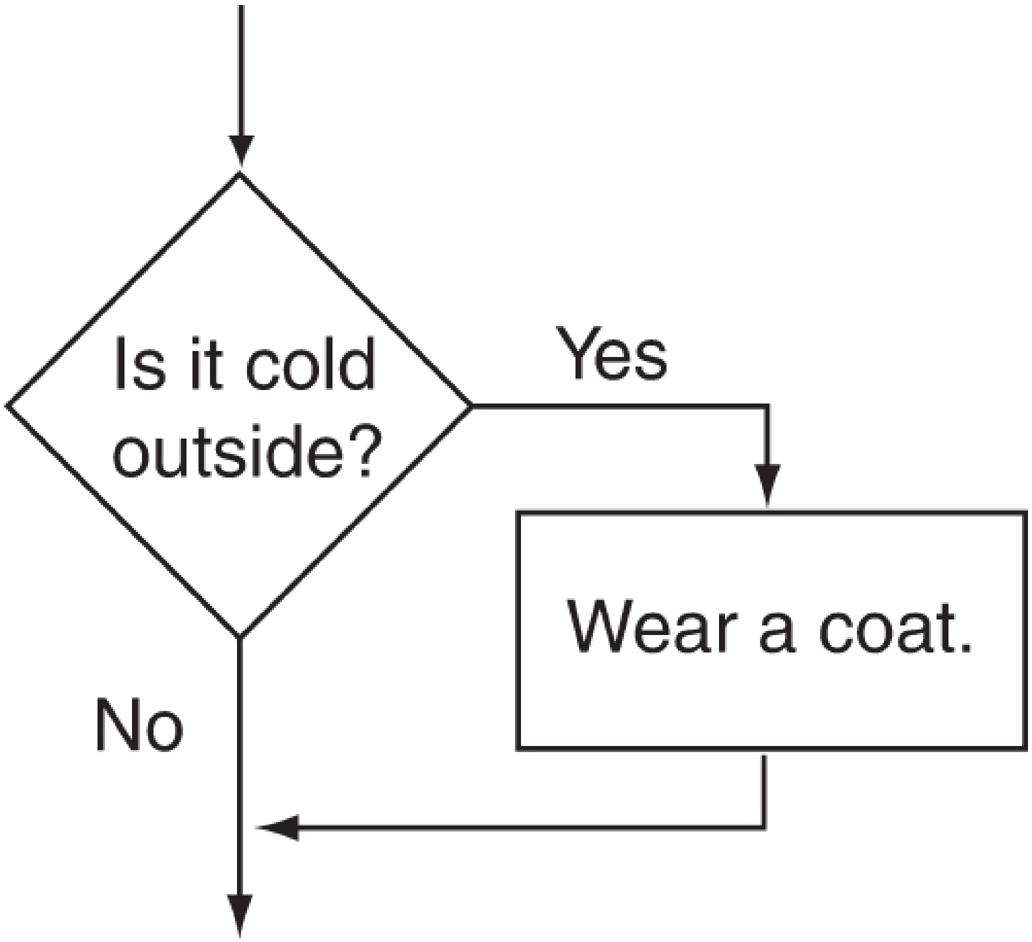
عبارة if

The `if` Statement

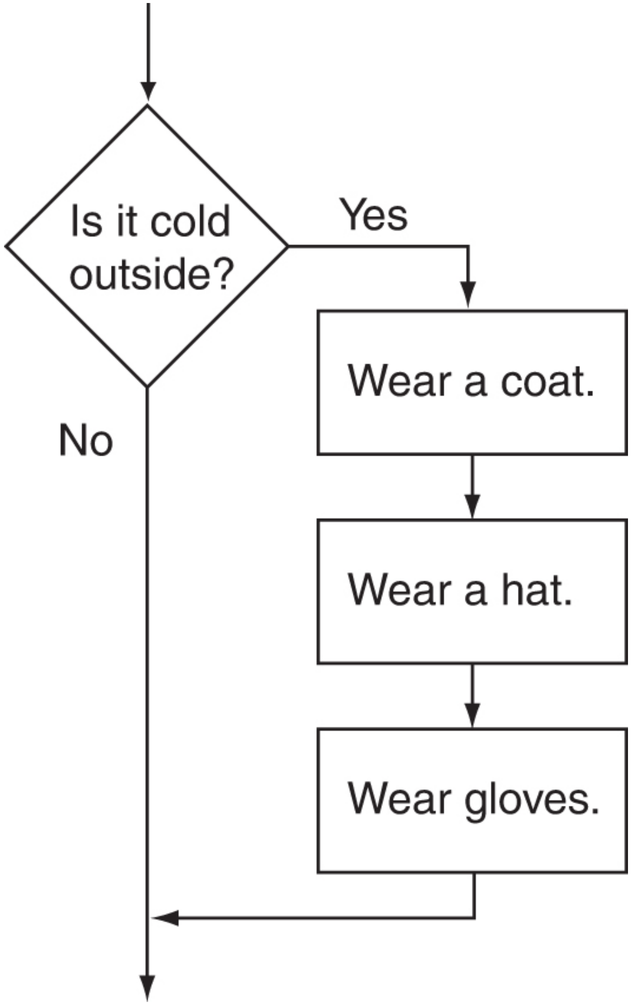
The `if` Statement

- تسمح بتنفيذ العبارات بشكل مشروط أو تجاوزها
- تقدم نموذجا مماثلا للطرق التي نقدر فيها الحلول ذهنيا:
  - "If it is raining, take an umbrella."
  - "If it is cold outside, wear a coat."

Flowchart for Evaluating a Decision



Flowchart for Evaluating a Decision



The `if` Statement

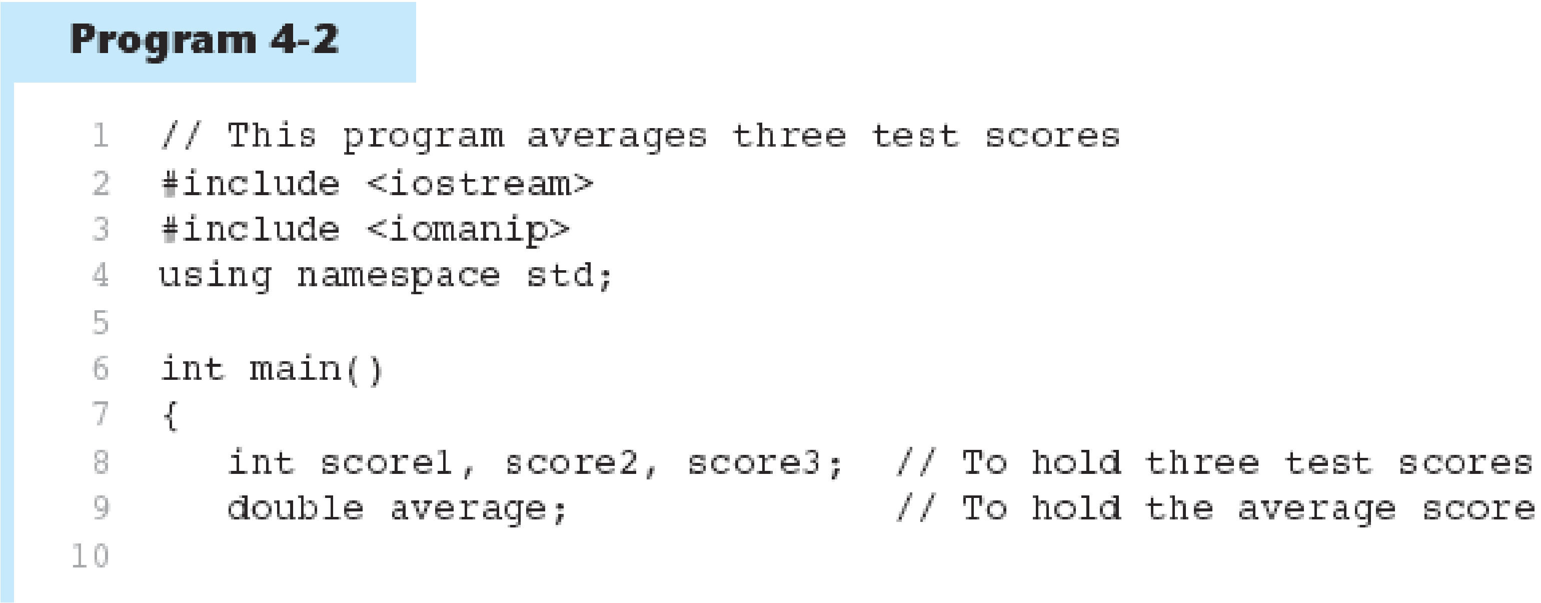
- General Format :الشكل العام

```
if (expression)
    statement;
```

لتقييمها:

- If the *expression* is `true`, then *statement* is executed.
- If the *expression* is `false`, then *statement* is skipped.

if Statement in Program 4-2



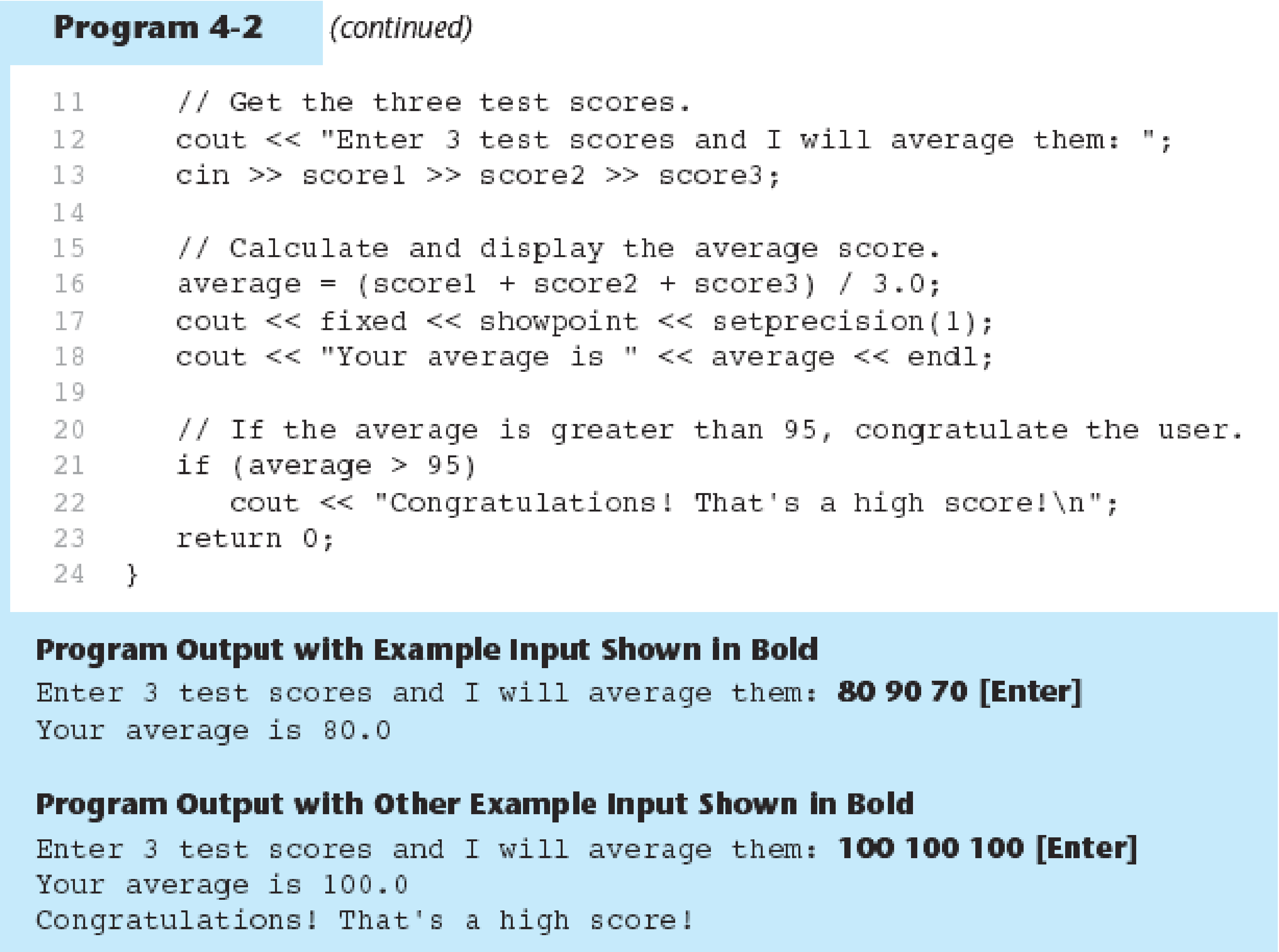
Program 4-2

```
// This program averages three test scores
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
   int score1, score2, score3;   // To hold three test scores
   double average;               // To hold the average score

```

Continued...يتبع

Pearson

if Statement in Program 4-2



Program 4-2 *(continued)*

```
    // Get the three test scores.
    cout << "Enter 3 test scores and I will average them: ";
    cin >> score1 >> score2 >> score3;

    // Calculate and display the average score.
    average = (score1 + score2 + score3) / 3.0;
    cout << fixed << showpoint << setprecision(1);
    cout << "Your average is " << average << endl;

    // If the average is greater than 95, congratulate the user.
    if (average > 95)
        cout << "Congratulations! That's a high score!\n";
    return 0;
}
```

Program Output with Example Input Shown in Bold

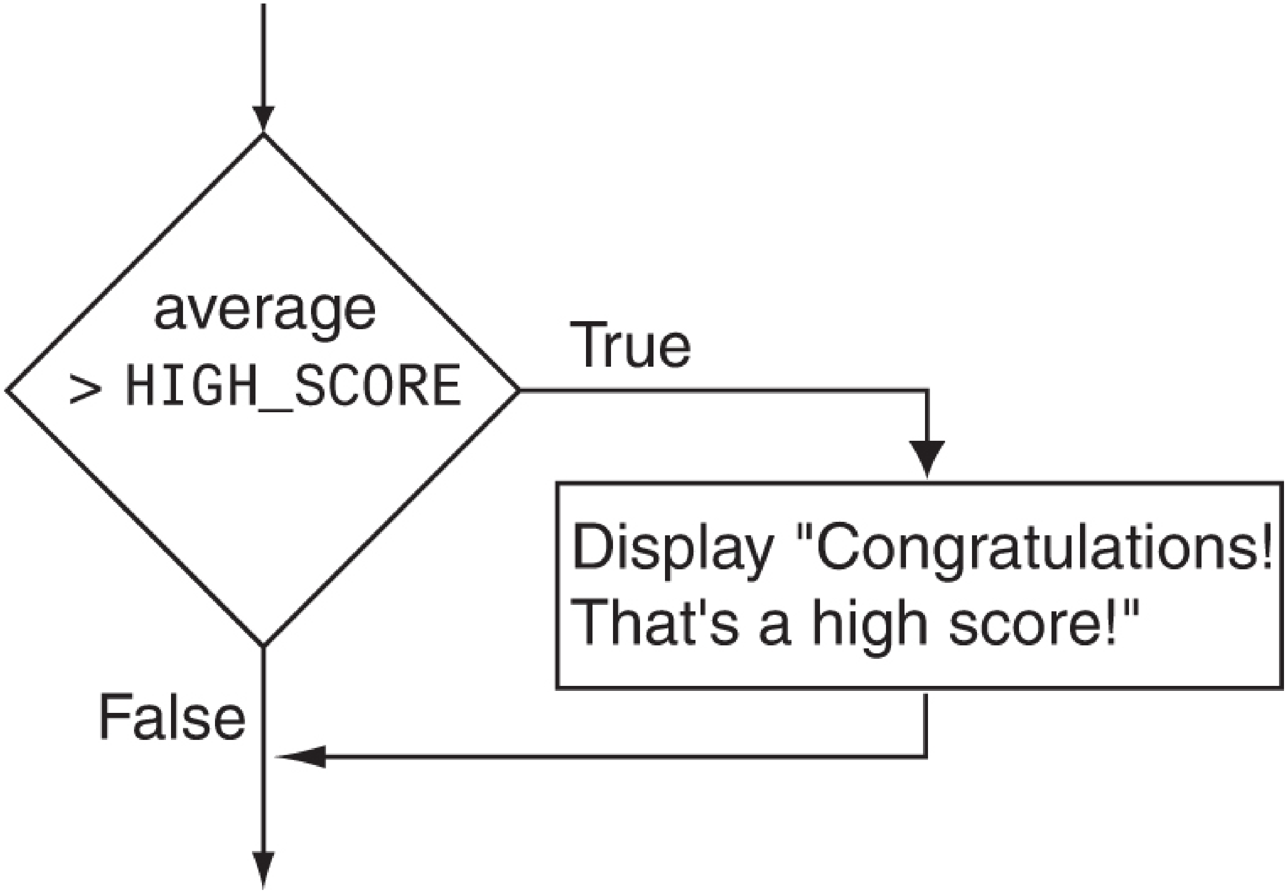
```
Enter 3 test scores and I will average them: 80 90 70 [Enter]
Your average is 80.0
```

Program Output with Other Example Input Shown in Bold

```
Enter 3 test scores and I will average them: 100 100 100 [Enter]
Your average is 100.0
Congratulations! That's a high score!
```

Pearson

Flowchart for Program 4-2 Lines 21 and 22



if Statement Notes

- لا تضع فاصلة منقوطة (;) بعد التعبير (*expression*)
- ضع *statement*; على سطر منفصل بعد (*expression*) بحيث تكون مزاحة indented.

```
if (score > 90)
    grade = 'A';
```

- كن حذرا، اختبر بالنسبة لمساواة القيم العائمة floats والـ doubles. (راجع برنامج 4-4)
- يعتبر الـ 0 خطأ false وأي قيمة أخرى صح true
- 0 is false; any other value is true

4.3

توسيع (امتداد) عبارة if

Expanding the `if` Statement

Expanding the `if` Statement

- لتنفيذ أكثر من عملية ضمن عبارة if قم بإحاطة هذه العبارات بالقوسين { }:

```
if (score > 90)
{
    grade = 'A';
    cout << "Good Job!\n";
}
```

- تنشئ الأقواس { } كتلة من الشيفرة block of code

4.4

if/else عبارة

The if/else Statement

The `if/else` statement

- تقدم مسارين ممكنين للتنفيذ.
- تنفذ عبارة واحدة أو أكثر إذا كان التعبير expression صحيحا وإلا فإنها تنفذ عبارة أو عبارات الأخرى.
- الصيغة العامة:

```
if (expression)
   statement1;  // or block
else
   statement2;  // or block
```

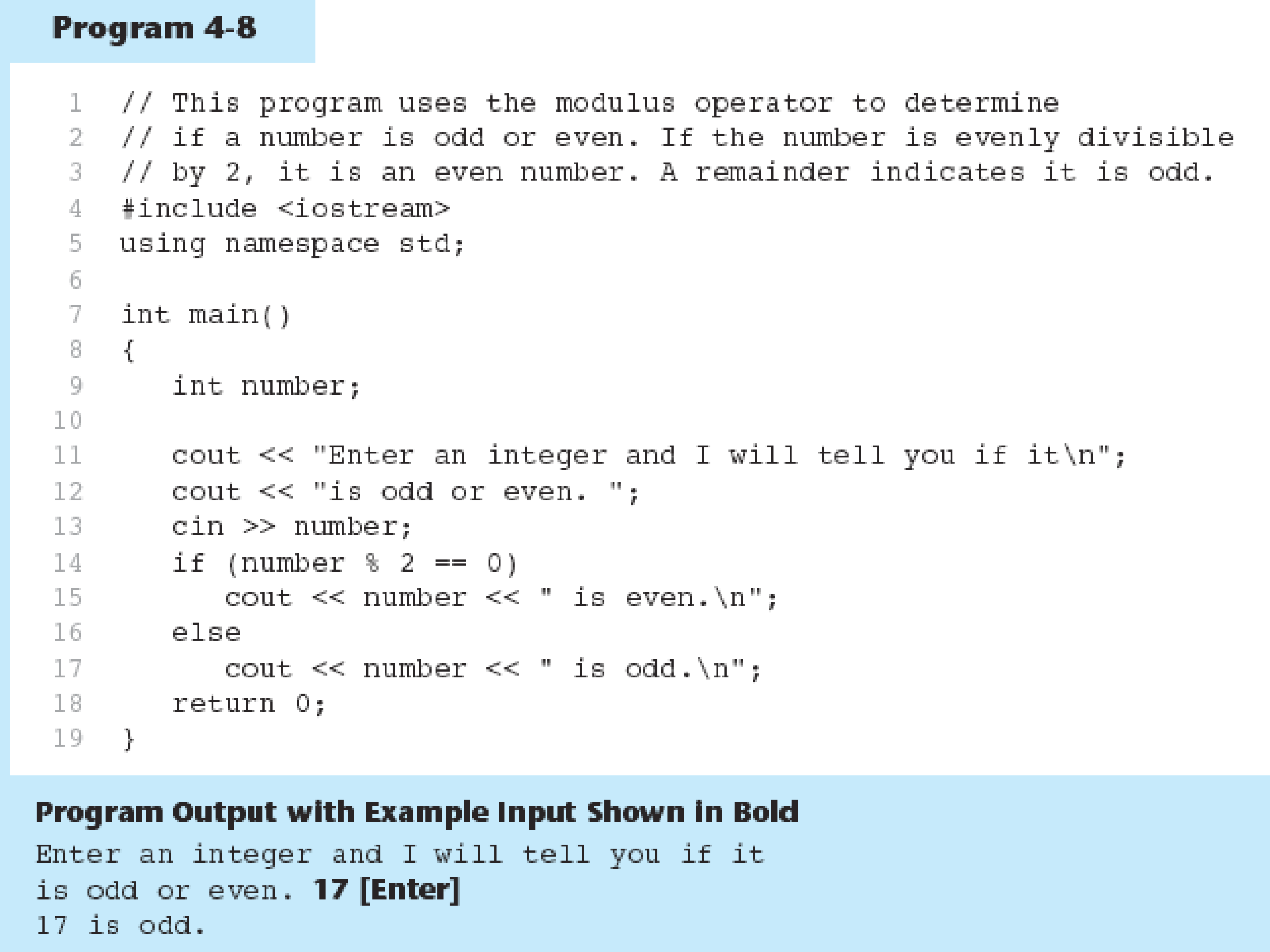
`if/else`-What Happens

لتقييم:

```
if (expression)
    statement1;
else
    statement2;
```

- If the *`expression`* is `true`, then *`statement1`* is executed and *`statement2`* is skipped.
- If the *`expression`* is *`false`*, then *`statement1`* is skipped and *`statement2`* is executed.

The if/else statement and Modulus Operator in Program 4-8



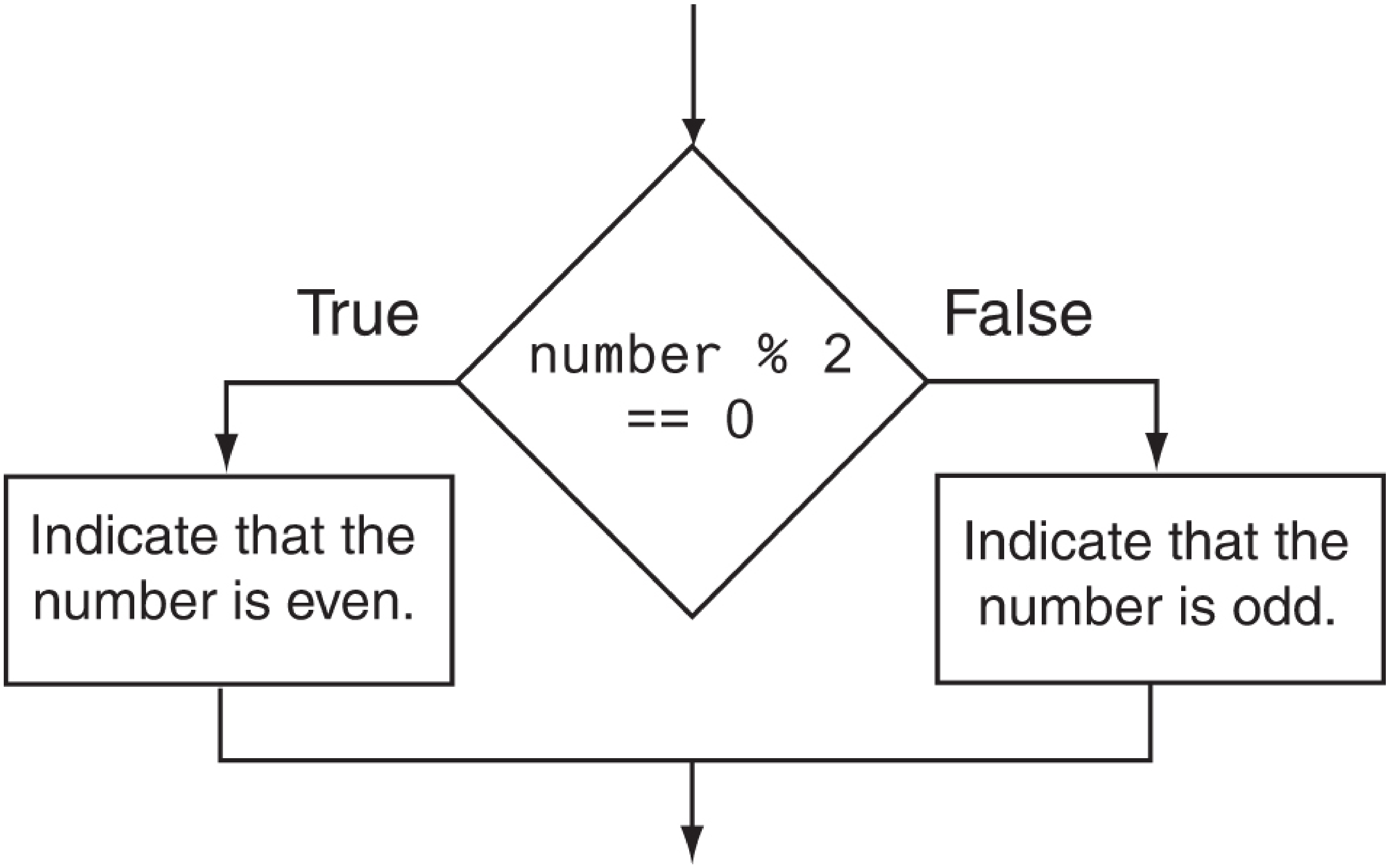
Program 4-8

```
// This program uses the modulus operator to determine
// if a number is odd or even. If the number is evenly divisible
// by 2, it is an even number. A remainder indicates it is odd.
#include <iostream>
using namespace std;

int main()
{
   int number;

   cout << "Enter an integer and I will tell you if it\n";
   cout << "is odd or even. ";
   cin >> number;
   if (number % 2 == 0)
      cout << number << " is even.\n";
   else
      cout << number << " is odd.\n";
   return 0;
}
```

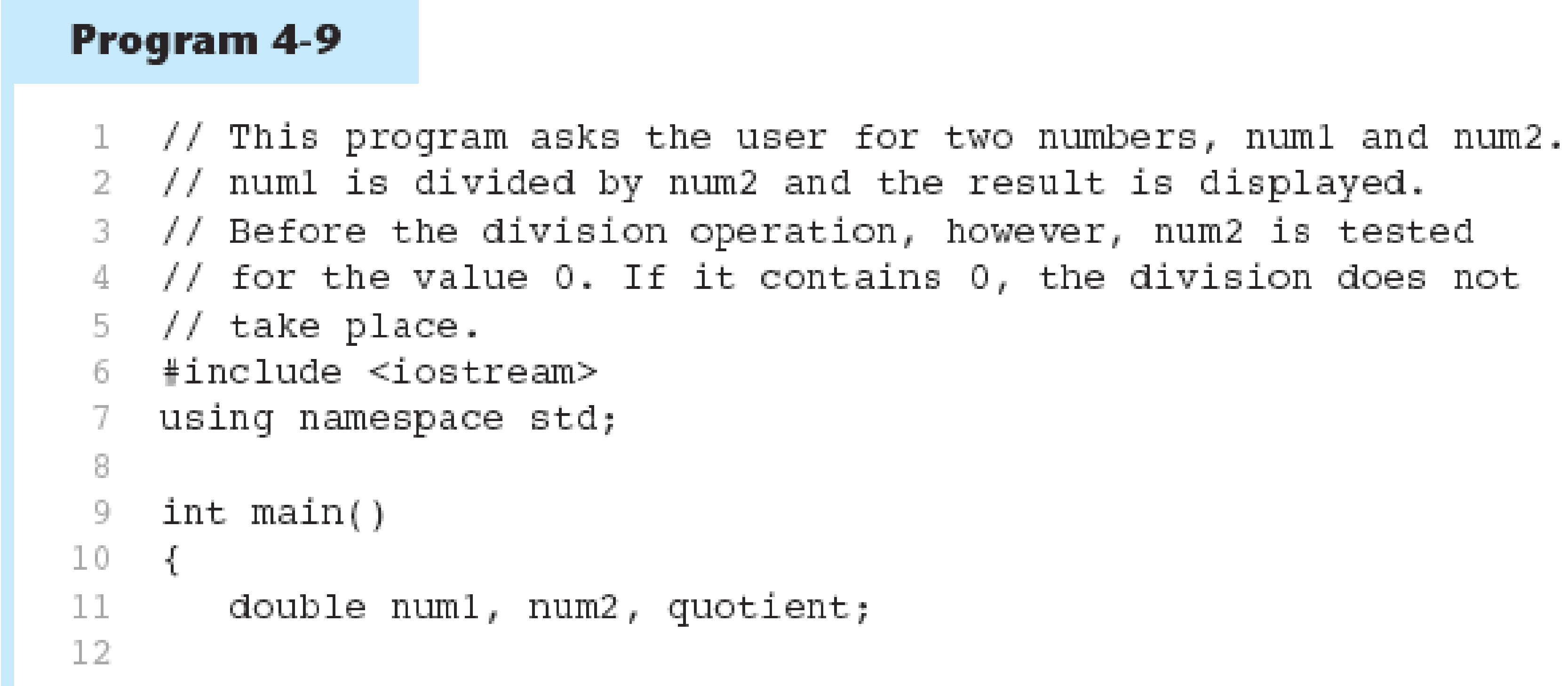
Program Output with Example Input Shown in Bold

```
Enter an integer and I will tell you if it
is odd or even. 17 [Enter]
17 is odd.
```

Flowchart for Program 4-8 Lines 14 through 18


True
number % 2 == 0
False
Indicate that the number is even.
Indicate that the number is odd.

اختبار المقسوم عليه في البرنامج 4-9
Testing the Divisor in Program 4-9

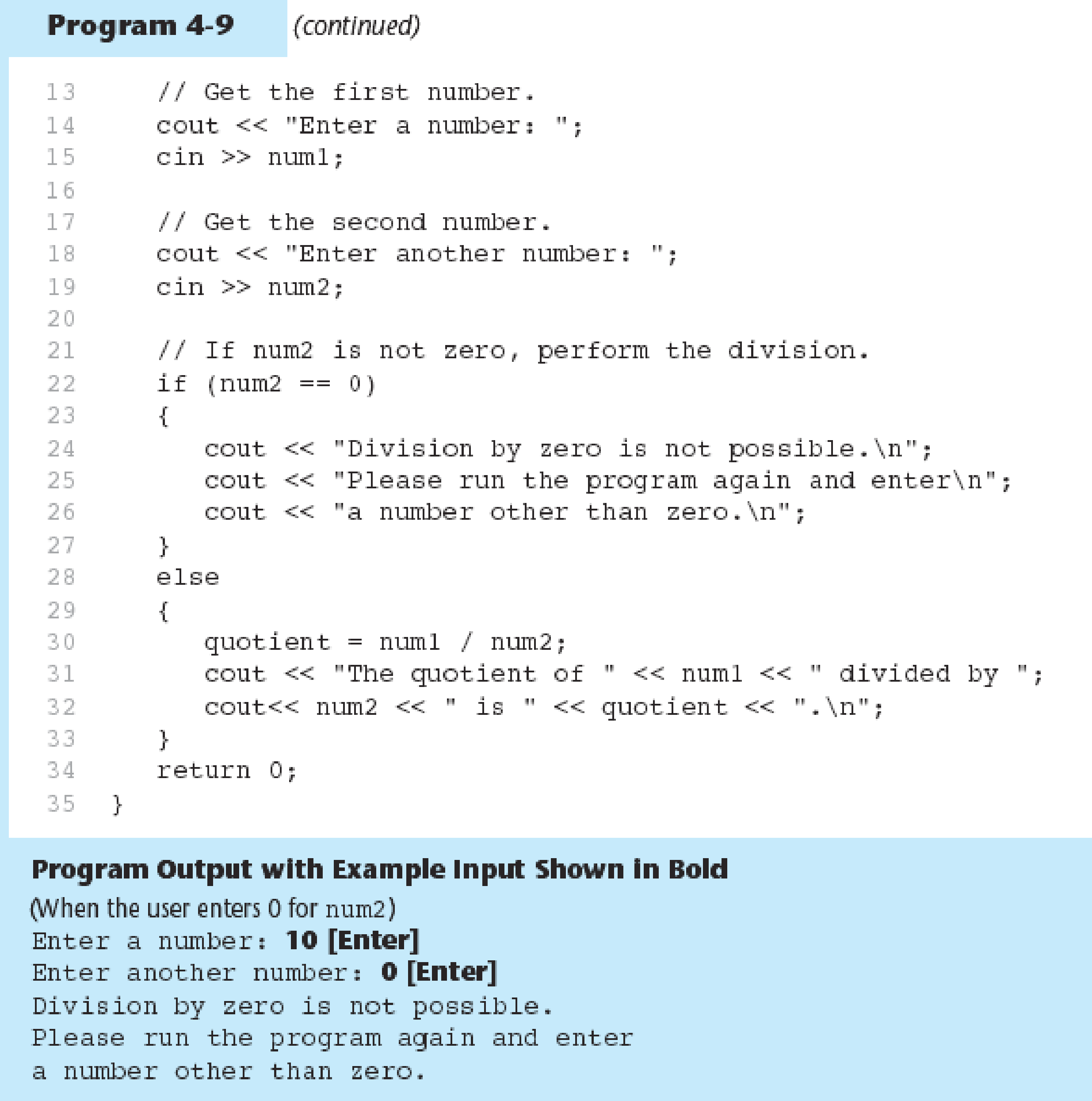


Program 4-9

```
// This program asks the user for two numbers, num1 and num2.
// num1 is divided by num2 and the result is displayed.
// Before the division operation, however, num2 is tested
// for the value 0. If it contains 0, the division does not
// take place.
#include <iostream>
using namespace std;

int main()
{
   double num1, num2, quotient;

```

Continued…

Testing the Divisor in Program 4-9



Program 4-9 *(continued)*

```cpp
    // Get the first number.
    cout << "Enter a number: ";
    cin >> num1;

    // Get the second number.
    cout << "Enter another number: ";
    cin >> num2;

    // If num2 is not zero, perform the division.
    if (num2 == 0)
    {
        cout << "Division by zero is not possible.\n";
        cout << "Please run the program again and enter\n";
        cout << "a number other than zero.\n";
    }
    else
    {
        quotient = num1 / num2;
        cout << "The quotient of " << num1 << " divided by ";
        cout<< num2 << " is " << quotient << ".\n";
    }
    return 0;
}
```

Program Output with Example Input Shown in Bold

(When the user enters 0 for num2)

```
Enter a number: 10 [Enter]
Enter another number: 0 [Enter]
Division by zero is not possible.
Please run the program again and enter
a number other than zero.
```

4.5

عبارات if المتداخلة

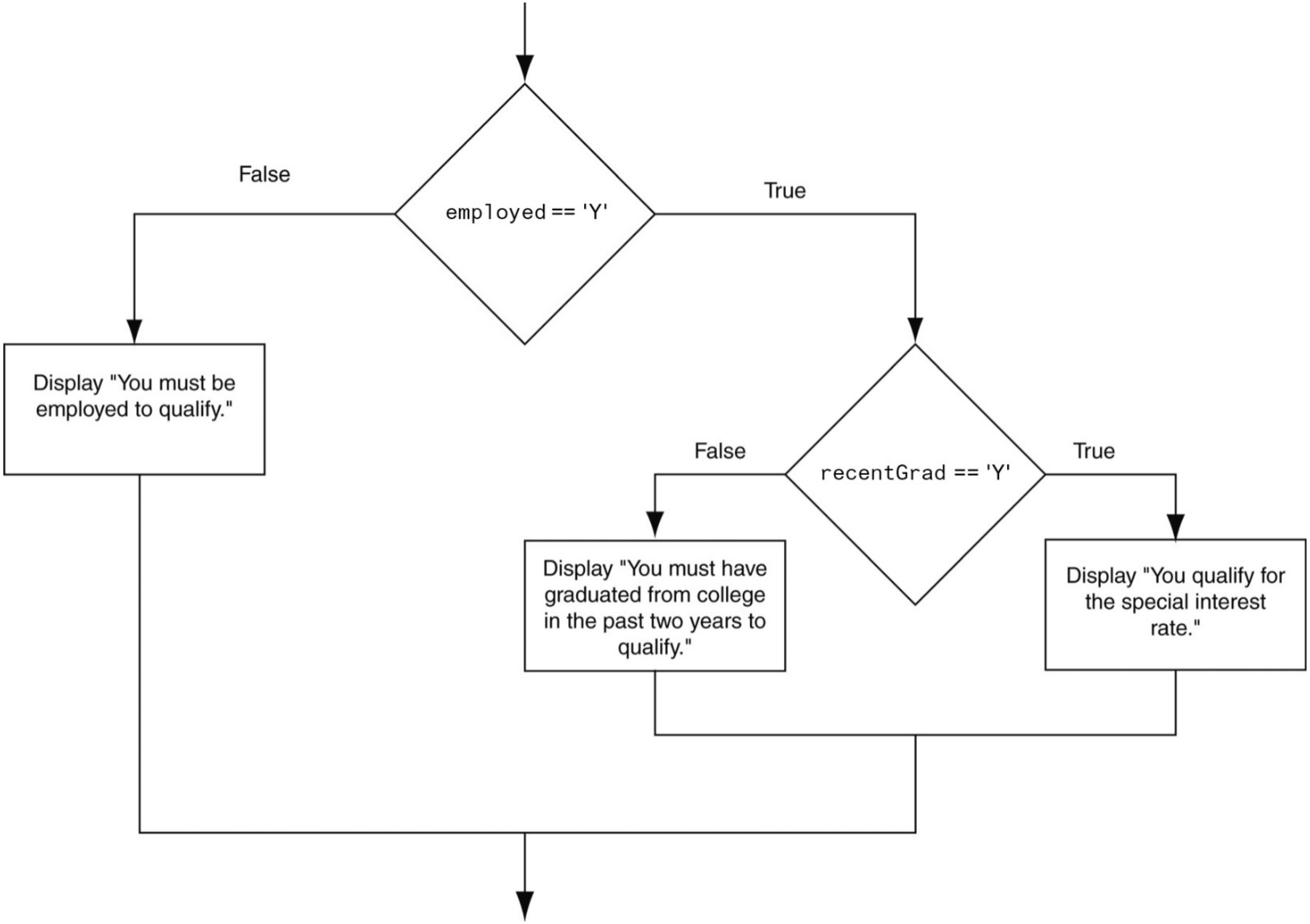
Nested if Statements

Nested `if` Statements

- هي عبارة `if` المتداخلة ضمن عبارة `if` أخرى.
- يمكن استخدام عبارات if المتداخلة لاختبار أكثر من شرط واحد.

المخطط الصندوقي لعبارة if المتداخلة (فقرة 4-5)

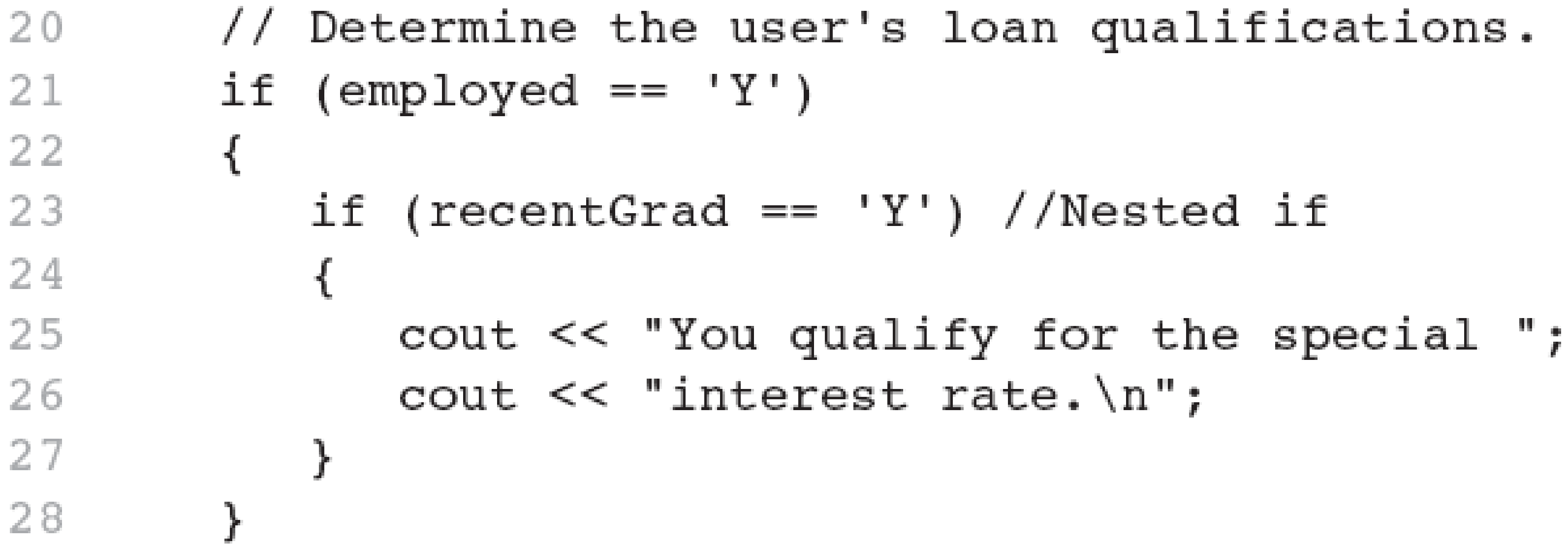
Flowchart for a Nested `if` Statement



False
employed == 'Y'
True
Display "You must be employed to qualify."
False
recentGrad == 'Y'
True
Display "You must have graduated from college in the past two years to qualify."
Display "You qualify for the special interest rate."

Nested `if` Statements

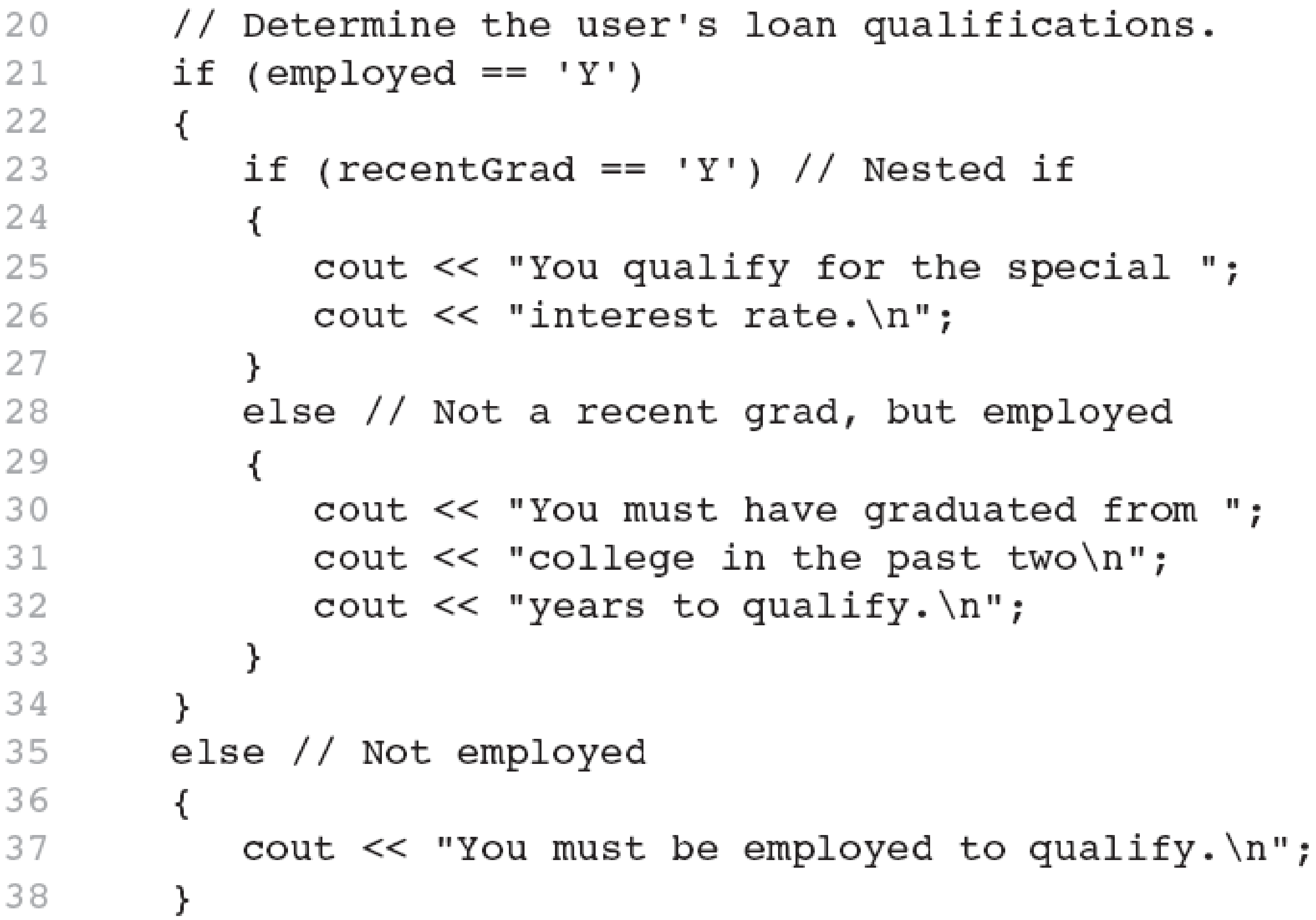
- From Program 4-10



```
// Determine the user's loan qualifications.
if (employed == 'Y')
{
   if (recentGrad == 'Y') //Nested if
   {
      cout << "You qualify for the special ";
      cout << "interest rate.\n";
   }
}
```

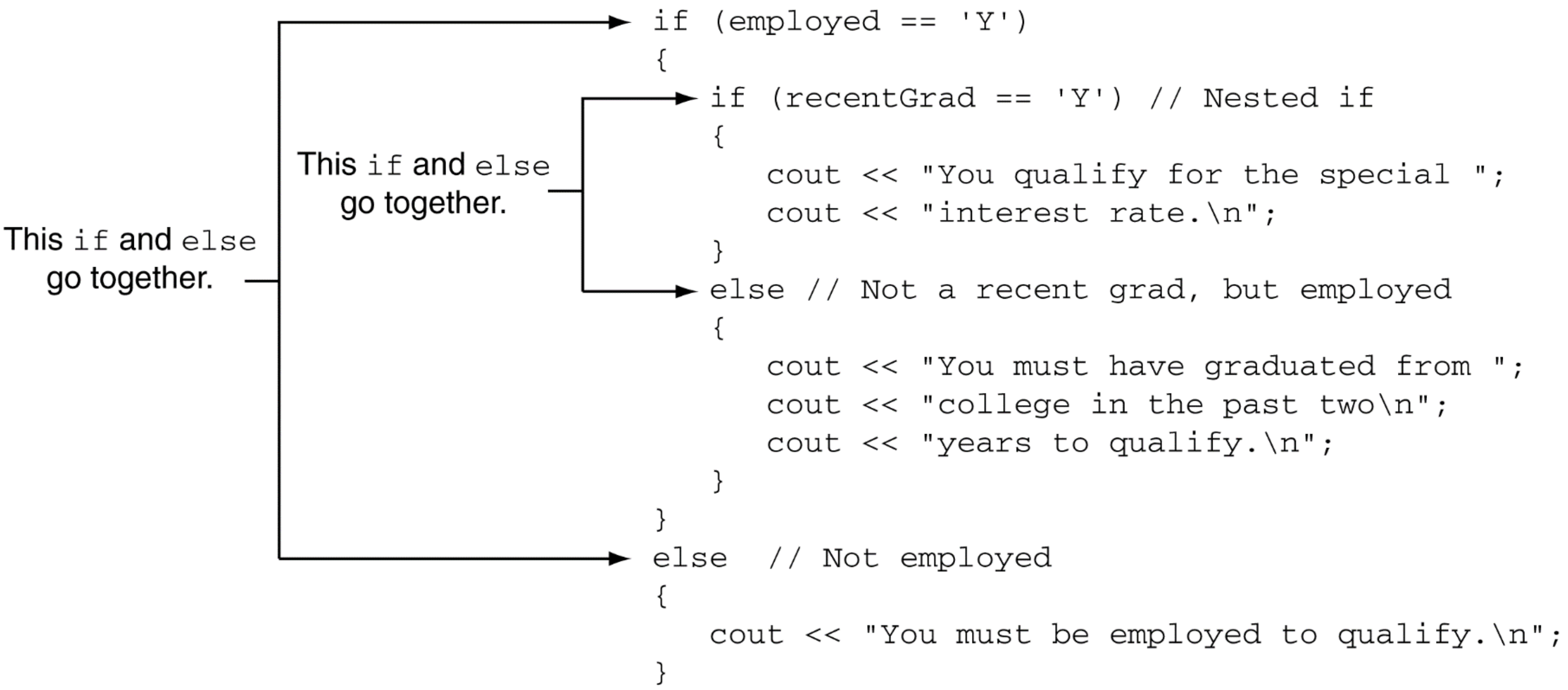
Nested `if` Statements

- Another example, from Program 4-1



```
    // Determine the user's loan qualifications.
    if (employed == 'Y')
    {
       if (recentGrad == 'Y') // Nested if
       {
          cout << "You qualify for the special ";
          cout << "interest rate.\n";
       }
       else // Not a recent grad, but employed
       {
          cout << "You must have graduated from ";
          cout << "college in the past two\n";
          cout << "years to qualify.\n";
       }
    }
    else // Not employed
    {
       cout << "You must be employed to qualify.\n";
    }
```

Use Proper Indentation!



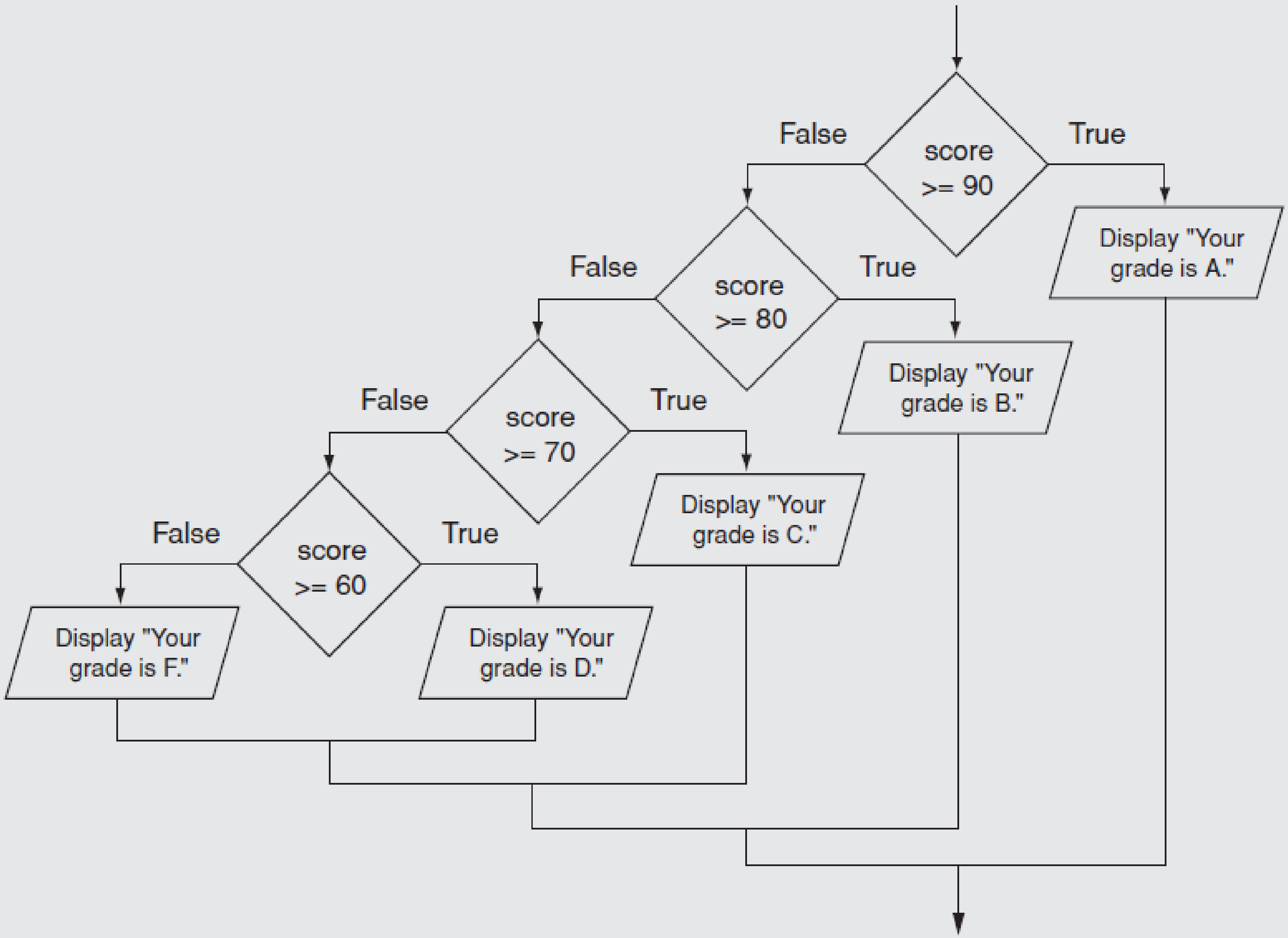
This if and else go together.

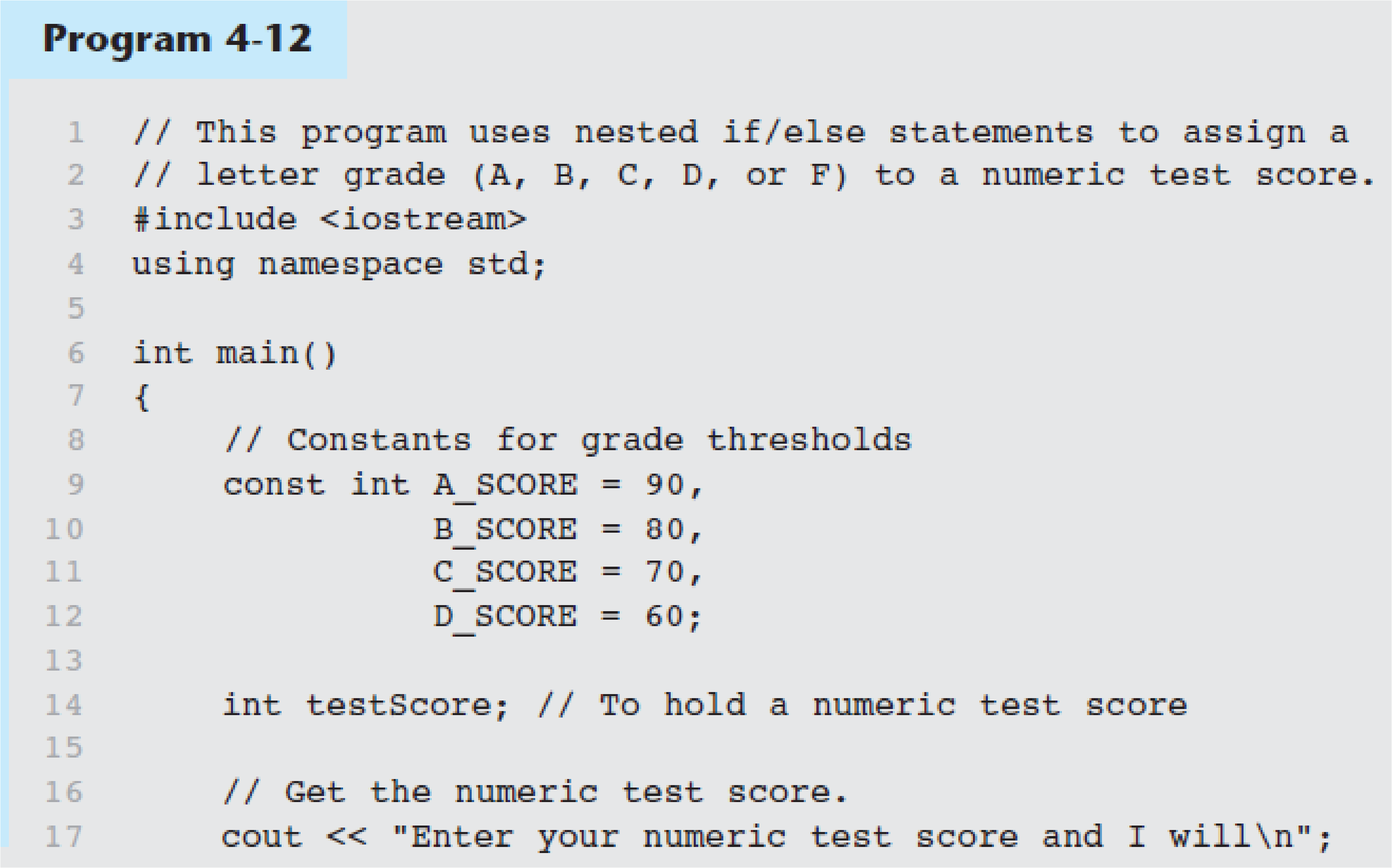
This if and else go together.

```
if (employed == 'Y')
{
   if (recentGrad == 'Y') // Nested if
   {
      cout << "You qualify for the special ";
      cout << "interest rate.\n";
   }
   else // Not a recent grad, but employed
   {
      cout << "You must have graduated from ";
      cout << "college in the past two\n";
      cout << "years to qualify.\n";
   }
}
else  // Not employed
{
   cout << "You must be employed to qualify.\n";
}
```

مثال أعم

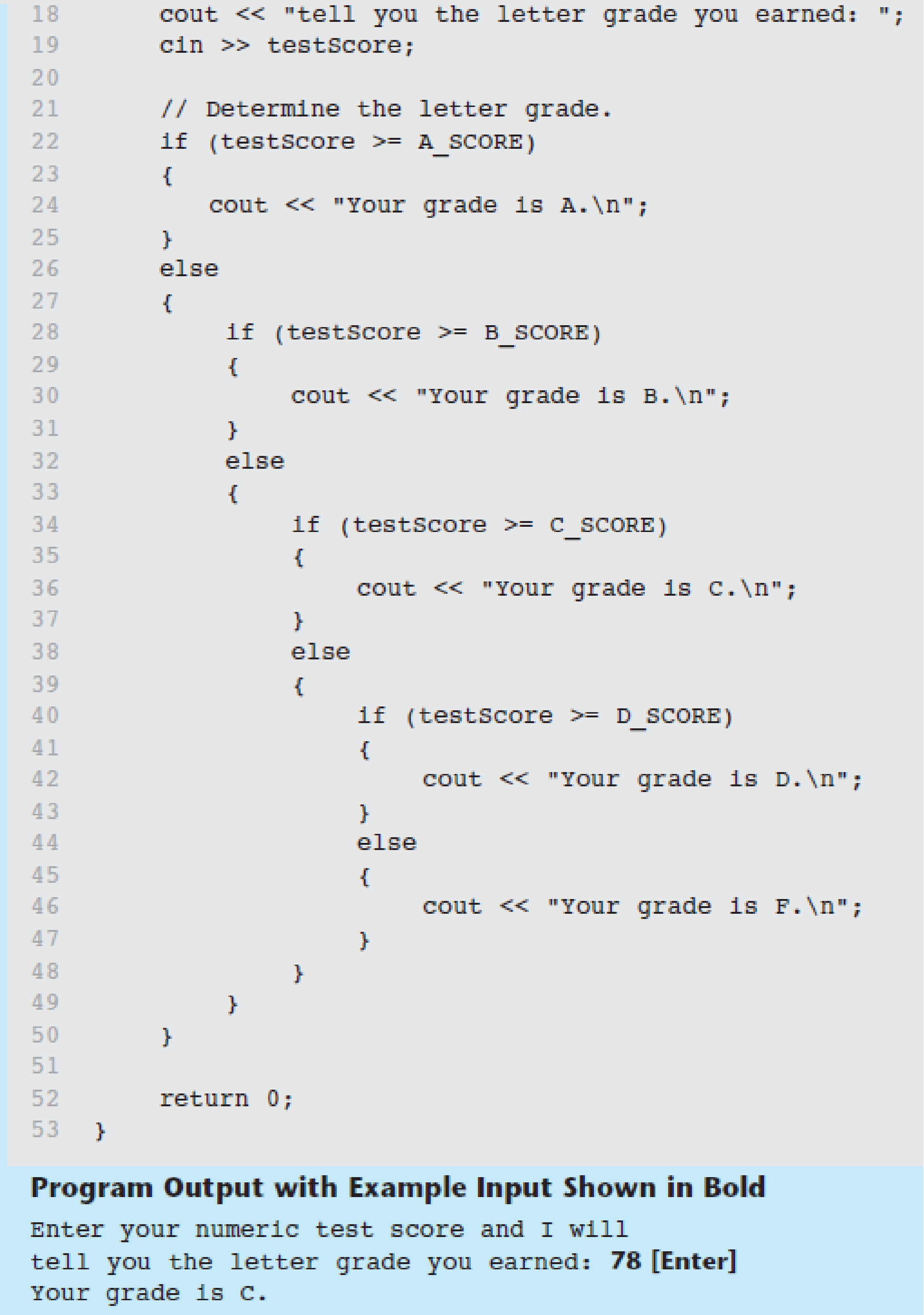
Figure 4-9 Nested decision structure to determine a grade


score >= 90
False
True
Display "Your grade is A."
score >= 80
False
True
Display "Your grade is B."
score >= 70
False
True
Display "Your grade is C."
score >= 60
False
True
Display "Your grade is F."
Display "Your grade is D."



Program 4-12

```
// This program uses nested if/else statements to assign a
// letter grade (A, B, C, D, or F) to a numeric test score.
#include <iostream>
using namespace std;

int main()
{
    // Constants for grade thresholds
    const int A_SCORE = 90,
              B_SCORE = 80,
              C_SCORE = 70,
              D_SCORE = 60;

    int testScore; // To hold a numeric test score

    // Get the numeric test score.
    cout << "Enter your numeric test score and I will\n";
```



```
    cout << "tell you the letter grade you earned: ";
    cin >> testScore;

    // Determine the letter grade.
    if (testScore >= A_SCORE)
    {
       cout << "Your grade is A.\n";
    }
    else
    {
        if (testScore >= B_SCORE)
        {
            cout << "Your grade is B.\n";
        }
        else
        {
            if (testScore >= C_SCORE)
            {
                cout << "Your grade is C.\n";
            }
            else
            {
                if (testScore >= D_SCORE)
                {
                    cout << "Your grade is D.\n";
                }
                else
                {
                    cout << "Your grade is F.\n";
                }
            }
        }
    }

    return 0;
}
```

Program Output with Example Input Shown in Bold

```
Enter your numeric test score and I will
tell you the letter grade you earned: 78 [Enter]
Your grade is C.
```

4.6

if/else if عبارة

The if/else if Statement

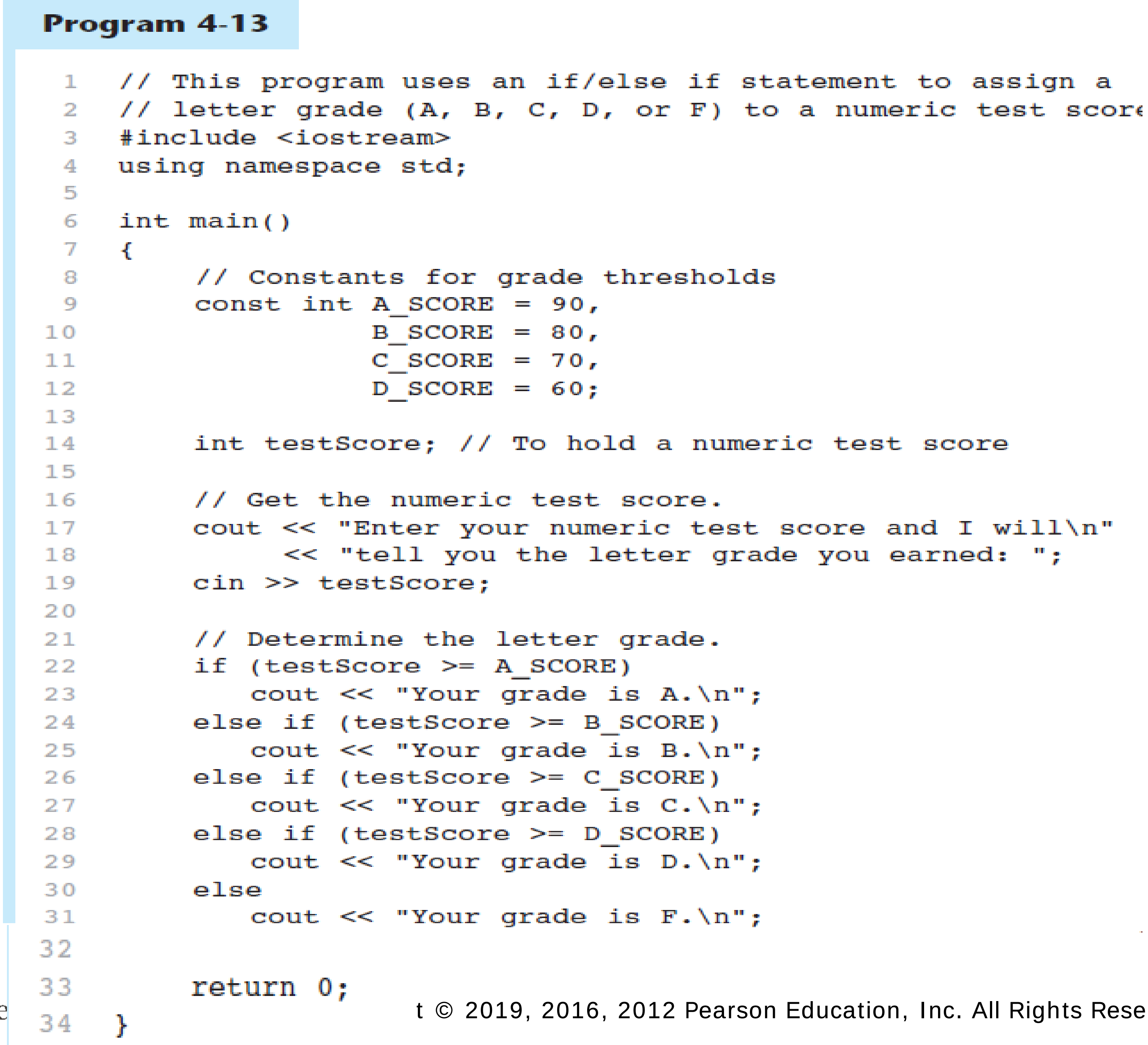
The `if/else if` Statement

- تختبر سلسلة من الشروط حتى يتحقق واحدا منها (يصبح true).
- هي في الغالب أبسط من استخدام عبارات `if/else` المتداخلة.
- يمكن استخدامها كي تنمذج عمليات التفكير thought processes مثل:

"If it is raining, take an umbrella,
else, if it is windy, take a hat,
else, take sunglasses”

الشكل العام لعبارة if/else if

```
if (expression)
     statement1;  // or block
else if (expression)
     statement2;  // or block
     .
     . // other else ifs
    .
else if (expression)
     statementn;  // or block
```



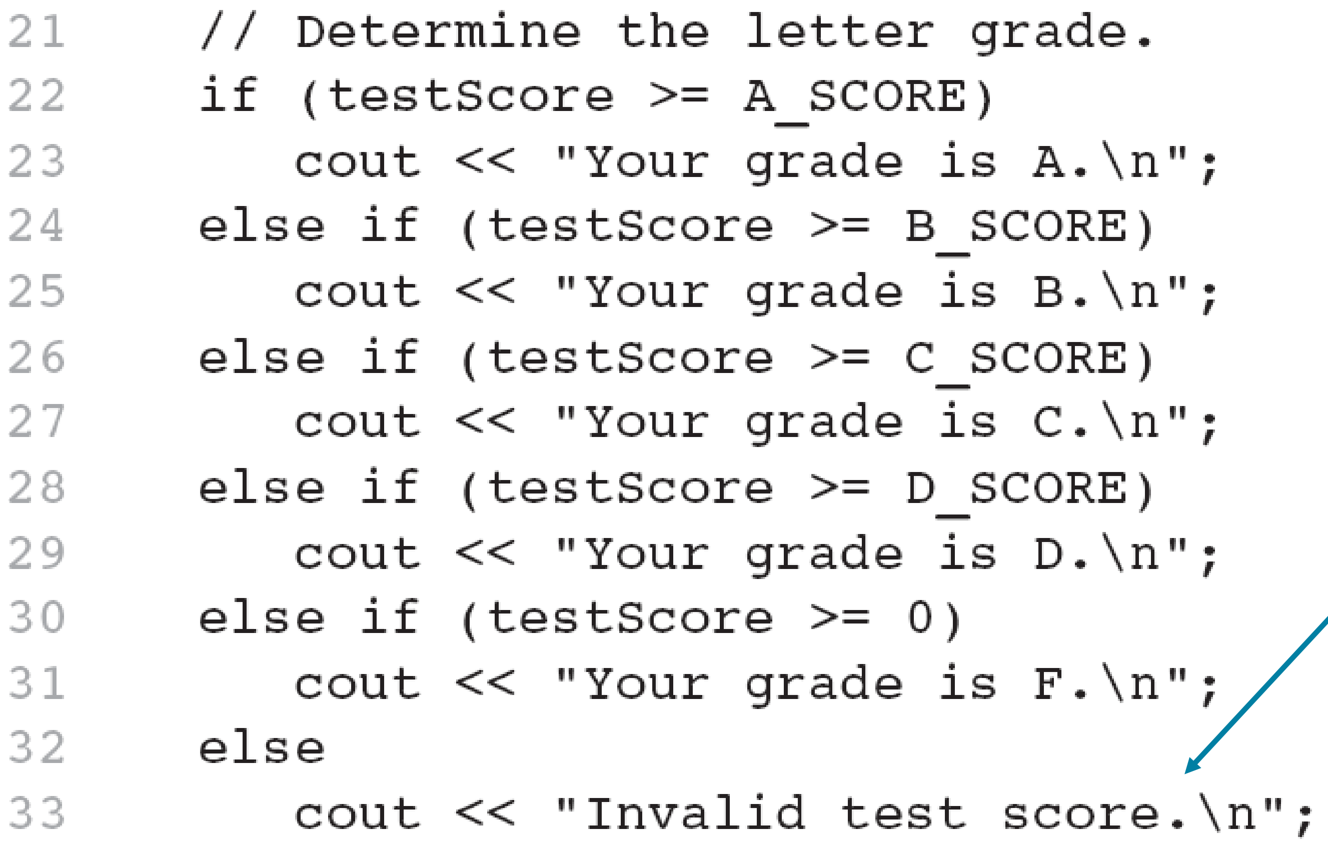
Program 4-13

```
// This program uses an if/else if statement to assign a
// letter grade (A, B, C, D, or F) to a numeric test scor
#include <iostream>
using namespace std;

int main()
{
     // Constants for grade thresholds
     const int A_SCORE = 90,
               B_SCORE = 80,
               C_SCORE = 70,
               D_SCORE = 60;

     int testScore; // To hold a numeric test score

     // Get the numeric test score.
     cout << "Enter your numeric test score and I will\n"
          << "tell you the letter grade you earned: ";
     cin >> testScore;

     // Determine the letter grade.
     if (testScore >= A_SCORE)
        cout << "Your grade is A.\n";
     else if (testScore >= B_SCORE)
        cout << "Your grade is B.\n";
     else if (testScore >= C_SCORE)
        cout << "Your grade is C.\n";
     else if (testScore >= D_SCORE)
        cout << "Your grade is D.\n";
     else
        cout << "Your grade is F.\n";

     return 0;
}
```

استخدام else أخيرة لمسك الأخطاء

Using a Trailing `else` to Catch Errors in Program 4-14

- تعتبر عبارة else الأخيرة اختيارية، لكنها تستخدم لكشف الأخطاء.



```
// Determine the letter grade.
if (testScore >= A_SCORE)
   cout << "Your grade is A.\n";
else if (testScore >= B_SCORE)
   cout << "Your grade is B.\n";
else if (testScore >= C_SCORE)
   cout << "Your grade is C.\n";
else if (testScore >= D_SCORE)
   cout << "Your grade is D.\n";
else if (testScore >= 0)
   cout << "Your grade is F.\n";
else
   cout << "Invalid test score.\n";
```

This trailing `else` catches invalid test scores

4.7

Flags (الأعلام) الرايات

4.8

العوامل (المؤثرات) المنطقية

Logical Operators

Logical Operators

- تستخدم لإنشاء تعابير علائقية من تعابير علائقية أخرى.
- العوامل، معناها وتوضيحها:

&&	AND	يكون التعبير النسبي الجديد صحيحا إذا كان كلا من التعبيرين على جانبي العامل صحيحا
||	OR	يكون التعبير النسبي الجديد صحيحا إذا كان أيا او كلا من التعبيرين على جانبي العامل صحيحا
!	NOT	تعكس قيمة تعبير ما- من صح إلى خطأ وبالعكس

Pearson

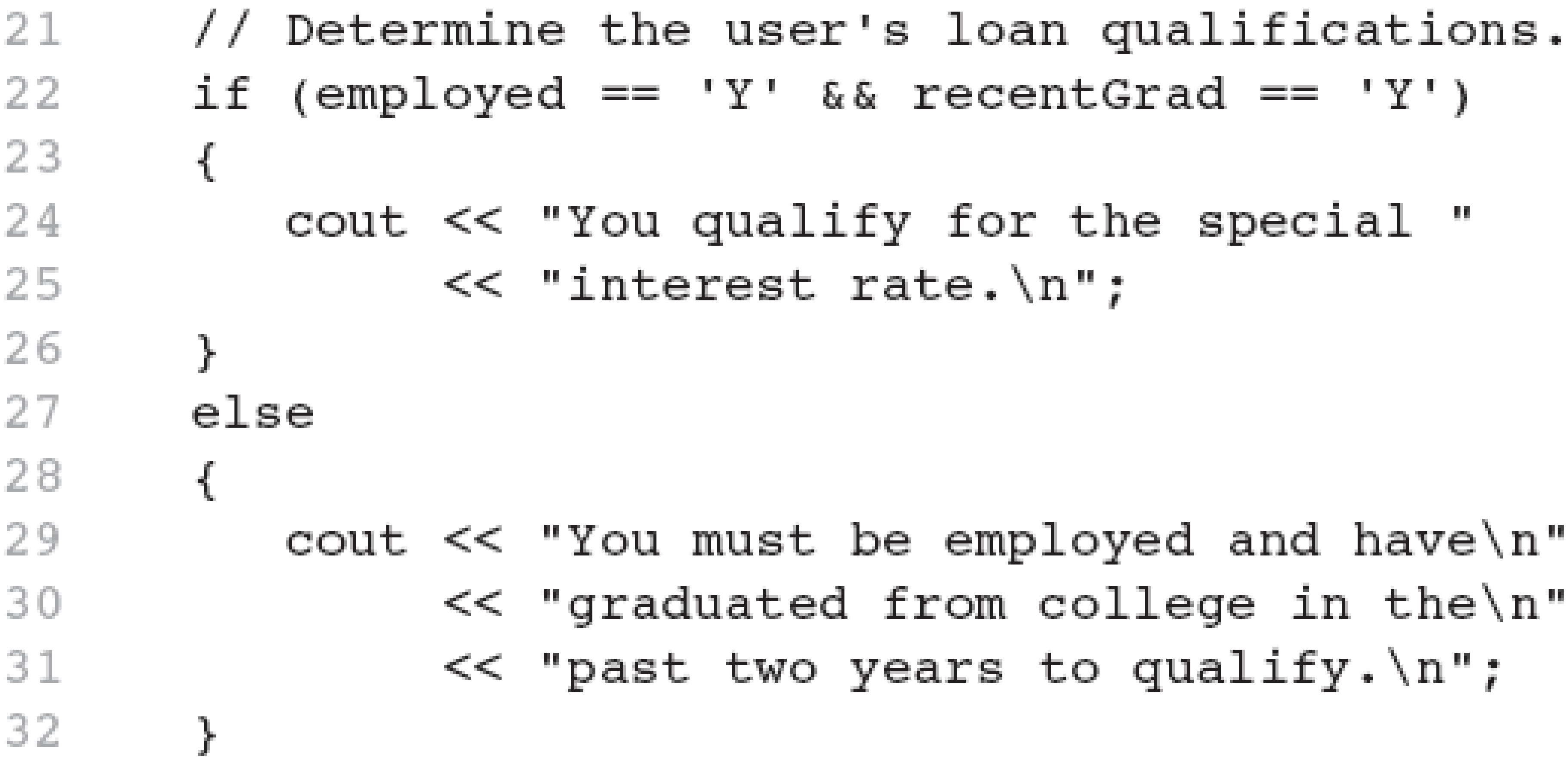
أمثلة عن العوامل (المؤثرات) المنطقية

Logical Operators-Examples

```
int x = 12, y = 5, z = -4;
```

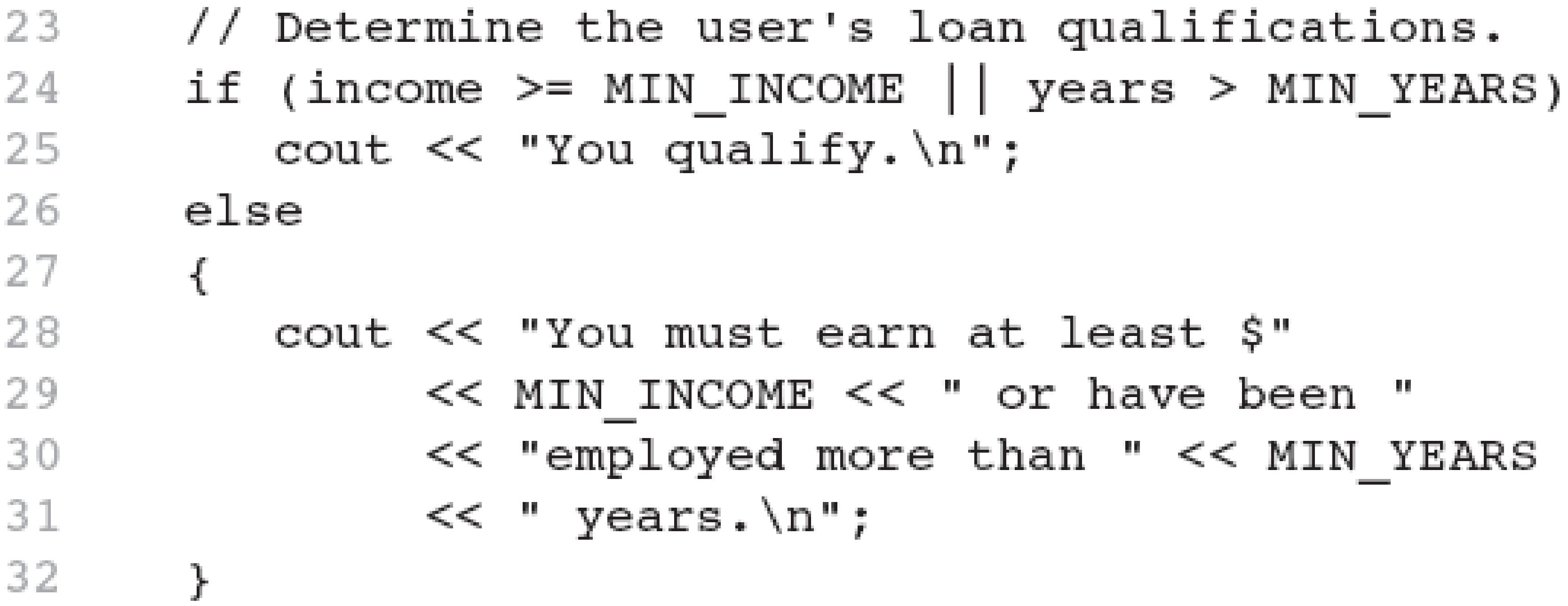
`(x > y) && (y > z)`	`true`
`(x > y) && (z > y)`	`false`
`(x <= z) || (y == z)`	`false`
`(x <= z) || (y != z)`	`true`
`!(x >= z)`	`false`

The logical && operator in Program 4-15



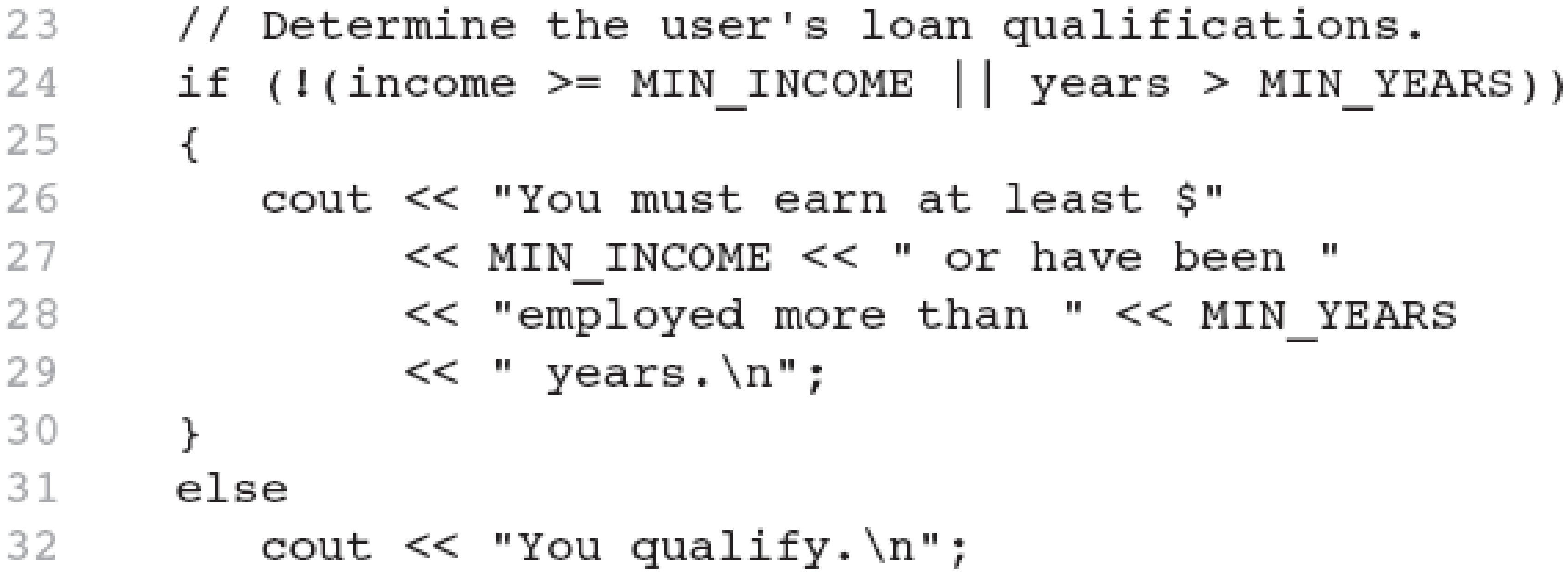
```
    // Determine the user's loan qualifications.
    if (employed == 'Y' && recentGrad == 'Y')
    {
       cout << "You qualify for the special "
            << "interest rate.\n";
    }
    else
    {
       cout << "You must be employed and have\n"
            << "graduated from college in the\n"
            << "past two years to qualify.\n";
    }
```

The logical || Operator in Program 4-16



```
    // Determine the user's loan qualifications.
    if (income >= MIN_INCOME || years > MIN_YEARS)
       cout << "You qualify.\n";
    else
    {
       cout << "You must earn at least $"
            << MIN_INCOME << " or have been "
            << "employed more than " << MIN_YEARS
            << " years.\n";
    }
```

The logical ! Operator in Program 4-17



```
   // Determine the user's loan qualifications.
   if (!(income >= MIN_INCOME || years > MIN_YEARS))
   {
      cout << "You must earn at least $"
           << MIN_INCOME << " or have been "
           << "employed more than " << MIN_YEARS
           << " years.\n";
   }
   else
      cout << "You qualify.\n";
```

Logical Operator-Notes

- ! has highest precedence, followed by &&, then | |
- If the value of an expression can be determined by evaluating just the sub-expression on left side of a logical operator, then the sub-expression on the right side will not be evaluated (*short circuit evaluation*)

4.9

اختبار المجالات العددية بالمعاملات المنطقية

Checking Numeric Ranges with Logical Operators

اختبار المجالات العددية بالمعاملات المنطقية
Checking Numeric Ranges with Logical Operators

- تستخدم لاختبار ما إذا كانت قيمة ما تقع ضمن مجال ما:

```
if (grade >= 0 && grade <= 100)
    cout << "Valid grade";
```

- يمكن أيضا أن تستخدم لاختبار ما إذا كانت قيمة ما تقع خارج مجال ما:

```
if (grade <= 0 || grade >= 100)
    cout << "Invalid grade";
```

- لا يمكن أن تستخدم الترميز الرياضي التالي كعبارة برمجية:

```
if (0 <= grade <= 100) //doesn’t work!
```

Pearson

4.10

القوائم

Menus

Menus

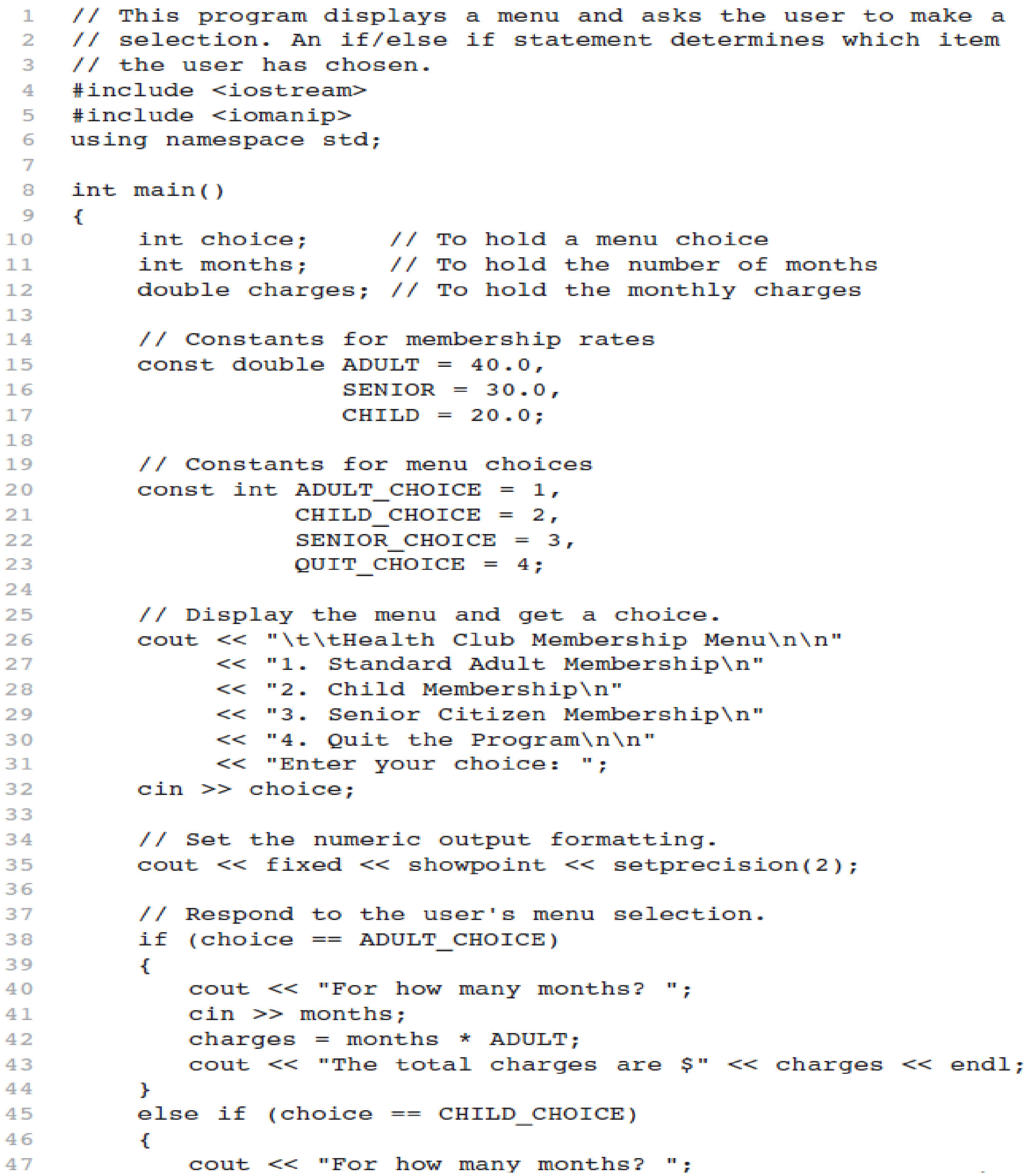
البرنامج المقاد بالقوائم: تعني أن التحكم بتنفيذ برنامجا ما يتم عبر انتقاء حدث ما من قائمة.

القائمة: هي مجموعة اختيارات تظهر كقائمة على الشاشة.

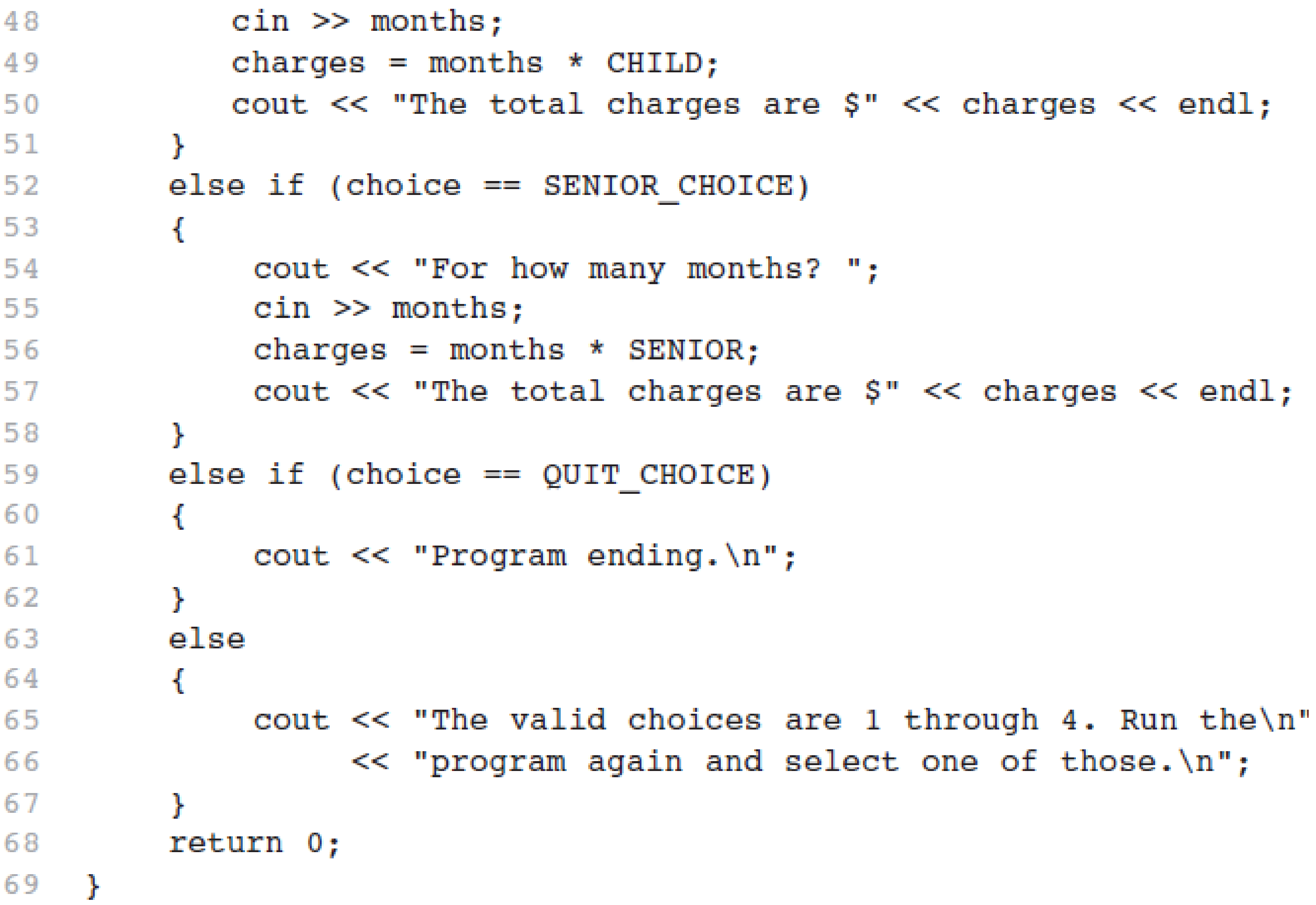
يمكن تنفيذ القوائم باستخدام عبارات if/else if.

تنظيم البرنامج المقاد بالقوائم
Menu-Driven Program Organization

- يظهر قائمة اختيارات أفعال (أحداث) مرقمة بأعداد أو أحرف.
- يجب حث المستخدم على إجراء أحد الاختيارات.
- يتم اختبار انتقاء المستخدم في التعبير:
  - إذا ناسب يتم تنفيذ الفعل
  - إذا لم يناسب ينتقل للتعبير الثاني



```cpp
// This program displays a menu and asks the user to make a
// selection. An if/else if statement determines which item
// the user has chosen.
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
    int choice;      // To hold a menu choice
    int months;      // To hold the number of months
    double charges; // To hold the monthly charges

    // Constants for membership rates
    const double ADULT = 40.0,
                 SENIOR = 30.0,
                 CHILD = 20.0;

    // Constants for menu choices
    const int ADULT_CHOICE = 1,
              CHILD_CHOICE = 2,
              SENIOR_CHOICE = 3,
              QUIT_CHOICE = 4;

    // Display the menu and get a choice.
    cout << "\t\tHealth Club Membership Menu\n\n"
         << "1. Standard Adult Membership\n"
         << "2. Child Membership\n"
         << "3. Senior Citizen Membership\n"
         << "4. Quit the Program\n\n"
         << "Enter your choice: ";
    cin >> choice;

    // Set the numeric output formatting.
    cout << fixed << showpoint << setprecision(2);

    // Respond to the user's menu selection.
    if (choice == ADULT_CHOICE)
    {
        cout << "For how many months? ";
        cin >> months;
        charges = months * ADULT;
        cout << "The total charges are $" << charges << endl;
    }
    else if (choice == CHILD_CHOICE)
    {
        cout << "For how many months? ";
```



```
       cin >> months;
       charges = months * CHILD;
       cout << "The total charges are $" << charges << endl;
    }
    else if (choice == SENIOR_CHOICE)
    {
        cout << "For how many months? ";
        cin >> months;
        charges = months * SENIOR;
        cout << "The total charges are $" << charges << endl;
    }
    else if (choice == QUIT_CHOICE)
    {
        cout << "Program ending.\n";
    }
    else
    {
        cout << "The valid choices are 1 through 4. Run the\n"
             << "program again and select one of those.\n";
    }
    return 0;
}
```

```
// This test scoring program does not accept test scores
// that are less than 0 or greater than 100.
#include <iostream>
using namespace std;

int main()
{
    // Constants for grade thresholds
    const int A_SCORE = 90,
              B_SCORE = 80,
              C_SCORE = 70,
              D_SCORE = 60,
              MIN_SCORE = 0,      // Minimum valid score
              MAX_SCORE = 100;    // Maximum valid score

    int testScore; // To hold a numeric test score

    // Get the numeric test score.
    cout << "Enter your numeric test score and I will\n"
         << "tell you the letter grade you earned: ";
    cin >> testScore;
    // Validate the input and determine the grade.
    if (testScore >= MIN_SCORE && testScore <= MAX_SCORE)
    {
        // Determine the letter grade.
        if (testScore >= A_SCORE)
          cout << "Your grade is A.\n";
        else if (testScore >= B_SCORE)
          cout << "Your grade is B.\n";
        else if (testScore >= C_SCORE)
          cout << "Your grade is C.\n";
        else if (testScore >= D_SCORE)
          cout << "Your grade is D.\n";
        else
          cout << "Your grade is F.\n";
    }
    else
    {
        // An invalid score was entered.
        cout << "That is an invalid score. Run the program\n"
             << "again and enter a value in the range of\n"
             << MIN_SCORE << " through " << MAX_SCORE << ".\n";
    }

    return 0;
}
```

4.11

Validating User Input

4.13

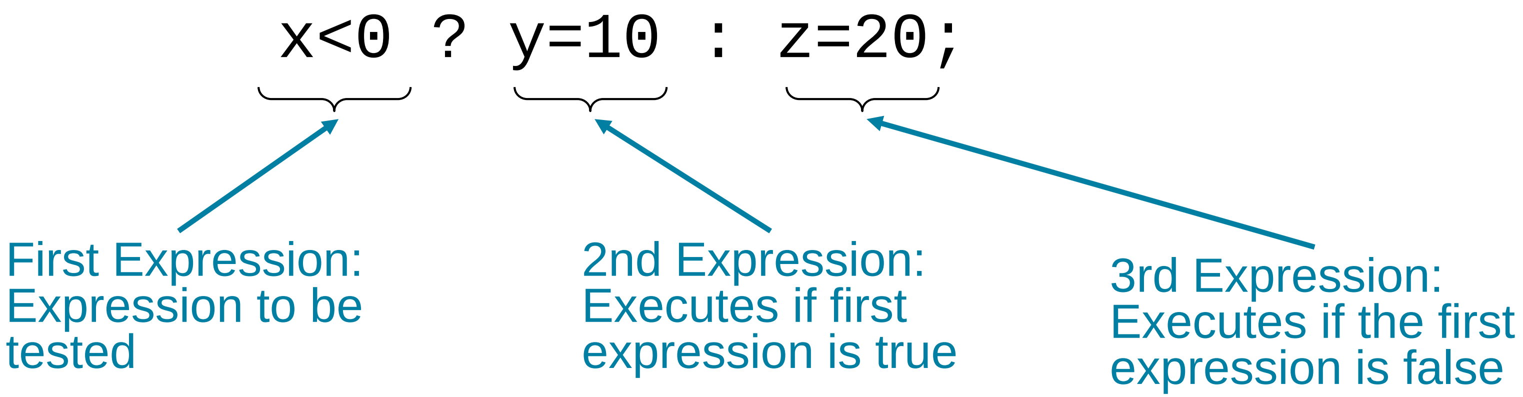
العامل الشرطي

The Conditional Operator

The Conditional Operator

- يمكن استخدامه لإنشاء عبارات if/else موجزة
- الشكل العام:

```
expr ? expr : expr;
```



```
x<0 ? y=10 : z=20;
```

First Expression: Expression to be tested

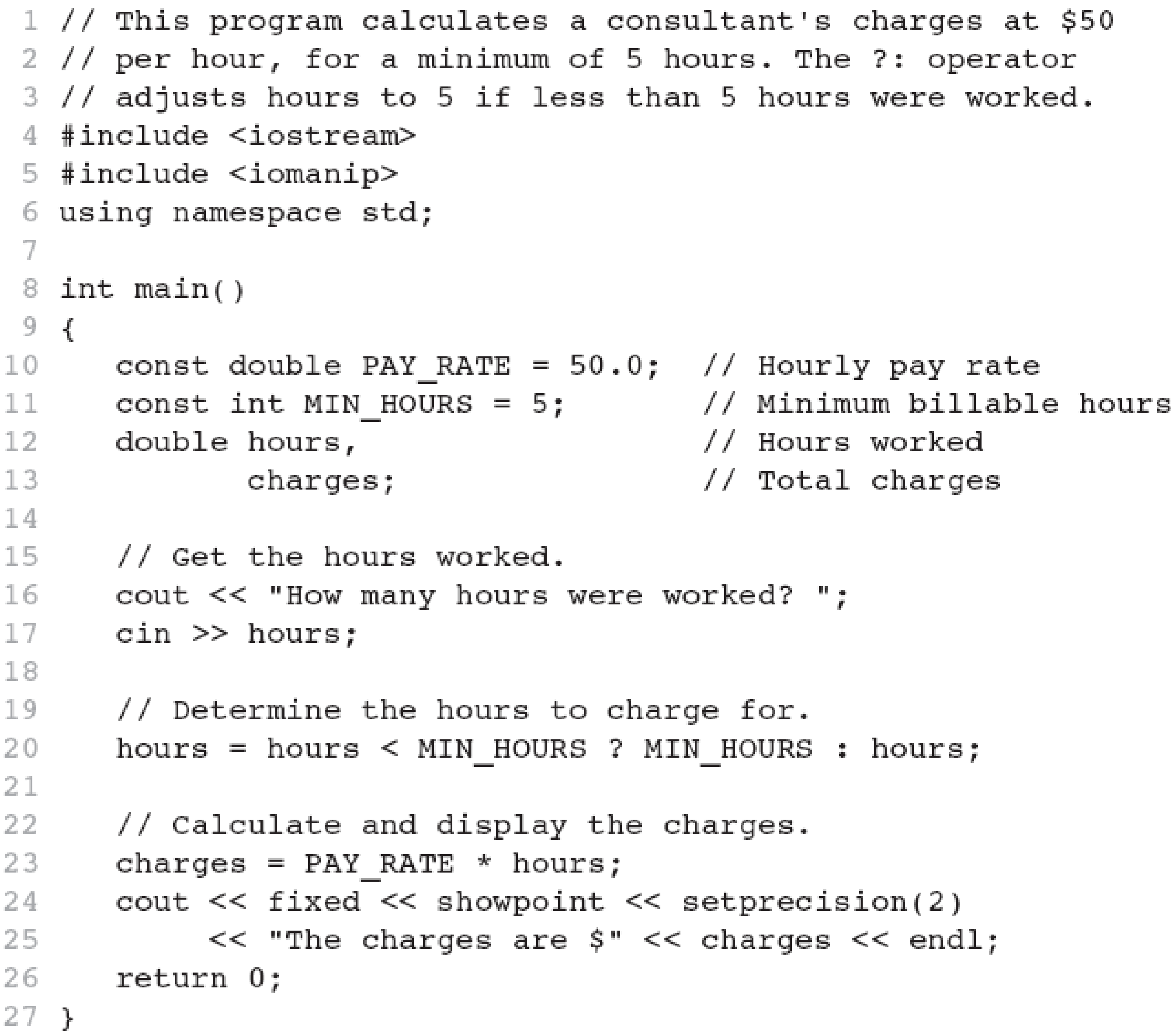
2nd Expression: Executes if first expression is true

3rd Expression: Executes if the first expression is false

The Conditional Operator

- قيمة التعبير الشرطي هي:
  - قيمة التعبير الثاني إذا كان التعبير الأول صحيحا
  - قيمة التعبير الثالث إذا كان التعبير الأول خاطئا
- قد تلزم الأقواس () في تعبير ما بسبب أولوية العامل الشرطي

The Conditional Operator in Program 4-22



```
// This program calculates a consultant's charges at $50
// per hour, for a minimum of 5 hours. The ?: operator
// adjusts hours to 5 if less than 5 hours were worked.
#include <iostream>
#include <iomanip>
using namespace std;

int main()
{
   const double PAY_RATE = 50.0;   // Hourly pay rate
   const int MIN_HOURS = 5;        // Minimum billable hours
   double hours,                   // Hours worked
          charges;                 // Total charges

   // Get the hours worked.
   cout << "How many hours were worked? ";
   cin >> hours;

   // Determine the hours to charge for.
   hours = hours < MIN_HOURS ? MIN_HOURS : hours;

   // Calculate and display the charges.
   charges = PAY_RATE * hours;
   cout << fixed << showpoint << setprecision(2)
        << "The charges are $" << charges << endl;
   return 0;
}
```

4.14

switch عبارة

The switch Statement

The `switch` Statement

- تستخدم لاختيار عبارة بين عدة عبارات أو بدائل.
- يمكن استخدامها في بعض الحالات عوضا عن عبارات `if/else if`.

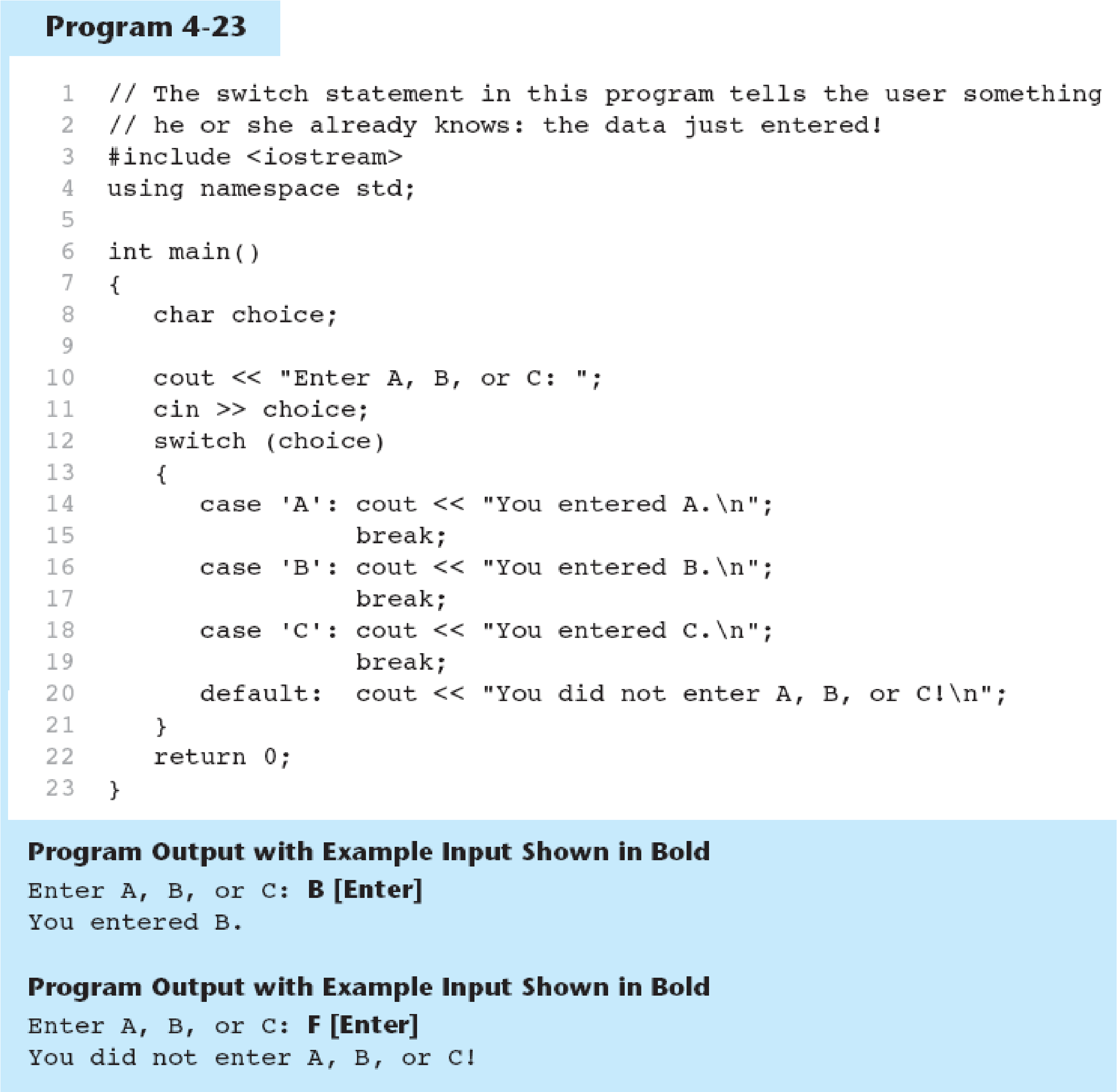
شكل عبارة switch

switch Statement Format

- switch (*expression*) //integer (character variables and literals are considered integers.)

```
{
 case exp1:  statement1;
 case exp2:  statement2;
 ...
 case expn:  statementn;
 default:    statementn+1;
}
```

The switch Statement in Program 4-23



Program 4-23

```
// The switch statement in this program tells the user something
// he or she already knows: the data just entered!
#include <iostream>
using namespace std;

int main()
{
   char choice;

   cout << "Enter A, B, or C: ";
   cin >> choice;
   switch (choice)
   {
      case 'A': cout << "You entered A.\n";
                break;
      case 'B': cout << "You entered B.\n";
                break;
      case 'C': cout << "You entered C.\n";
                break;
      default:  cout << "You did not enter A, B, or C!\n";
   }
   return 0;
}
```

Program Output with Example Input Shown in Bold

```
Enter A, B, or C: B [Enter]
You entered B.
```

Program Output with Example Input Shown in Bold

```
Enter A, B, or C: F [Enter]
You did not enter A, B, or C!
```

`switch` Statement Requirements

١) يجب أن يكون التعبير *`expression`* متغيرا صحيحا أو تعبيرا نتيجته قيمة صحيحة

٢) يجب أن تكون التعابير exp1 حتى expn تعابير أو محارف صحيحة ثابتة constant integer expressions or literals ويجب أن تكون فريدة (مميزة) في عبارة switch

٣) تعتبر default اختيارية لكن يوصى بها.

`switch` Statement-How it Works

1) *`expression`* is evaluated

2) The value of *`expression`* is compared against *`exp1`* through *`expn`*.

3) If *`expression`* matches value *`expi`*, the program branches to the statement following *`expi`* and continues to the end of the `switch`

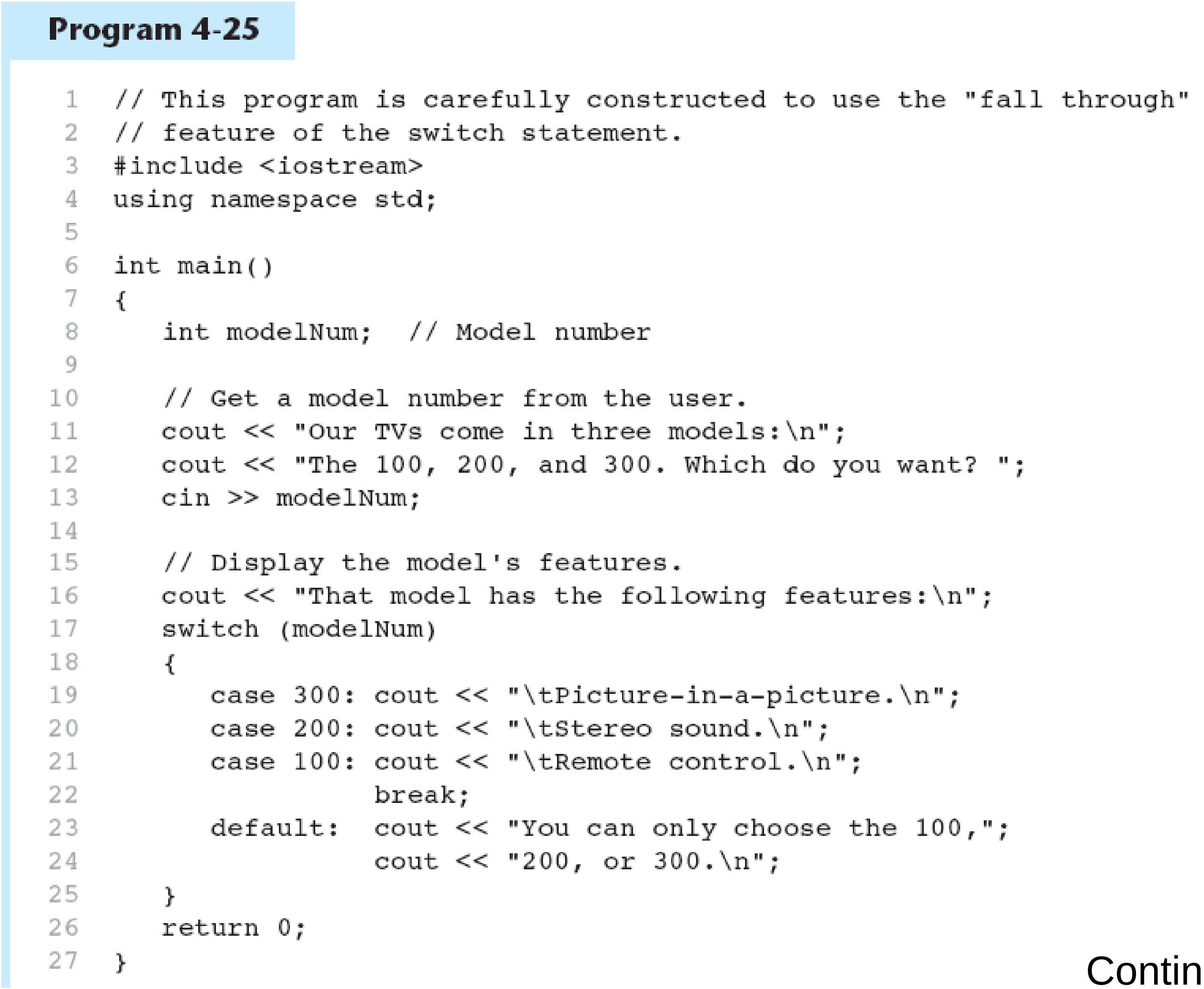
4) If no matching value is found, the program branches to the statement after `default:`

عبارة break
break Statement

- تستخدم للخروج من عبارة switch

- If it is left out, the program "falls through" the remaining statements in the switch statement

break and default statements in Program 4-25

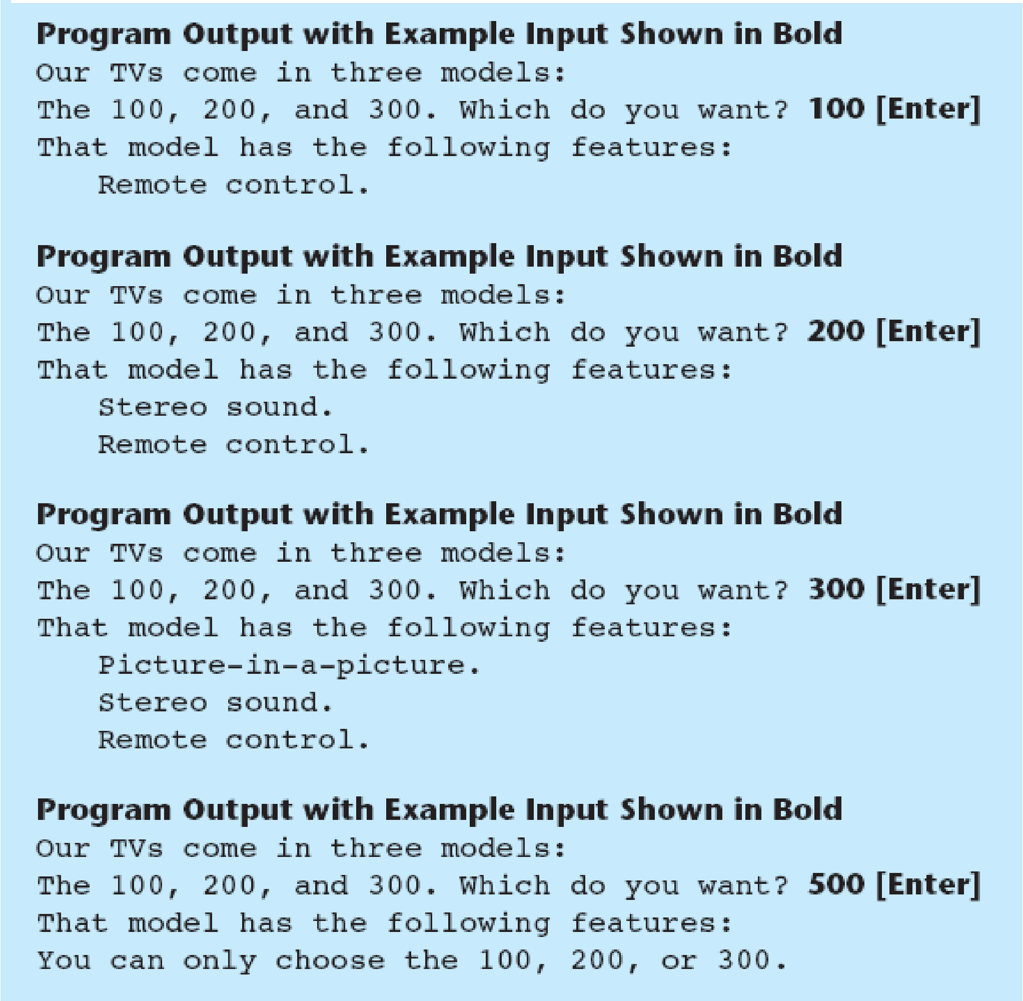


Program 4-25

```
// This program is carefully constructed to use the "fall through"
// feature of the switch statement.
#include <iostream>
using namespace std;

int main()
{
   int modelNum;  // Model number

   // Get a model number from the user.
   cout << "Our TVs come in three models:\n";
   cout << "The 100, 200, and 300. Which do you want? ";
   cin >> modelNum;

   // Display the model's features.
   cout << "That model has the following features:\n";
   switch (modelNum)
   {
      case 300: cout << "\tPicture-in-a-picture.\n";
      case 200: cout << "\tStereo sound.\n";
      case 100: cout << "\tRemote control.\n";
                break;
      default:  cout << "You can only choose the 100,";
                cout << "200, or 300.\n";
   }
   return 0;
}
```

Continued…

`break` and `default` statements in Program 4-25



Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 100 [Enter]
That model has the following features:
   Remote control.
```

Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 200 [Enter]
That model has the following features:
   Stereo sound.
   Remote control.
```

Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 300 [Enter]
That model has the following features:
   Picture-in-a-picture.
   Stereo sound.
   Remote control.
```

Program Output with Example Input Shown in Bold

```
Our TVs come in three models:
The 100, 200, and 300. Which do you want? 500 [Enter]
That model has the following features:
You can only choose the 100, 200, or 300.
```

Using `switch` in Menu Systems

- `switch` statement is a natural choice for menu-driven program:
  - display the menu
  - then, get the user's menu selection
  - use user input as `expression` in `switch` statement
  - use menu choices as *`expr`* in `case` statements

Pearson

4.15

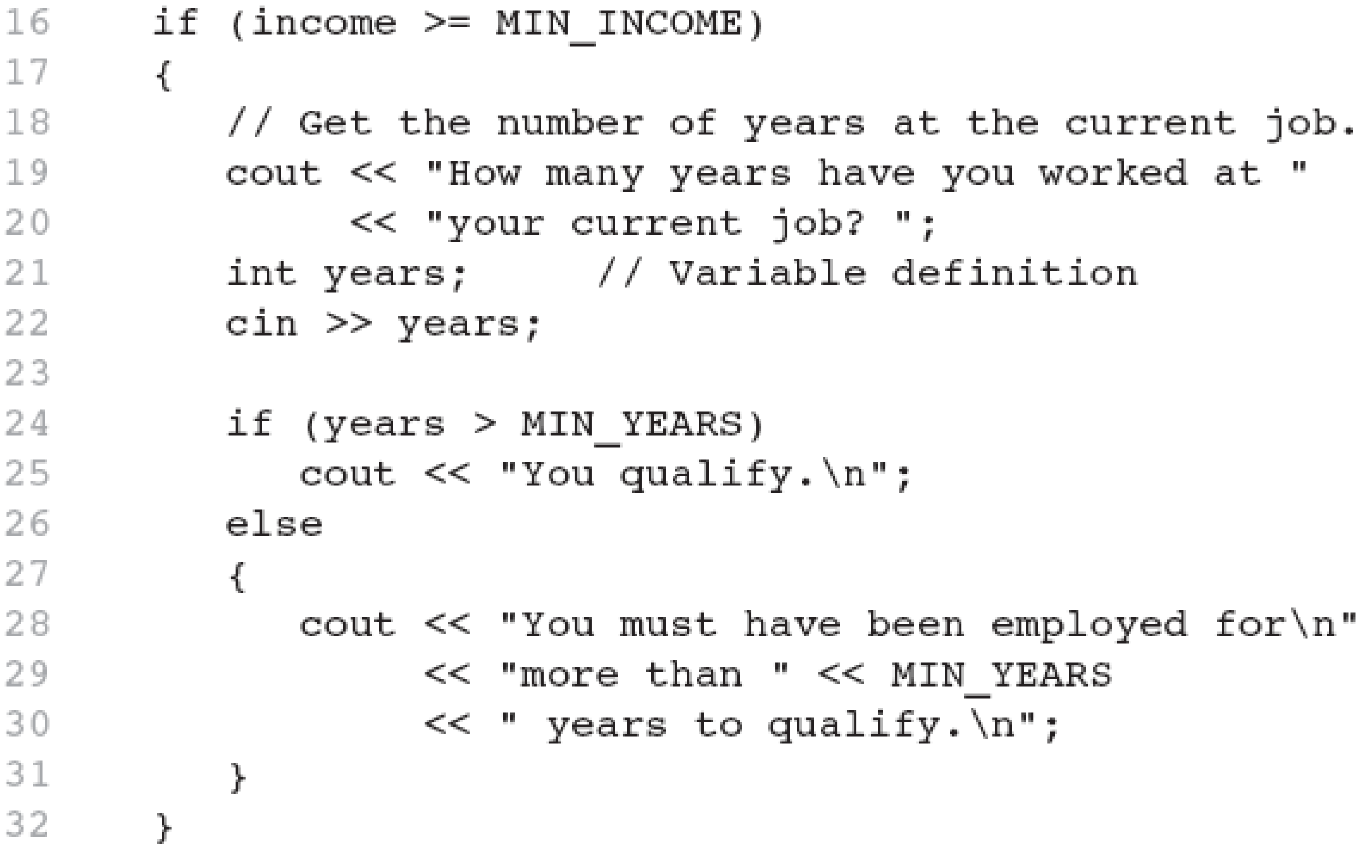
More About Blocks and Scope

More About Blocks and Scope

- مجال متغير ما هو الكتلة البرمجية (البلوك) التي عرف فيها من نقطة التعريف حتى نهاية البلوك.
- في العادة يتم تعريف المتغير في بداية الدالة
- قد يعرف بالقرب من أول استخدام.

تعريف متغير الكتلة الداخلي في البرنامج 4-29

Inner Block Variable Definition in Program 4-29



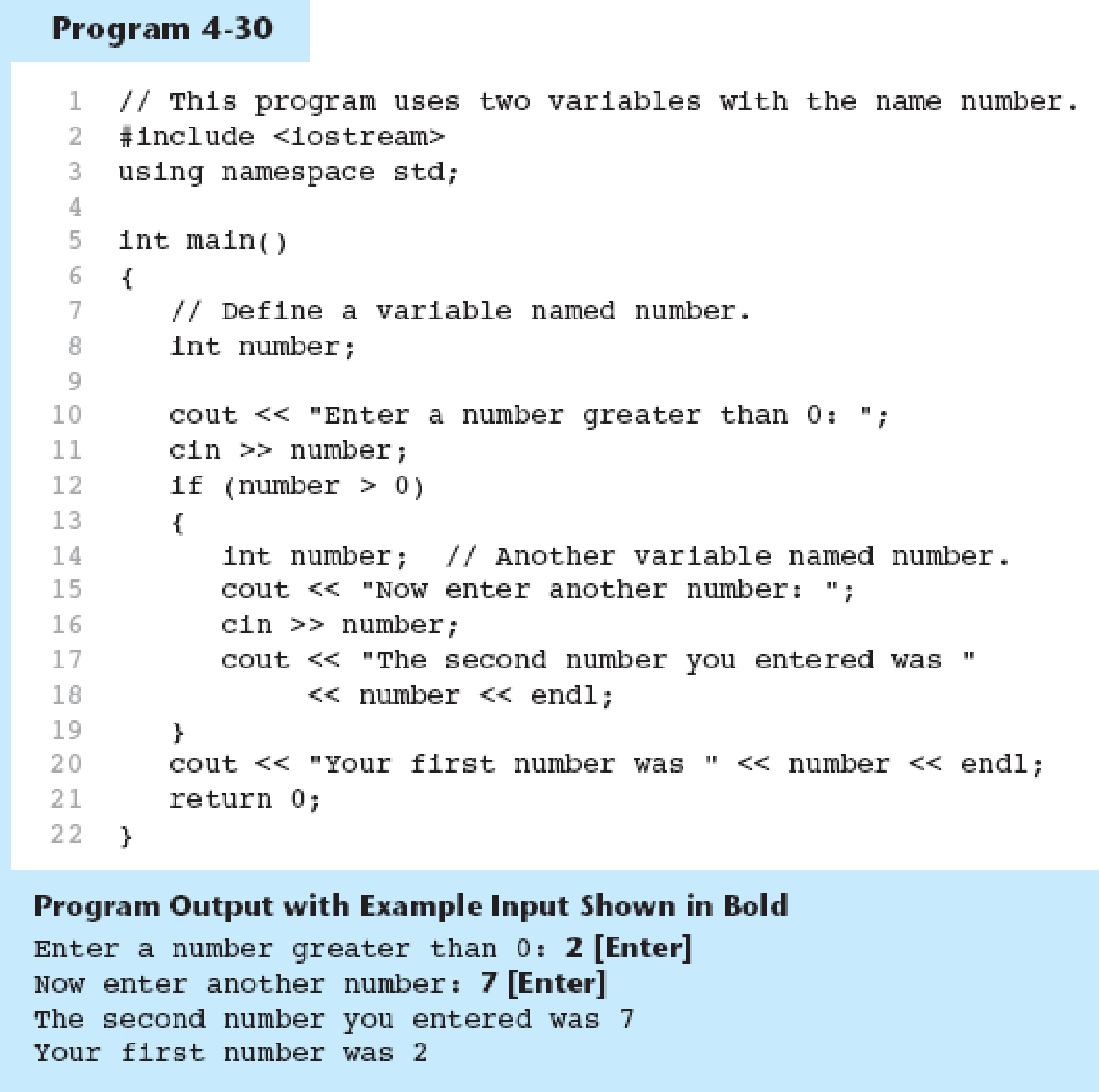
```
if (income >= MIN_INCOME)
{
   // Get the number of years at the current job.
   cout << "How many years have you worked at "
        << "your current job? ";
   int years;     // Variable definition
   cin >> years;

   if (years > MIN_YEARS)
      cout << "You qualify.\n";
   else
   {
      cout << "You must have been employed for\n"
           << "more than " << MIN_YEARS
           << " years to qualify.\n";
   }
}
```

المتغيرات التي لها نفس الاسم

Variables with the Same Name

- Variables defined inside { } have local or block scope
- When inside a block within another block, can define variables with the same name as in the outer block.
  - When in inner block, outer definition is not available
  - Not a good idea

Two Variables with the Same Name in Program 4-30



Program 4-30

```
// This program uses two variables with the name number.
#include <iostream>
using namespace std;

int main()
{
   // Define a variable named number.
   int number;

   cout << "Enter a number greater than 0: ";
   cin >> number;
   if (number > 0)
   {
      int number;  // Another variable named number.
      cout << "Now enter another number: ";
      cin >> number;
      cout << "The second number you entered was "
           << number << endl;
   }
   cout << "Your first number was " << number << endl;
   return 0;
}
```

Program Output with Example Input Shown in Bold

```
Enter a number greater than 0: 2 [Enter]
Now enter another number: 7 [Enter]
The second number you entered was 7
Your first number was 2
```

Copyright