Chapter 4
Control Statements: Part 1

李慧颖
huiyingli@seu.edu.cn

OBJECTIVES

- Basic problem-solving techniques.
- To develop algorithms through the process of top-down, stepwise refinement (自顶向下-逐步求精).
- To use the if and if...else selection statements(选择语句) to choose among alternative actions.
- To use the while repetition statement (循环语句) to execute statements in a program repeatedly.
- Counter-controlled(计数器控制) repetition and sentinel-controlled(标记值控制) repetition.
- To use the increment(自增), decrement(自减) and assignment(赋值) operators

OBJECTIVES

- **4.1 Introduction**
- 4.2 Control Structures
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.1 Introduction

- **Step 1: Before writing a program:**
  - **Have a through understanding of the problem**
  - **Carefully plan an approach for solving it**
- **Step 2: While writing a program:**
  - **Know what "building blocks(组件)" are available**
  - **Use good programming principles**

（1）算法设计（2） 编程实现

4.1 Introduction

- 1. 算法设计
  - 算法(Algorithm)
  - 伪代码(Pseudocode)
  - 自顶向下-逐步求精(Top-down, Stepwise refinement) …Ch4.7
- 2. 编程实现
  - 组件: 控制语句(1顺序-3选择-3循环)
  - 编程原则: 结构化编程(Structured Programming)

4.1 Introduction

- **Pseudocode(伪代码)**

  介于算法和程序语言之间,辅助程序员进行算法设计的非正式语言

- 目的：辅助算法设计
- 特点：
  - 不是真正的编程语言,无法编译执行
  - 独立于编程语言,与具体语言无关,无需考虑特定语言的语法
  - 使用日常英语描述,仅包含字符

4.1 I

- 伪代码仅描述可执
  - 变量的声明不会
- 伪代码仅描
  - 例如 retu
- 例： 两数相

Figure 4.1. Pseudocode for the addition program

```
1 Prompt the user to enter the first integer
2 Input the first integer
3
4 Prompt the user to enter the second integer
5 Input the second integer
6
7 Add first integer and second integer, store result
8 Display result
```

```
#include <iostream> // allows program to perform input and output
int main()
{
   int number1; // first integer to add
   int number2; // second integer to add
   int sum; // sum of number1 and number2

   std::cout << "Enter first integer: "; // prompt user for data
   std::cin >> number1; // read first integer from user into number1
   std::cout << "Enter second integer: "; // prompt user for data
   std::cin >> number2; // read second integer from user into number2

   sum = number1 + number2; // add the numbers; store result in sum
   std::cout << "Sum is " << sum << std::end1; // display sum; end line

   return 0; // indicate that program ended successfully
} // end function main
```

OBJECTIVES

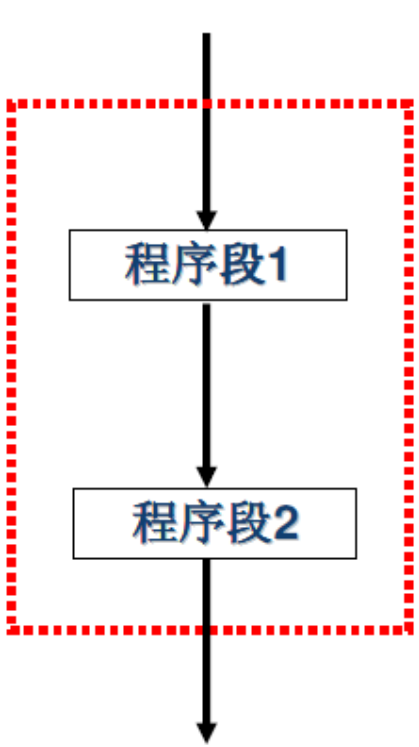
- 4.1 Introduction
- **4.2 Control Structures**
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.2 Control Structures

- 结构化程序设计的概念是E.W.Dijkstra在60年代末提出的，主要目的是控制编程中的复杂性。
- 该方法的要点是：
  - 1. 没有goto语句
  - 2. 一个入口，一个出口
  - 3. 自顶向下、逐步求精的分解
  - 4. 主程序员组

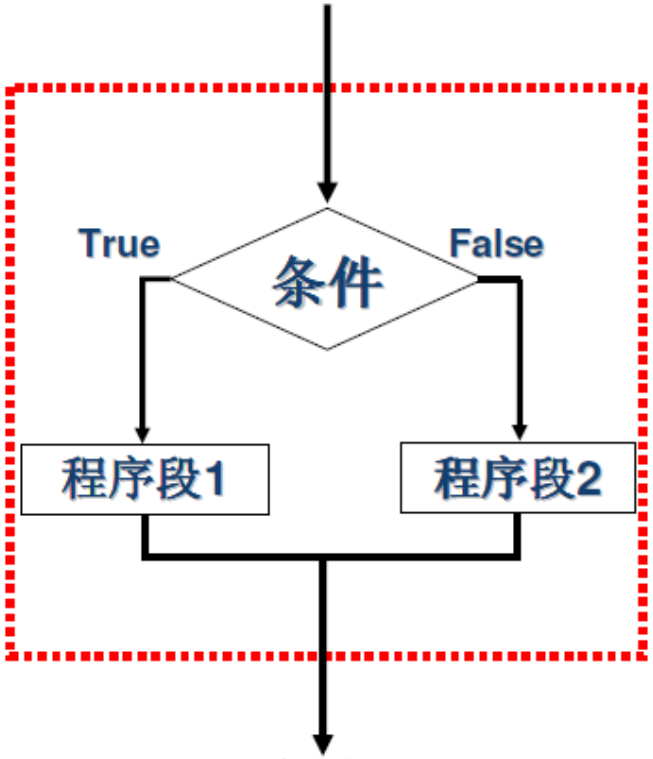
- 1, 2 解决程序结构规范化问题
- 3 解决将大划小，将难化简的求解方法问题
- 4 解决软件开发的人员组织结构问题

4.2 Control Structures



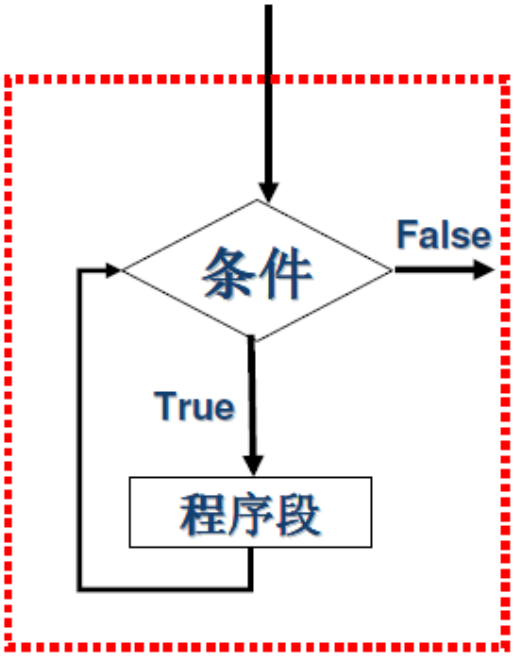
顺 序

Sequence



选 择

Selection



循 环

Repetition

4.2 Control Structures —顺序结构

- 顺序结构：计算机顺序执行C++语句

4.2 Control Structures
--选择结构

- 单选(single-selection statement)
  - if….
  - 选择或忽略某个（组）行为
- 双选(double-selection statement)
  - if…else
  - 在两个（组）行为中选择其一
- 多选(multiple-selection statement)
  - switch
  - 在多个（组）行为中选择

4.2 Control Structures
--循环结构

- While
- 执行某个(组)行为0 或多次(0+)

- for
- 执行某个(组)行为0 或多次(0+)

- do...while
- 执行某个(组)行为1 或多次(1+)

Figure 4.3. C++ keywords.

C++ Keywords

Keywords common to the C and C++ programming languages

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

C++-only keywords

and	and_eq	asm	bitand	bitor
bool	catch	class	compl	const_cast
delete	dynamic_cast	explicit	export	false
friend	inline	mutable	namespace	new
not	not_eq	operator	or	or_eq
private	protected	public	reinterpret_cast	static_cast
template	this	throw	true	try
typ				
xor				

所有的C++关键词都只包含小写字母！

4.2 Control Structures --程序的组成

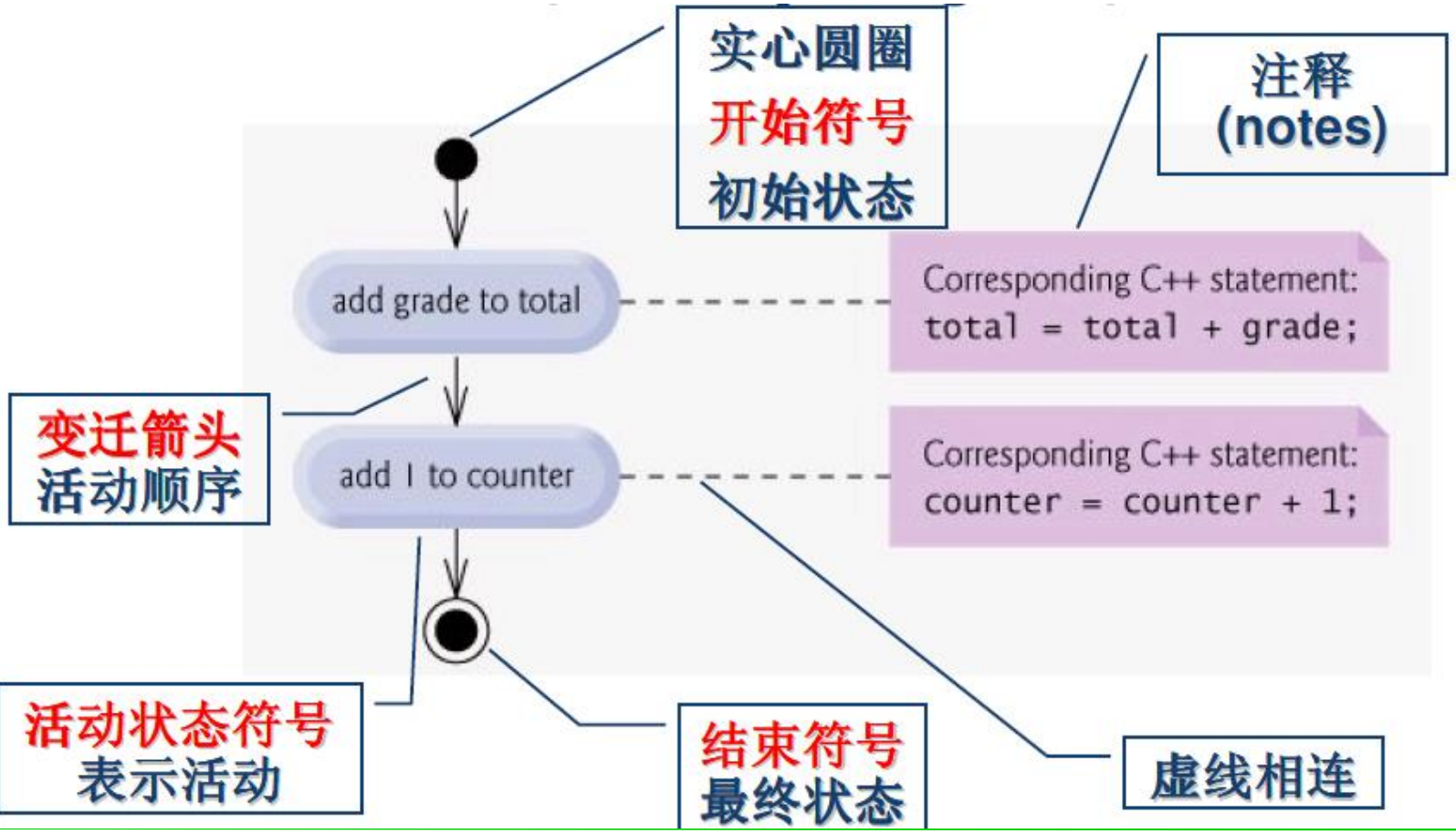
- C++程序有七种控制结构
  - 顺序结构
  - 3种选择结构
  - 3种循环结构
- 每个控制结构可以用活动图表示，只有一个入口和一个出口,即仅有一对UML开始符号+结束符号

4.2 Control Structures
—程序的组成

- UML活动图(Activity Diagram)：用于描述系统工作流，由一些专用符号组成
  - 活动状态符号（Action State Symbol）
  - 判断符号（菱形，Diamond）
  - 小圆圈（Small Circle）
  - 变迁箭头（Transition Arrow）
  - 等等

4.2 Control Structures
—程序的组成

- UML的活动图(Activity Diagram)

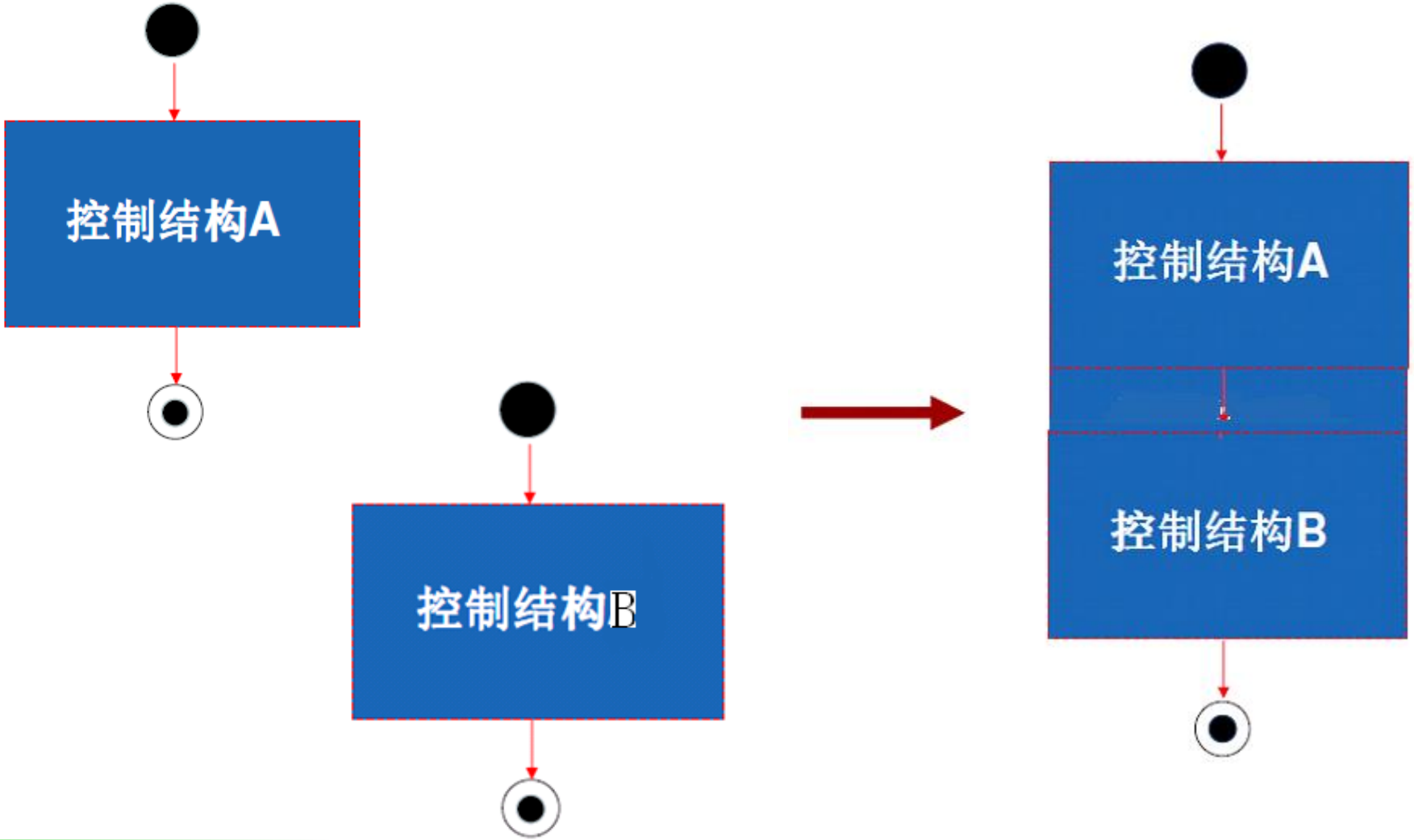


4.2 Control Structures --程序的组成

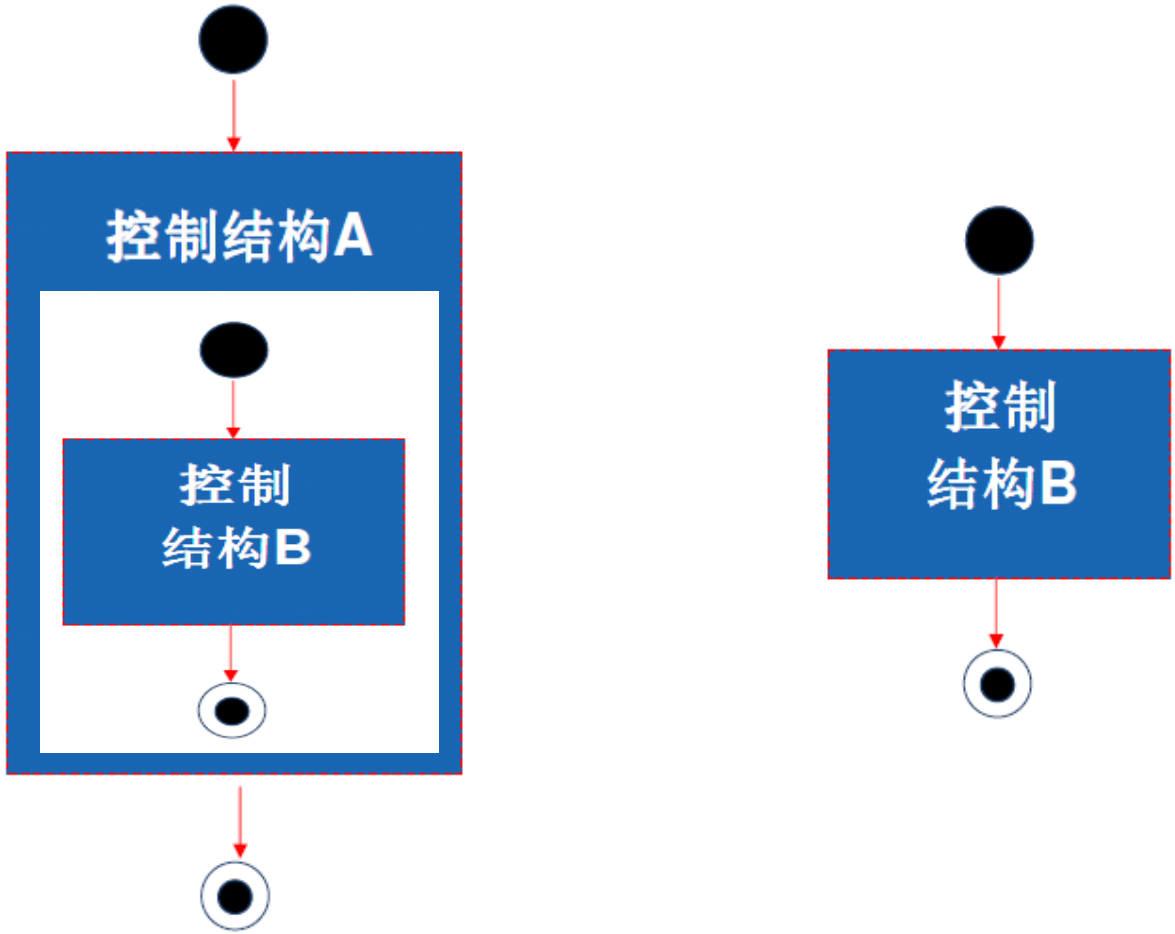
- 每个C++程序都是根据程序算法对这七种控制结构的组合

- 控制结构的两种组合方式：
  - 堆叠(control-statement stacking)：一个控制结构的出口与另一个的入口连接
  - 嵌套(control-statement nesting)：一个控制结构中包含另一个控制结构

4.2 Control Structures --堆叠



4.2 Control Structures --嵌套



4.2 Control Structures --程序的组成

- 总结
- 任何C++程序都是根据程序算法对这七种控制结构（顺序+ 3选择+ 3循环）的进行两种方式（堆叠+ 嵌套）组合而成。

OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- **4.3 if Selection Statement**
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.3 if Selection Statement

- 作用：根据判断条件condition，执行或者忽略某个（组）动作
- ① 编程任务：当学生成绩大于等于60分，输出合格
- ② Pseudocode（伪代码）：

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

  Print "Passed"

- ③ C++代码：

```
if ( grade >= 60 )
    cout << "Passed";
```

```
if ( grade >= 60 ) cout << "Passed";
```

4.3 if Selection Statement

- Condition
- 可以是bool 类型的关系（>,<,>=,<=）或者等价（==, !=）表达式
- 也可以是任何数值表达式
  - 表达式的值为0，则表示false
  - 表达式的值非0(nonzero)，则表示true

```
int a = 3, b = 3;// b=-3;
if (a+b)
    cout<<"TRUE"<<endl;
```

4.3 if Selection Statement

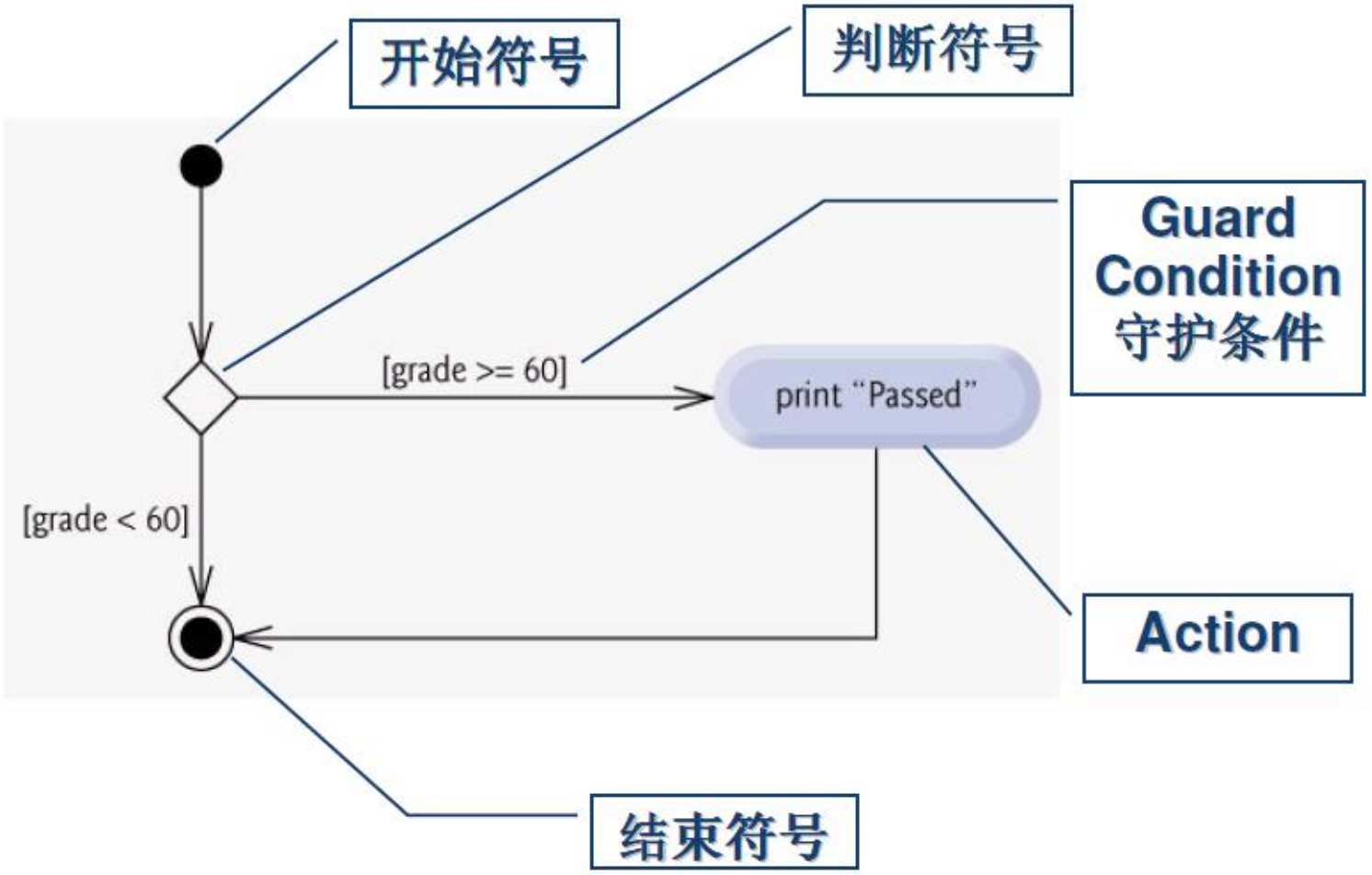
- 如果条件满足时，需执行一组动作呢？

```
if ( grade >= 60 ) {
    cout << "Passed";
    counter++;
}
```

- 程序中任何可以放置单条语句的地方，都可以放置语句块！

- •{ }内的语句称为复合语句(Compound Statements)或者语句块(Block)
- •{ }内可以是>=0条语句，一般是多条语句
  - 0，不执行任何语句，称为null语句(空语句)
  - 1，通常{ }可省略，特例摇摆else

4.3 if Selection Statement



OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- 4.3 if Selection Statement
- **4.4 if...else Double-Selection Statement**
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.4 if.....else Double-Selection Statement

```
18  void GradeBook::setCourseName( string name )          P87 Fig3.16
19  {
20     if ( name.length() <= 25 ) // if name has 25 or fewer characters
21        courseName = name; // store the course name in the object
22
23     if ( name.length() > 25 ) // if name has more than 25 characters
24     {
25        // set courseName to first 25 characters of parameter name
26        courseName = name.substr( 0, 25 ); // start at 0, length of 25
27
28        cout << "Name \"" << name << "\" exceeds maximum length (25).\n"
29           << "Limiting courseName to first 25 characters.\n" << endl;
30     } // end if
31  } // end function setCourseName
```

- 第20-30行：

  如果长度不大于25，则保持不变；

  如果大于25，则截取前25个字符

4.4 if.....else Double-Selection Statement

- 作用：根据条件值为true或者false，分别执行不同的行为
- ① 编程任务：当学生成绩大于等于60分，输出合格，否则输出不合格
- ② Pseudocode（伪代码）：

```
If student's grade is greater than or equal to 60
    Print “Passed”
Else
    Print “Failed”
```

- ③ C++代码

```
if ( grade >= 60 )
   cout << "Passed";
else
   cout << "Failed";
```

4.4 if……else Double-Selection Statement

```
if (a >= b){
    largest = a;
    smallest = b;
}
if (a < b){
    largest = b;
    smallest = a;
}
```

```
if (a >= b){
    largest = a;
    smallest = b;
}
else{
    largest = b;
    smallest = a;
}
```

4.4 if……else Double-Selection Statement

- ?: Conditional Operator(条件操作符)
  - C++中唯一的三元运算符(ternary operator)
  - 操作数1 ? 操作数2 : 操作数3
  - 第一个操作数是条件，第二个操作数是条件为true时整个条件表达式的值，第三个操作数是条件为false时整个条件表达式的值

```
if ( grade >= 60 )
   cout << "Passed";
else
   cout << "Failed";
```

4.4 if.....else Double-Selection Statement

- 操作数 ? Action1 : Action2
- 以操作数为条件, 当条件为true时执行Action1, 为false时执行Action2

```
if ( grade >= 60 )
   cout << "Passed";
else
   cout << "Failed";
```

- grade>=60 ? cout<<"Passed" : cout<<"Failed";

4.4 if.....else Double-Selection Statement

Operators						Associativity	Type
::						left to right	scope resolution
()						left to right	parentheses
++	--	static_cast<*type*>()				left to right	unary (postfix)
++	--	+	-			right to left	unary (prefix)
*	/	%				left to right	multiplicative
+	-					left to right	additive
<<	>>					left to right	insertion/extraction
<	<=	>	>=			left to right	relational
==	!=					left to right	equality
?:						right to left	conditional
=	+=	-=	*=	/=	%=	right to left	assignment

P130 Fig 4.22

4.4 if……else Double-Selection Statement

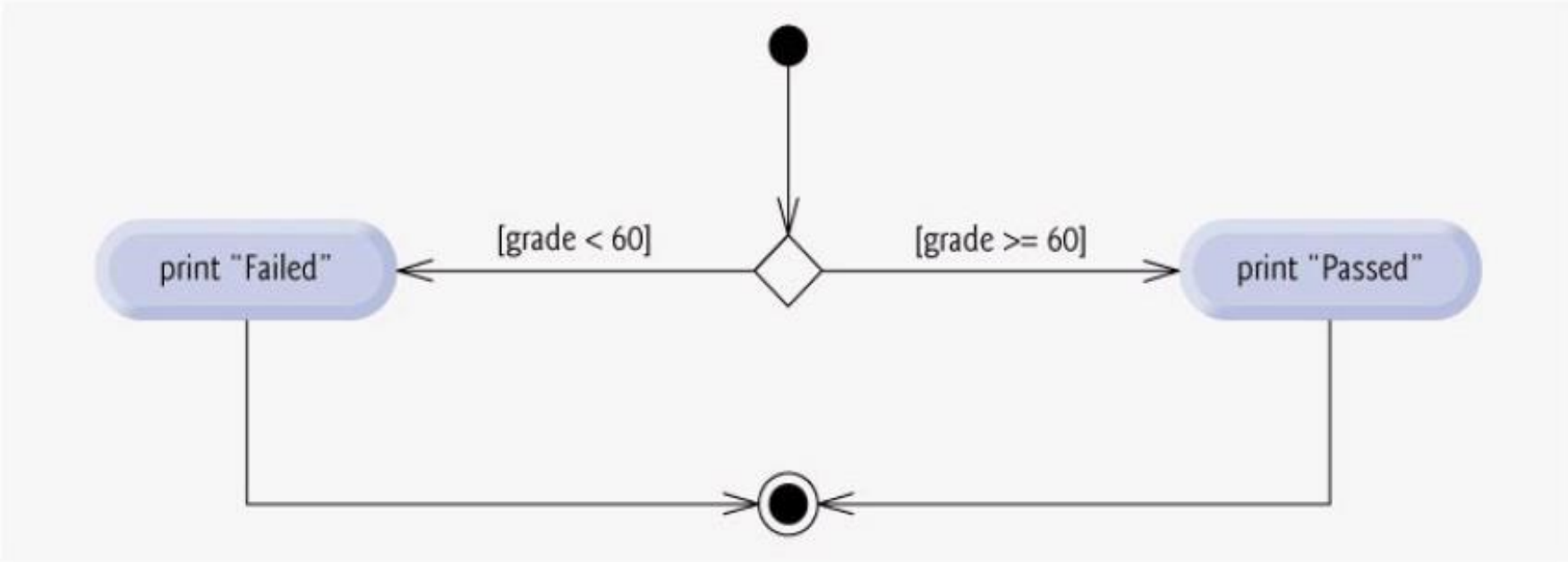
```
cout << ( grade >= 60 ? "Passed" : "Failed" );
```

```
string  s = grade >= 60 ? "Passed" : "Failed";
cout << s;
```

```
grade>=60 ? cout<<"Passed":cout<<"Failed";
```

```
if ( grade >= 60 )
   cout << "Passed";
else
   cout << "Failed";
```

4.4 if……else Double-Selection Statement



4.4 if……else Double-Selection Statement --摇摆else问题

- 需求：x>5且y>5时, 输出“x and y are > 5 ”; x不大于5时，输出“ x is <= 5”

```
if ( x > 5 )
  if ( y > 5 )
    cout << "x and y are > 5";
else
   cout << "x is <= 5";
```

- x=6, y=6时， 输出？ x=3, y=6时， 输出？
- Dangling else: 为了避免多义性，C++编译器将else与之前最近的if关联

4.4 if……else Double-Selection Statement --摇摆else问题

```
if ( x > 5 )
   if ( y > 5 )
      cout << "x and y are > 5";
else
    cout << "x is <= 5";
```

编译器的理解：else子句与前面最近的、没有else子句的if配对

```
if ( x > 5 )
    if ( y > 5 )
       cout << "x and y are > 5";
    else
       cout << "x is <= 5";
```

x>5 而且 y>5

x>5 而且 y<=5

4.4 if……else Double-Selection Statement --摇摆else问题

```
if ( x > 5 )
  if ( y > 5 )
    cout << "x and y are > 5";
else
  cout << "x is <= 5";
```

```
if ( x > 5 )
  if ( y > 5 )
    cout << "x and y are > 5";
  else
    cout << "x is <= 5";
```

正确的写法

```
if ( x > 5 )
{
  if ( y > 5 )
    cout << "x and y are > 5";
}
else
  cout << "x is <= 5";
```

4.4 if……else Double-Selection Statement --嵌套

```
if ( studentGrade >= 90 )
   cout << "A";
else
   if (studentGrade >= 80 )
      cout << "B";
   else
      if (studentGrade >= 70 )
         cout << "C";
      else
         if ( studentGrade >= 60 )
            cout << "D";
         else
            cout << "F";
```

```
if ( studentGrade >= 90 )
   cout << "A";
else if (studentGrade >= 80 )
   cout << "B";
else if (studentGrade >= 70 )
   cout << "C";
else if ( studentGrade >= 60 )
   cout << "D";
else
   cout << "F";
```

OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- **4.5 while Repetition Statement**
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.5 while Repetition Statement

- 需求：程序设计中经常需要对相同的操作重复执行多次
- 循环(loop): 根据某个条件的满足与否决定是否重复执行一个（组）行为
- 循环一般由四个部分组成：
  - 循环初始化
  - 循环条件
  - 循环体
  - 下一次循环准备

4.5 while Repetition Statement

- While there are more items on my shopping list, Purchase next item and cross it off my list
- 循环初始化： Prepare the shopping list
- 循环条件： there are more items on my shopping list
- 循环体： Purchase next item， cross it off my list
- 下一次循环准备： cross it off my list

4.5 while Repetition Statement

- Repeat an action while some condition remains ture

  while (<条件>) <语句>

表示条件的关系或逻辑表达式

简单语句或复合语句{ }

- Find the first power of 3 larger than 100

循环初始化

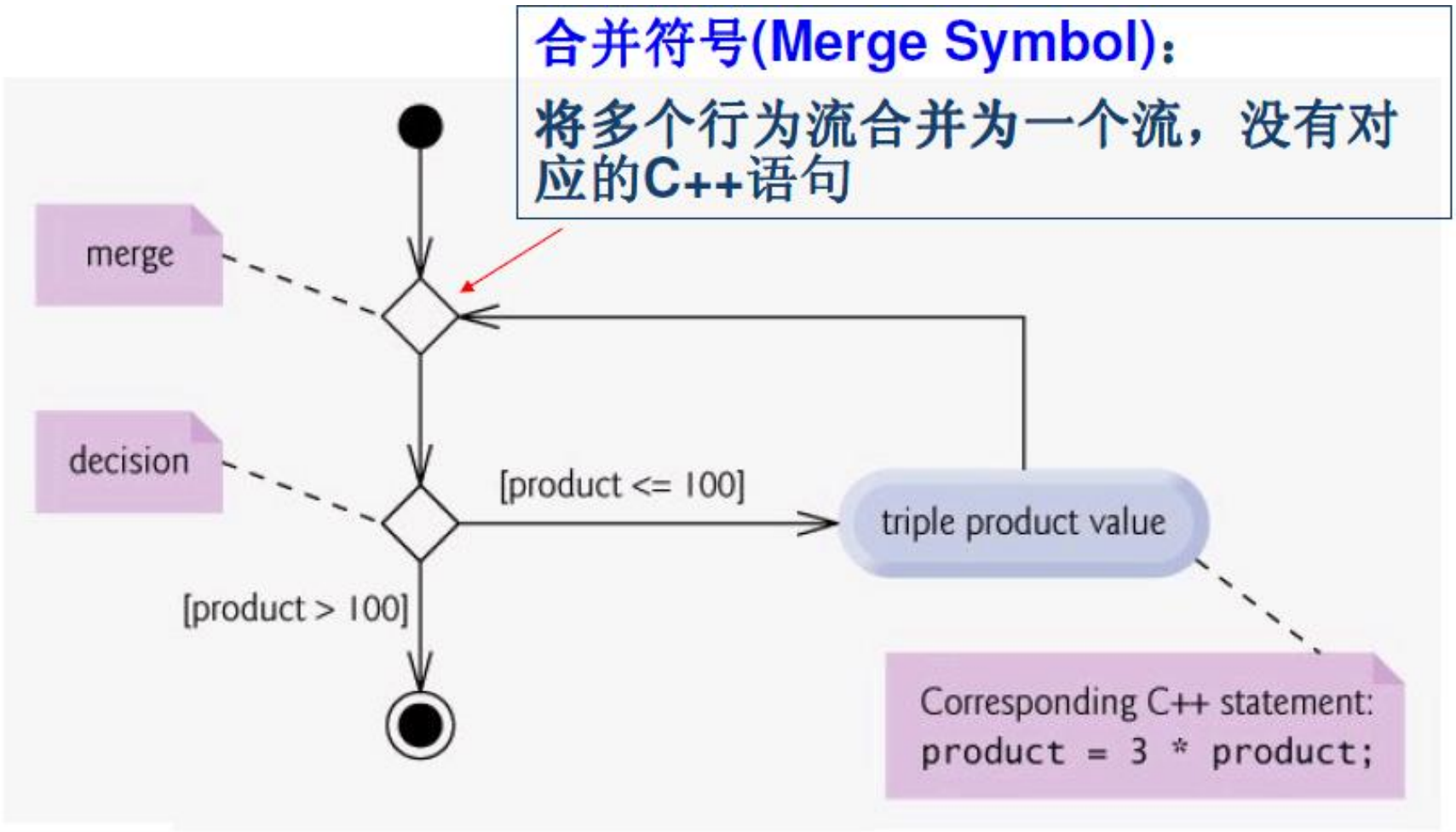
```
int product = 3;
while ( product <= 100)
   product = 3 * product;
```

循环条件

循环体和下一次循环准备

243

4.5 while Repetition Statement



OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- **4.6 Formulating Algorithms: Counter-Controlled Repetition**
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.6 Formulating Algorithms: Counter-Controlled Repetition

计数器控制的循环

- Use a variable called a counter to control the number of times a group of statements will execute
- definite repetition: the number of repetitions is known before the loop begins executing

4.6 Formulating Algorithms: Counter-Controlled Repetition

- 编程任务

  A class of ten students took a quiz. The grades for this c display t average

- **Pseudoc**

Set total to zero
Set grade counter to one
} 初始化变量

计数器

While grade counter is less than or equal to ten
    Prompt the user to enter the next grade
    Input the next grade
    Add the grade into the total
    Add one to the grade counter
} 循环体

Set the class average to the total divided by ten
Print the total of the grades for all students in the class
Print the class average

4.6 Formulating Algorithms: Counter-Controlled Repetition

- 类定义(Page 88)
- **Fig. 4.8: 接口文件GradeBook.h**
- **Fig. 4.9: 成员函数定义文件GradeBook.cpp**

- 类使用
- **Fig. 4.10: 在main函数中创建并使用类对象**

4.6 Formulating Algorithms: Counter-Controlled Repetition

```
53    total = 0; // initialize total
54    gradeCounter = 1; // initialize loop counter
55
56    // processing phase
57    while ( gradeCounter <= 10 ) // loop 10 times
58    {
59       cout << "Enter grade: "; // prompt for input
60       cin >> grade; // input next grade
61       total = total + grade; // add grade to total
62       gradeCounter = gradeCounter + 1; // increment by 1
63    } // end while
```

4.6 Formulating Algorithms: Counter-Controlled Repetition

变量需要初始化

- 总和变量通常被初始化为0
- 根据使用情况，计数器变量通常应先初始化为0或1
- 未初始化变量会包含垃圾值(garbage value)，也称为未定义值(undefined value)，为该变量保存内存地址中最后存放的值

OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- **4.7 Formulating Algorithms: Sentinel-Controlled Repetition**
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

- 问题描述：

  Develop a class average program that processes grades for an arbitrary number of students each time its run

- 解决办法：

  a sentinel value(标记值) (also called signal value, dummy value, flag value) is used to indicate the end of data entry

- 标记值必须能与可接受的有效输入值有所区分！

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

- 定义
- **sentinel controlled repetition is also called indefinite repetition because the number of repetitions is *not known* before the loop begins executing**

Determine the class average for the quiz

① Initialize variables
② Input, sum and count the quiz grades
③ Calculate and print the total of all student grades and the class average

① Initialize total to zero; Initialize counter to zero
② Prompt the user to enter the first grade
Input the first grade (possibly the sentinel)
While the user has not yet entered the sentinel
    Add this grade into the running total
    Add one to the grade counter
    Prompt the user to enter the next grade
    Input the next grade (possibly the sentinel)
③ If the counter is not equal to zero
    Set the average to the total divided by the counter
    Print the total of the grades for all students in the class
    Print the class average
else
    Print "No grades were entered"

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

- 类定义（Page 93）
- Fig. 4.12（接口GradeBook.h）
- Fig. 4.13（成员函数定义GradeBook.cpp）
- 类使用（Page 95）
- Fig. 4.14 （在main函数中创建类对象）

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
08 using std::fixed; // ensures that decimal point is displayed
09
10 #include <iomanip> // parameterized stream manipulators
11 using std::setprecision; // sets numeric output precision
```

- std::fixed 浮点数输出格式，定点输出
- <iomanip> 带参数的流操作算子
- std::setprecision 设置输出精度

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
59      gradeCounter = 0; // initialize loop counter
60
61      // processing phase
62      // prompt for input and read grade from user
63      cout << "Enter grade or -1 to quit: ";
64      cin >> grade; // input grade or sentinel value
65
66      // loop until sentinel value read from user
67      while ( grade != -1 ) // while grade is not -1
68      {
69          total = total + grade; // add grade to total
70          gradeCounter = gradeCounter + 1; // increment counter
71
72          // prompt for input and read next grade from user
73          cout << "Enter grade or -1 to quit: ";
74          cin >> grade; // input grade or sentinel value
75      } // end while
```

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
55    double average; // number with decimal point for average

78    if ( gradeCounter != 0 ) // if user entered at least one grade...
79    {
80       // calculate average of all grades entered
81       average = static_cast< double >( total ) / gradeCounter;
82
83       // display total and average (with two digits of precision)
84       cout << "\nTotal of all " << gradeCounter << " grades entered is "
85            << total << endl;
86       cout << "Class average is " << setprecision( 2 ) << fixed << average
87            << endl;
88    } // end if
89    else // no grades were entered, so output appropriate message
90       cout << "No grades were entered" << endl;
```

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
55    double average; // number with decimal point for average
```

```
80    // calculate average of all grades entered
81    average = static_cast< double >( total ) / gradeCounter;
```

- C++常用浮点数类型(Floating-Point Number)
  - float 类型（单精度），7位有效数字，4个字节
  - double 类型（双精度），15位有效数字，8个字节
  - 计算机不能100%精确表示小数，如Pi值
  - 单精度浮点数的精度比双精度低，但是所需内存少，运算速度快
- Floating-Point Constant
  - 程序代码中的小数（如3.1415），缺省作为double处理

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
55    double average; // number with decimal point for average
```

```
80    // calculate average of all grades entered
81    average = static_cast< double >( total ) / gradeCounter;
```

- 显式类型转换(explicit conversion)
- 类型转换操作符。**total**中存放的值仍是整数，而计算时建一个临时的浮点数值（**total**的临时**double**类型的值）除以**gradeCounter**

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
55    double average; // number with decimal point for average
```

```
80    // calculate average of all grades entered
81    average = static_cast< double >( total ) / gradeCounter;
```

C++进行表达式计算时,要求操作数类型一致

- 隐性类型转换(implicit conversion)
- gradeCounter必须提升(Promote)为double之后进行计算，将浮点数除法得到的结果赋给average

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

Operators						Associativity	Type
()						left to right	parentheses
++	--	static_cast< *type* >()				left to right	unary (postfix)
++	--	+	-			right to left	unary (prefix)
*	/	%				left to right	multiplicative
+	-					left to right	additive
<<	>>					left to right	insertion/extraction
<	<=	>	>=			left to right	relational
==	!=					left to right	equality
?:						right to left	conditional
=	+=	-=	*=	/=	%=	right to left	assignment

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
Welcome to the grade book for
CS101 C++ Programming!
Enter grade or -1 to quit: 97
Enter grade or -1 to quit: 88
Enter grade or -1 to quit: 72
Enter grade or -1 to quit: -1
Total of all 3 grades entered is 257
Class average is 85.6667
Press any key to continue
```

```
84    cout << "\nTotal of all " << gradeCounter << " grades entered is "
85         << total << endl;
86    cout << "Class average is " << setprecision( 2 ) << fixed << average
87         << endl;
```

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

setprecision(int n)

- 带参数的流操作算子, parameterized stream manipulator
- 必须: #include <iomanip>
- 定点输出时，参数n指定小数显示的位数，四舍五入: 872.946→872.95
- 仅设置输出格式，原值不变
- 如果不指定输出精度，则浮点数值通常输出六位精度(默认精度, default precision) – 有效数字位

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

- 两个流操作算子
- **fixed**: 使浮点数值以定点格式(而不是科学计数法，浮点)输出
- **showpoint**: 即使数值为整数，仍强制打印小数点和尾部的0，如88.00

- 和endl相同, 无参数, 不需要头文件 <iomanip>

4.7 Formulating Algorithms: Sentinel-Controlled Repetition

```
Welcome to the grade book for
CS101 C++ Programming!
Enter grade or -1 to quit: 97
Enter grade or -1 to quit: 88
Enter grade or -1 to quit: 72
Enter grade or -1 to quit: -1
Total of all 3 grades entered is 257
Class average is 85.67
Press any key to continue
```

OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- **4.8 Formulating Algorithms: Nested Control Statements**
- 4.9 Assignment Operators
- 4.10 Increment and Decrement Operators

4.8 Formulating Algorithms: Nested Control Statements

- 问题描述
- 学校开了一门课，让学生参加房地产经纪人资格考试。去年，10位同学读完这门课并参加了证书考试。学校想知道学生考试情况，请编写一个程序来总结这个结果。已经得到了10个学生的名单，每个姓名后面写1时表示考试通过，写2时表示没有通过。

4.8 Formulating Algorithms: Nested Control Statements

- 问题分析
- 1. 输入每个考试成绩(即1 或2), 程序请求输入考试成绩时, 在屏幕上显示消息“enter result”
- 2. 计算通过和没有通过的考试成绩数
- 3. 显示总成绩, 表示及格人数和不及格人数
- 4. 如果超过8个学生及格, 则打印消息“Raise tuition”

4.8 Formulating Algorithms: Nested Control Statements

- 设计解决方法
- ① 程序要处理10个考试成绩,用计数器控制循环
- ② 每个考试成绩为数字1或2,每次程序读取考试成绩时,如果不是1,则假设其为2
- ③ 使用两个计数器,分别计算及格人数和不及格人数
- ④ 程序处理所有结果之后,要确定是否有超过8个学生及格

4.8 Formulating Algorithms: Nested Control Statements

```
 1  Initialize passes to zero
 2  Initialize failures to zero
 3  Initialize student counter to one
 4
 5  While student counter is less than or equal to 10
 6     Prompt the user to enter the next exam result
 7     Input the next exam result
 8
 9     If the student passed
10        Add one to passes
11     Else
12        Add one to failures
13
14     Add one to student counter
15
16  Print the number of passes
17  Print the number of failures
19  If more than eight students passed
20     Print "Raise tuition"
```

```
21    // process 10 students using counter-controlled loop
22    while ( studentCounter <= 10 )
23    {
24       // prompt user for input and obtain value from user
25       cout << "Enter result (1 = pass, 2 = fail): ";
26       cin >> result; // input result
27
28       // if...else nested in while
29       if ( result == 1 )           // if result is 1,
30          passes = passes + 1;      // increment passes;
31       else                         // else result is not 1, so
32          failures = failures + 1; // increment failures
33
34       // increment studentCounter so loop eventually terminates
35       studentCounter = studentCounter + 1;
36    } // end while
37
38    // termination phase; display number of passes and failures
39    cout << "Passed " << passes << "\nFailed " << failures << endl;
40
41    // determine whether more than eight students passed
42    if ( passes > 8 )
43       cout << "Raise tuition " << endl;
44 } // end function processExamResults
```

OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- **4.9 Assignment Operators**
- 4.10 Increment and Decrement Operators

4.9 Assignment Operators
（赋值运算符）

- 赋值表达式缩写
- c = c + 3; → c+=3;
- +=：加法赋值运算符

- 二元运算符: +, -, *, /, or %
- variable = variable operator expression;
- variable operator= expression;

4.9 Assignment Operators
（赋值运算符）

Assignment operator	Sample expression	Explanation	Assigns
Assume: `int c = 3, d = 5, e = 4, f = 6, g = 12;`			
`+=`	`c += 7`		
`-=`	`d -= 4`		
`*=`	`e *= 5`		
`/=`	`f /= 3`		
`%=`	`g %= 9`		

OBJECTIVES

- 4.1 Introduction
- 4.2 Control Structures
- 4.3 if Selection Statement
- 4.4 if...else Double-Selection Statement
- 4.5 while Repetition Statement
- 4.6 Formulating Algorithms: Counter-Controlled Repetition
- 4.7 Formulating Algorithms: Sentinel-Controlled Repetition
- 4.8 Formulating Algorithms: Nested Control Statements
- 4.9 Assignment Operators
- **4.10 Increment and Decrement Operators**

4.10 Increment and Decrement Operators (自增和自减运算符)

- 一元自增运算符(increment operator) ++
- 一元自减运算符(decrement operator) --

Operator	名称	示例	说明
++	Preincrement (前置自增)	++a	先将a加1，然后在表达式中使用a的新值
++	Postincrement (后置自增)	a++	在a出现的表达式中使用当前值，然后将a加1
--	Predecrement (前置自减)	--b	先将b减1，然后在表达式中使用b的新值
--	Postdrecrement (后置自减)	b--	在b出现的表达式中使用当前值，然后将b减1

4.10 Increment and Decrement Operators (自增和自减运算符)

- passes = passes + 1;
- passes += 1;
- passes++;
- ++passes;
- passes = 10;

1. x = passes++;
2. y = ++passes;

```
1. x=10, passes=11
2. passes=11, y=11
```

- (x+y)++; ✗

4.10 Increment and Decrement Operators (自增和自减运算符)

```
12    c = 5; // assign 5 to c
13    cout << c << endl; // print 5
14    cout << c++ << end
15    cout << c << end
16
17    cout << endl; // skip a line
18
19    // demonstrate preincrement
20    c = 5; // assign 5 to c
21    cout << c << endl; // print 5
22    cout << ++c << end
23    cout << c << end
```

4.10 Increment and Decrement Operators (自增和自减运算符)

高

低

Operators						Associativity	Type
()						left to right	parentheses
++	--	static_cast< *type* > ()				left to right	unary (postfix)
++	--	+	-			right to left	unary (prefix)
*	/	%				left to right	multiplicative
+	-					left to right	additive
<<	>>					left to right	insertion/extraction
<	<=	>	>=			left to right	relational
==	!=					left to right	equality
?:						right to left	conditional
=	+=	-=	*=	/=	%=	right to left	assignment

4.10 Increment and Decrement Operators (自增和自减运算符)

```
int c=5,x=2;
cout<<x+c++<<endl;
cout<<c<<endl;
```

```
7
6
```

```
int c=5,x=2;
cout<<x+++c<<endl;
cout<<c<<endl;
cout<<x<<endl;
```

```
7
5
3
```

Summary

- 理解算法、伪代码、程序的概念
- 掌握“自顶向下，逐步求精”的结构化编程方法
- 使用if、if..else选择结构进行选择操作
- 使用while循环结构
- 理解嵌套的概念
- 使用计数器控制循环与标记控制循环
- 使用自增、自减、赋值运算符

Homework

- 实验必选题目：
- 4.14 , 4.19, 4.26, 4.34

- 实验任选题目：

  4.35
- 作业题目(Homework)：
- 4.11, 4.23, 4.24