

bca, 可以指出它的前一排列是 bac, 它的下一排列是 cab。序列 abc 之前没有排列, cba 之后也没有下一排列。

标准库提供两个排列算法, 按字典序产生序列的排列。这些算法重新排列序列, 以便(按字典序)保存给定序列的下一个或前一个排列。它们返回指出是否存在下一个或前一个排列的 bool 值。

每个算法有两个版本: 一个使用元素类型的<操作符, 另一个接受指定用于比较元素的比较关系的实参。这些算法假定序列中的元素是唯一的, 也就是说, 算法假定序列中没有两个元素具有相同值。

要求双向迭代器的排列算法

为了产生排列, 必须对序列进行前向和后向处理, 因此要求双向迭代器。

```
next_permutation(beg, end)
next_permutation(beg, end, comp)
```

如果序列已经是在最后一个排列中, 则 next_permutation 将序列重新排列为最低排列并返回 false; 否则, 它将输入序列变换为下一个排列, 即字典序的下一个排列, 并返回 true。第一个版本使用元素的<操作符比较元素, 第二个版本使用指定的比较关系。

```
prev_permutation(beg, end)
prev_permutation(beg, end, comp)
```

与 next_permutation 很像, 但变换序列以形成前一个排列。如果这是最小的排列, 则它将序列重新排列为最大排列, 并返回 false。

A.2.8 有序序列的集合算法

集合算法实现有序序列的通用集合运算。



这些算法不同于标准库中的 set 容器, 不应该与 set 的操作相混淆, 相反, 这些算法提供普通顺序容器 (vector, list, 等等) 或其他序列 (如输入流) 上的集合式行为。

除了 includes 之外, 它们也接受输出迭代器。程序员照常必须保证目的地足以保存生成的序列。这些算法返回它们的 dest 迭代器, dest 迭代器增量至指向紧接在写至 dest 的最后一个元素之后的元素。

每个算法都提供两种形式: 第一种形式使用元素类型的<操作符比较两个输入序列中的元素, 第二种形式接受一个用于比较元素的比较关系。

要求输入迭代器的集合算法

这些算法顺序处理元素, 要求输入迭代器。

```
includes(beg, end, beg2, end2)
includes(beg, end, beg2, end2, comp)
```

如果输入序列包含第二个序列中的每个元素, 就返回 true; 否则, 返回 false。

```
set_union(beg, end, beg2, end2, dest)
set_union(beg, end, beg2, end2, dest, comp)
```

创建在任一序列中存在的元素的有序序列。两个序列中都存在的元素在输出序列中只出现一次。将序列存储在 dest 中。

```
set_intersection(beg, end, beg2, end2, dest)
set_intersection(beg, end, beg2, end2, dest, comp)
```

创建在两个序列中都存在的元素的有序序列。将序列存储在 dest 中。

```
set_difference(beg, end, beg2, end2, dest)
set_difference(beg, end, beg2, end2, dest, comp)
```

创建在第一个容器中但不在第二个容器中的元素的有序序列。

```
set_symmetric_difference(beg, end, beg2, end2, dest)
set_symmetric_difference(beg, end, beg2, end2, dest, comp)
```

创建在任一容器中存在但不在两个容器中同时存在的元素的有序序列。

A.2.9 最大值和最小值

这些算法的第一组在标准库中是独特的，它们操作值而不是序列。第二组接受一个由输入迭代器表示的序列。

```
min(val1, val2)
min(val1, val2, comp)
max(val1, val2)
max(val1, val2, comp)
```

返回 val1 和 val2 的最大值/最小值。实参必须是完全相同的类型。使用元素类型的 < 操作符或指定的比较关系。实参和返回类型都是 const 引用，表示对象不是复制的。

```
min_element(beg, end)
min_element(beg, end, comp)
max_element(beg, end)
max_element(beg, end, comp)
```

822 返回指向输入序列中最小/最大元素的迭代器。使用元素类型的 < 操作符或指定的比较关系。

字典序比较关系

字典序比较关系检查两个序列中的对应元素，并基于第一个不相等的元素对确定比较关系。因为算法顺序地处理元素，所以它们要求输入迭代器。如果一个序列比另一个短，并且它的元素与较长序列中对应元素相匹配，则较短的序列在字典序上较小。如果序列长短相同且对应元素匹配，则在字典序上两者都不小于另一个。

```
lexicographical_compare(beg1, end1, beg2, end2)
lexicographical_compare(beg1, end1, beg2, end2, comp)
```

对两个序列中的元素进行逐个比较。如果第一个序列在字典序上小于第二个序列，就返回 true；否则，返回 false。使用元素类型的 < 操作符或指定的比较关系。

A.2.10 算术算法

算术算法要求输入迭代器，如果算法修改输出，它就使用目的地的输出迭代器。

这些算法执行它们的输入序列的简单算术操纵。要使用算术算法必须包含头文件 numeric。

```
accumulate(beg, end, init)
accumulate(beg, end, init, BinaryOp)
```

返回输入范围中所有值的总和。求和从指定的初始值 `init` 开始。返回类型是与 `init` 相同的类型。

给定序列 1, 1, 2, 3, 5, 8 以及初始值 0, 结果是 20。第一个版本应用元素类型的+操作符, 第二个版本应用指定的二元操作符。

```
inner_product(beg1, end1, beg2, init)
inner_product(beg1, end1, beg2, init, BinOp1, BinOp2)
```

返回作为两个序列乘积而生成的元素的总和。步调一致地检查两个序列, 将来自两个序列的元素相乘, 将相乘的结果求和。由 `init` 指定和的初值。假定从 `beg2` 开始的第二个序列具有至少与第一个序列一样多的元素, 忽略第二个序列中超出第一个序列长度的任何元素。`init` 的类型决定返回类型。

第一个版本使用元素的乘操作符 (*) 和加操作符 (+): 给定两个序列 2, 3, 5, 8 和 1, 2, 3, 4, 5, 6, 7, 结果是初值加上下面的乘积对:

```
initial_value + (2 * 1) + (3 * 2) + (5 * 3) + (8 * 4)
```

如果提供初值 0, 则结果是 55。

第二个版本应用指定的二元操作, 使用第一个操作代替加而第二个操作代替乘。作为例子, 可以使用 `inner_product` 来产生以括号括住的元素的名-值对的列表, 这里从第一个输入序列获得名字, 从第二个序列中获得对应的值:

```
// combine elements into a parenthesized, comma-separated pair
string combine(string x, string y)
{
    return "(" + x + ", " + y + ")";
}
// add two strings, each separated by a comma
string concatenate(string x, string y)
{
    if (x.empty())
        return y;
    return x + ", " + y;
}

cout << inner_product(names.begin(), names.end(),
                      values.begin(), string(),
                      concatenate, combine);
```

如果第一个序列包含 `if`、`string` 和 `sort`, 且第二个序列包含 `keyword`、`library type` 和 `algorithm`, 则输出将是

```
(if, keyword), (string, library type), (sort, algorithm)
```

```
partial_sum(beg, end, dest)
partial_sum(beg, end, dest, BinaryOp)
```

将新序列写至 `dest`, 其中每个新元素的值表示输入范围中在它的位置之前 (不包括它的位置) 的所有元素的总和。第一个版本使用元素类型的+操作符, 第二个版本应用指定的二元操作符。程序员必须保证 `dest` 至少与输入序列一样大。返回 `dest`, `dest` 增量到指向被写入的最后

元素的下一位置。

给定输入序列 0, 1, 1, 2, 3, 5, 8, 目的序列将是 0, 1, 2, 4, 7, 12, 20。例如, 第四个元素是前三个值 (0, 1, 1) 的部分和加上它自己的值 (2), 获得值 4。

```
adjacent_difference(beg, end, dest)
```

```
adjacent_difference(beg, end, dest, BinaryOp)
```

将新序列写至 `dest`, 其中除了第一个元素之外每个新元素表示当前元素和前一元素的差。第一个版本使用元素类型的 `-` 操作符, 第二个版本应用指定的二元操作。程序员必须保证 `dest` 至少与输入序列一样大。

给定序列 0, 1, 1, 2, 3, 5, 8, 新序列的第一个元素是原序列第一个元素的副本: 0, 第二个元素是前两个元素的差: 1, 第三个元素是原序列第三个元素和第二个元素的差, 为 0, 以此类推, 新序列是 0, 1, 0, 1, 1, 2, 3。

824

A.3 再谈 IO 库

第 8 章介绍过 IO 库的基本体系结构以及最常使用的部分, 本附录完成对 IO 库的讨论。

A.3.1 格式状态

除了条件状态 (8.2 节) 之外, 每个 `iostream` 对象还维持一个控制 IO 格式化细节的格式状态。格式状态控制格式化特征, 如整型值的基数、浮点值的精度、输出元素的宽度等。标准库还定义了一组操纵符 (在表 A-2 和表 A-3 列出) 来修改对象的格式状态。简单说来, 操纵符是可用作输入或输出操作符操作数的函数或对象。操纵符返回其应用于的流对象, 所以可以在一个语句中输出多个操纵符和数据。

表 A-2 `iostream` 中定义的操纵符

<code>boolalpha</code>	将真和假显示为字符串
<code>x noboolalpha</code>	将真和假显示为 1, 0
<code>showbase</code>	产生指出数的基数的前缀
<code>x noshowbase</code>	不产生记数基数前缀
<code>showpoint</code>	总是显示小数点
<code>x noshowpoint</code>	有小数部分才显示小数点
<code>showpos</code>	显示非负数中的 +
<code>x noshowpos</code>	不显示非负数中的 +
<code>uppercase</code>	在十六进制中打印 0X, 科学记数法中打印 E
<code>x nouppercase</code>	在十六进制中打印 0x, 科学记数法中打印 e
<code>x dec</code>	用十进制显示
<code>hex</code>	用十六进制显示
<code>oct</code>	用八进制显示
<code>left</code>	在值的右边增加填充字符

(续)

right	在值的左边增加填充字符
internal	在符号和值之间增加填充字符
fixed	用小数形式显示浮点数
scientific	用科学记数法显示浮点数
flush	刷新 ostream 缓冲区
ends	插入空字符，然后刷新 ostream 缓冲区
endl	插入换行符，然后刷新 ostream 缓冲区
unitbuf	在每个输出操作之后刷新缓冲区
x nounitbuf	恢复常规缓冲区刷新
x skipws	为输入操作符跳过空白
noskipws	不为输入操作符跳过空白
ws	“吃掉”空白

注：带x的是默认流状态。

表 A-3 iomanip 中定义的操纵符

setfill(ch)	用 ch 填充空白
setprecision(n)	将浮点精度置为 n
setw(w)	读写 w 个字符的值
setbase(b)	按基数 b 输出整数

读写操纵符的时候，不读写数据，相反，会采取某种行动。示例程序已经使用过一个操纵符——endl，我们将它“写”至输出流，就好像它是一个值一样。但 endl 并不是一个值，相反，它执行一个操作：写换行符并刷新缓冲区。

A.3.2 许多操纵符改变格式状态

许多操纵符改变流的格式状态。它们改变显示浮点数的格式，将 bool 值显示为数值还是使用 bool 字面值 true 和 false 的格式，诸如此类。



改变流格式状态的操纵符通常为后续 IO 保留改变后的格式状态。

大多数改变格式状态的操纵符提供设置/复原对，一个操纵符将格式状态置为新值而另一个进行复原，恢复常规默认格式。

操纵符进行格式状态的持久改变，在有一组 IO 操作希望使用相同格式化的时候，这一事实有用。事实上，一些程序利用操纵符的这个特征，为自己的所有输入或输出重置一个或多个格式化规则的行为。这种情况下，操纵符改变流是希望得到的性质。

但是，许多程序（更重要的是，许多程序员）希望流的状态与标准库默认值匹配。这些情况下，使流状态停留在非标准状态可能会导致错误。

取消操纵符的任何状态改变通常是最好的。一般而言，流应该在每个 IO 操作之后处于通常的默认状态。

用 flags 操作恢复格式状态

管理格式状态改变的一个更好的办法是使用 flags 操作。flags 操作类似于管理流的条件状态的 rdstate 和 setstate 操作。这种情况下，标准库定义了一对 flags 函数：

- 不带实参的 flags() 返回流的当前格式状态。返回值是名为 fmtflags 的标准库定义类型。
- flags(arg) 接受一个实参并将流格式置为实参所指定的格式。

可以使用这些函数记住并恢复输入或输出流的格式状态：

```
void display(ostream& os)
{
    // remember the current format state
    ostream::fmtflags curr_fmt = os.flags();
    // do output that uses manipulators that change the format state of os
    os.flags(curr_fmt); // restore the original format state of os
}
```

A.3.3 控制输出格式

许多操纵符使我们能够改变输出的外观。有两大类的输出控制：控制数值的表示，以及控制填充符的数量和布局。

1. 控制布尔值的格式

改变对象格式化状态的操纵符的一个例子是 boolalpha 操纵符。默认情况下，将 bool 值显示为 1 或 0，true 值显示为 1，而 false 值显示为 0。可以通过流的 boolalpha 操纵符覆盖这个格式化：

```
cout << "default bool values: "
      << true << " " << false
      << "\nalpha bool values: "
      << boolalpha
      << true << " " << false
      << endl;
```

执行时，这段程序产生下面的输出：

```
default bool values: 1 0
alpha bool values: true false
```

一旦将 boolalpha“写”至 cout，从这个点起就改变了 cout 将怎样显示 bool 值，后续显示 bool 值的操作将用 true 或 false 进行显示。

要取消 cout 的格式状态改变，必须应用 noboolalpha：

```
bool bool_val;
cout << boolalpha // sets internal state of cout
     << bool_val
     << noboolalpha; // resets internal state to default formatting
```

现在只改变 bool 值的格式化来显示 bool_val，并且立即将流重置为原来的状态。

2. 指定整型值的基数

默认情况下，用十进制读写整型值。通过使用操纵符 hex、oct 和 dec，程序员可以将表示进制改为八进制、十六进制或恢复十进制（浮点值的表示不受影响）：

```
const int ival = 15, jval = 1024; // const, so values never change
cout << "default: ival = " << ival
    << " jval = " << jval << endl;
cout << "printed in octal: ival = " << oct << ival
    << " jval = " << jval << endl;
cout << "printed in hexadecimal: ival = " << hex << ival
    << " jval = " << jval << endl;
cout << "printed in decimal: ival = " << dec << ival
    << " jval = " << jval << endl;
```

编译和执行的时候，程序产生下面的输出：

```
default: ival = 15 jval = 1024
printed in octal: ival = 17 jval = 2000
printed in hexadecimal: ival = f jval = 400
printed in decimal: ival = 15 jval = 1024
```

注意，像 boolalpha 一样，这些操纵符改变格式状态。它们影响紧接在后面的输出，以及所有后续的整型输出，直到通过调用另一操纵符重置格式为止。

3. 指出输出的基数

默认情况下，显示数值的时候，不存在关于所用基数的可见记号。例如，20 是 20，还是 16 的八进制表示？按十进制模式显示数值的时候，会按我们期待的格式打印数值。如果需要打印八进制或十六进制值，可能应该也使用 showbase 操纵符。showbase 操纵符导致输出流使用的约定，与指定整型常量基数所用的相同：

- 以 0x 为前导表示十六进制。
- 以 0 为前导表示八进制。
- 没有任何前导表示十进制。

修改程序使用 showbase 如下：

```
const int ival = 15, jval = 1024; // const so values never change
cout << showbase; // show base when printing integral values
cout << "default: ival = " << ival
    << " jval = " << jval << endl;
cout << "printed in octal: ival = " << oct << ival
    << " jval = " << jval << endl;
cout << "printed in hexadecimal: ival = " << hex << ival
    << " jval = " << jval << endl;
cout << "printed in decimal: ival = " << dec << ival
    << " jval = " << jval << endl;
cout << noshowbase; // reset state of the stream
```

修改后的输出使得基础值到底是什么很清楚：

```
default: ival = 15 jval = 1024
printed in octal: ival = 017 jval = 02000
```

```
printed in hexadecimal: ival = 0xf jval = 0x400
printed in decimal: ival = 15 jval = 1024
```

noshowbase 操纵符重置 cout，以便它不再显示整型值的表示基数。

默认情况下，十六进制值用带小写 x 的小写形式打印。可以应用 uppercase 操纵符显示 x 并将十六进制数字 a~f 显示为大写字母。

```
cout << uppercase << showbase << hex
    << "printed in hexadecimal: ival = " << ival
    << " jval = " << jval << endl
    << nouppercase << endl;
```

前面的程序产生下面的输出：

```
printed in hexadecimal: ival = 0XF jval = 0X400
```

要恢复小写，就应用 nouppercase 操纵符。

4. 控制浮点值的格式

对于浮点值的格式化，可以控制下面三个方面：

- 精度：显示多少位数字。
- 记数法：用小数还是科学记数法显示。
- 对是整数的浮点值的小数点的处理。

默认情况下，使用六位数字的精度显示浮点值。如果值没有小数部分，则省略小数点。使用小数形式还是科学记数法显示数值取决于被显示的浮点数的值，标准库选择增强数值可读性的格式，非常大和非常小的值使用科学记数法显示，其他值使用小数形式。

828

5. 指定显示精度

默认情况下，精度控制显示的数字总位数。显示的时候，将浮点值四舍五入到当前精度。因此，如果当前精度是 4，则 3.14159 成为 3.142；如果精度是 3，打印为 3.14。

通过名为 precision 的成员函数，或者通过使用 setprecision 操纵符，可以改变精度。precision 成员是重载的（7.8 节）：一个版本接受一个 int 值并将精度设置为那个新值，它返回先前的精度值；另一个版本不接受实参并返回当前精度值。setprecision 操纵符接受一个实参，用来设置精度。

829

下面的程序说明控制显示浮点值所用精度的不同方法：

```
// cout.precision reports current precision value
cout << "Precision: " << cout.precision()
    << ", Value: " << sqrt(2.0) << endl;
// cout.precision(12) asks that 12 digits of precision to be printed
cout.precision(12);
cout << "Precision: " << cout.precision()
    << ", Value: " << sqrt(2.0) << endl;
// alternative way to set precision using setprecision manipulator
cout << setprecision(3);
cout << "Precision: " << cout.precision()
    << ", Value: " << sqrt(2.0) << endl;
```

编译并执行之后，程序产生下面的输出：

```
Precision: 6, Value: 1.41421
Precision: 12, Value: 1.41421356237
Precision: 3, Value: 1.41
```

这个程序调用标准库中的 `sqrt` 函数，可以在头文件 `cmath` 中找到它。`sqrt` 函数是重载的，可以用 `float`、`double` 或 `long double` 实参调用，它返回实参的平方根。



操纵符和其他接受实参的操纵符定义在头文件 `iomanip` 中。

6. 控制记数法

默认情况下，用于显示浮点值的记数法取决于数的大小：如果数很大或很小，将按科学记数法显示，否则，使用固定位数的小数。标准库选择使得数容易阅读的记数法。



将浮点数显示为普通数（相对于显示货币、百分比，那时我们希望控制值的外观）的时候，通常最好让标准库来选择使用的记数法。要强制科学记数法或固定位数小数的一种情况是在显示表的时候，表中的小数点应该对齐。

如果希望强制科学记数法或固定位数小数表示，可以通过使用适当的操纵符做到这一点：`scientific` 操纵符将流变为使用科学记数法。像在十六进制值上显示 `x` 一样，也可以通过 `uppercase` 操纵符控制科学记数法中的 `e`。`fixed` 操纵符将流变为使用固定位数小数表示。

这些操纵符改变流精度的默认含义。执行 `scientific` 或 `fixed` 之后，精度值控制小数点之后的数位。默认情况下，精度指定数字的总位数——小数点之前和之后。使用 `fixed` 或 `scientific` 使我们能够按列对齐来显示数，这一策略保证小数点总是在相对于被显示的小数部分固定的位置。

830

7. 恢复浮点值的默认记数法

与其他操纵符不同，不存在将流恢复为根据被显示值选择记数法的默认状态的操纵符，相反，我们必须调用 `unsetf` 成员来取消 `scientific` 或 `fixed` 所做的改变。要将流恢复为浮点值的默认处理，将名为 `floatfield` 的标准库定义值传给 `unsetf` 函数：

```
// reset to default handling for notation
cout.unsetf(ostream::floatfield);
```

除了取消它们的效果之外，使用这些操纵符像使用任意其他操纵符一样：

```
cout << sqrt(2.0) << '\n' << endl;
cout << "scientific: " << scientific << sqrt(2.0) << '\n'
    << "fixed decimal: " << fixed << sqrt(2.0) << "\n\n";
cout << uppercase
    << "scientific: " << scientific << sqrt(2.0) << '\n'
    << "fixed decimal: " << fixed << sqrt(2.0) << endl
    << nouppercase;
// reset to default handling for notation
cout.unsetf(ostream::floatfield);
cout << '\n' << sqrt(2.0) << endl;
```

产生如下输出：

```
1.41421
```

```
scientific: 1.414214e+00
fixed decimal: 1.414214
```

```
scientific: 1.414214E+00
fixed decimal: 1.414214
```

```
1.41421
```

8. 显示小数点

默认情况下，当浮点值的小数部分为 0 的时候，不显示小数点。showpoint 操纵符强制显示小数点：

```
cout << 10.0 << endl;           // prints 10
cout << showpoint << 10.0       // prints 10.0000
    << noshowpoint << endl; // revert to default handling of decimal point
```

831

noshowpoint 操纵符恢复默认行为。下一个输出表达式将具有默认行为，即，如果浮点值小数部分为 0，就取消小数点。

9. 填充输出

按栏显示数据的时候，经常很希望很好地控制数据的格式化。标准库提供下面几个操纵符（图 A-3）帮助我们实现需要的控制：

- setw, 指定下一个数值或字符串的最小间隔。
- left, 左对齐输出。
- right, 右对齐输出。输出默认为右对齐。
- internal, 控制负值的符号位置。internal 左对齐符号且右对齐值，用空格填充介于其间的空间。
- setfill, 使我们能够指定填充输出时使用的另一个字符。默认情况下，值是空格。



像 endl 一样，setw 不改变输出流的内部状态，它只决定下一个输出的长度。

下面程序段说明了这些操纵符：

```
int i = -16;
double d = 3.14159;
// pad first column to use minimum of 12 positions in the output
cout << "i: " << setw(12) << i << "next col" << '\n'
    << "d: " << setw(12) << d << "next col" << '\n';
// pad first column and left-justify all columns
cout << left
    << "i: " << setw(12) << i << "next col" << '\n'
    << "d: " << setw(12) << d << "next col" << '\n'
    << right; // restore normal justification
// pad first column and right-justify all columns
cout << right
    << "i: " << setw(12) << i << "next col" << '\n'
    << "d: " << setw(12) << d << "next col" << '\n';
// pad first column but put the padding internal to the field
cout << internal
```

```

    << "i: " << setw(12) << i << "next col" << '\n'
    << "d: " << setw(12) << d << "next col" << '\n';
// pad first column, using # as the pad character
cout << setfill('#')
    << "i: " << setw(12) << i << "next col" << '\n'
    << "d: " << setw(12) << d << "next col" << '\n'
    << setfill(' '); // restore normal pad character

```

832

执行时，该程序段产生如下输出：

```

i:          -16next col
d:         3.14159next col
i: -16          next col
d: 3.14159      next col
i:          -16next col
d:         3.14159next col
i: -          16next col
d:         3.14159next col
i: -#####16next col
d: #####3.14159next col

```

A.3.4 控制输入格式化

默认情况下，输入操作符忽略空白（空格、制表符、换行符、进纸和回车）。对下面的循环：

```

while (cin >> ch)
    cout << ch;

```

给定输入序列

```

a b c
d

```

循环执行四次从字符 a 读到 d，跳过介于其间的空格、可能的制表符和换行符。该程序段的输出是：

```

abcd

```

`noskipws` 操纵符导致输入操作符读（而不是跳过）空白。要返回默认行为，应用 `skipws` 操纵符：

```

cin >> noskipws;           // set cin so that it reads whitespace
while (cin >> ch)
    cout << ch;
cin >> skipws;             // reset cin to default state so that it discards whitespace

```

给定与前面相同的输入，该循环进行 7 次迭代，读输入中的空白以及字符。该循环产生如下输出：

```

a b c
d

```

833

A.3.5 未格式化的输入/输出操作

迄今为止，示例程序中只使用过格式化的 IO 操作。输入和输出操作符（>>和<<）根据被处理数据的类型格式化所读写的数据。输入操作符忽略空白，输出操作符应用填充、精度等。

标准库还提供了丰富的支持未格式化 IO 的低级操作，这些操作使我们能够将流作为未解释的字节序列处理，而不是作为数据类型（如 char、int、string 等）的序列处理。

A.3.6 单字节操作

几个未格式化的操作（表 A-4）一次一个字节地处理流，它们不忽略空白地读。例如，可以使用未格式化 IO 操作 get 和 put 一次读一个字符：

```
char ch;
while (cin.get(ch))
    cout.put(ch);
```

该程序段保持输入中的空白。它的输出与输入相同。给定与前面使用 noskipws 的程序段相同的输入，该程序段产生相同的输出：

```
a b c
d
```

表 A-4 单字节低级 IO 操作	
is.get(ch)	将 istream is 的下一个字节放入 ch，返回 is
os.put(ch)	将字符 ch 放入 ostream，返回 os
is.get()	返回 is 的下一字节作为一个 int 值
is.putback(ch)	将字符 ch 放回 is，返回 is
is.unget()	将 is 退回一个字节，返回 is
is.peek()	将下一字节作为 int 值返回但不移出它

1. 在输入流上倒退

有时我们需要读一个字符才知道还没有为它作好准备，在这种情况下，希望将字符放回流上。标准库给出三种方法做到这一点，它们之间有下列微妙的区别：

- peek，返回输入流上下一字符的副本但不改变流。peek 返回的值留在流上，且将是下一个被检索的。
- unget，使输入流倒退，以便最后返回的值仍在流上。即使不知道最近从流获得的是什么值，也可以调用 unget。
- putback，unget 的更复杂的版本。它返回从流中读到的最后一个值，但接受最后一次读的同—实参。很少有程序使用 putback，因为更简单的 unget 也可以完成相同工作而约束更少。

一般而言，保证能够在下一次读之前放回最多一个值，也就是说，不保证能够连续调用 putback 或 unget 而没有介于其间的读操作。

2. 输入操作的 int 返回值

不接受实参的 get 版本和 peek 函数从输入流返回一个字符作为 int 值。这个事实可能令人惊讶，因为这些函数返回 char 值似乎更自然。

这些函数返回 int 值的原因是为了允许它们返回一个文件结束标记。允许给定字符集使用

834

char 范围的每一个值来表示实际字符，因此，该范围中没有额外值用来表示文件结束符。

相反，这些函数将字符转换为 unsigned char，然后将那个值提升为 int，因此，即使字符集有映射到负值的字符，从这些操作返回的值也将是一个正值（2.1.1 节）。通过将文件结束符作为负值返回，标准库保证文件结束符区别于任意合法字符值。为了不要求我们知道返回的实际值，头文件 iostream 定义了名为 EOF 的 const，可以使用它来测试 get 的返回值是否为文件结束符。实质上我们使用 int 对象来保存这些函数的返回值：

```
int ch; // NOTE: int, not char!!!!
// loop to read and write all the data in the input
while ((ch = cin.get()) != EOF)
    cout.put(ch);
```

这个程序段与 A.3.6 节中的那个程序段同样操作，唯一的区别是用来读输入的 get 版本不同。

A.3.7 多字节操作

其他未格式化 IO 操作一次处理数据块。如果速度是一个问题，这些操作可能就很重要。但像其他低级操作一样，它们是容易出错的。尤其是，这些操作要求我们分配和管理用于存储和检索数据的字符数组（4.3.1 节）。

多字节操作在表 A-5 列出。值得注意的是，get 成员是重载的，存在读字符序列的第三个版本。

835

表 A-5 多字节低级 IO 操作	
is.get(sink, size, delim)	从 is 中读 size 个字节并将它们存储到 sink 所指向的字符数组中。读操作直到遇到 delim 字符，或已经读入了 size 个字节，或遇到文件结束符才结束。如果出现了 delim，就将它留在输入流上，不读入到 sink 中
is.getline(sink, size, delim)	与三个实参的 get 行为类似，但读并丢弃 delim
is.read(sink, size)	读 size 个字节到数组 sink。返回 is
is.gcount()	返回最后一个未格式化读操作从流 is 中读到的字节数
os.write(source, size)	将 size 个字节从数组 source 写至 os。返回 os
is.ignore(size, delim)	读并忽略至多 size 个字符，直到遇到 delim，但不包括 delim。默认情况下，size 是 1 而 delim 是文件结束符

警告：低级例程容易出错

一般提倡使用标准库所提供的高级抽象。返回 int 值的 IO 操作是一个很好的例子。将 get 或其他返回 int 值的函数的返回值赋给 char 对象而不是 int 对象，是常见的错误，但编译器不检测这样的错误，相反，发生什么取决于机器和输入数据。例如，在将 char 实现为 unsigned char 的机器上，这是一个死循环：

```
char ch; // Using a char here invites disaster!
// return from cin.get is converted from int to char and then compared to an int
while ((ch = cin.get()) != EOF)
    cout.put(ch);
```

问题在于，当 `get` 返回 EOF 的时候，那个值将被转换为 `unsigned char` 值，转换后的值不再等于 EOF 的整型值，循环将永远继续。

至少还可能在测试中捕获这个错误。在将 `char` 实现为 `signed char` 的机器上，不能确切地说出该循环的行为。当将超出边界的值赋给 `signed` 值时发生什么由编译器负责。在许多机器上，这个循环看起来能工作，除非输入中的字符与 EOF 值匹配。虽然这样的字符不可能在普通数据中，但是，大概只有在读不直接映射到普通字符和数值的二进制值的时候，低级 IO 才是必要的。例如，在我们的机器上，如果输入包含值为 `'\377'` 的字符，循环就提前终止。`'\377'` 是我们的机器上 -1 作为 `signed char` 使用的时候所转换的值，如果输入中有这个值，就将它当作（提早的）文件结束指示符对待。

在读写类型化值的时候不会发生这样的错误。如果可以使用标准库支持的更为类型安全的、更高级的操作，就使用。

`get` 函数和 `getline` 函数接受相同形参，它们的行为类似但不相同。在每个函数中，`sink` 是一个存放数据的 `char` 数组。函数进行读，直到下面条件中的一个发生：

- 读到了 `size-1` 个字符。
- 遇到文件结束符。
- 遇到分隔符。

遵循这些条件中的任意一个，将一个空字符放到数组中下一个开放位置。两个函数之间的区别是对待分隔符的方法。`get` 将分隔符留作 `istream` 的下一个字符，`getline` 读入并丢弃分隔符，两种情况下，都不在 `sink` 中存储分隔符。

836



想要从流中移去分隔符但忘了这样做，是一个常见的错误。

确定读了多少字符

有几个读操作中从输入中读未知数目的字节。可以调用 `gcount` 操作来确定最后一个未格式化输入操作读了多少字符。有必要在任意插入的未格式化输入操作之前调用 `gcount`。尤其是，将字符放回流上的单字符操作也是未格式化输入操作，如果在调用 `gcount` 之前调用 `peek`、`unget` 或 `putback`，则返回值将是 0！

A.3.8 流的随机访问

不同的流类型一般支持对相关流中数据的随机访问。可以重新定位流，以便环绕跳过，首先读最后一行，再读第一行，以此类推。标准库提供一对函数来定位（`seek`）给定位置并告诉（`tell`）相关流中的当前位置。



随机 IO 是一个固有的随系统而定的特征。要理解怎样使用这些特征，必须查阅系统的文档。

1. seek 和 tell 函数

为了支持随机访问，IO 类型维持一个标记，该标记决定下一个读或写发生在哪里。IO 类型还提供两个函数：一个通过定位指定位置重新安置该标记，另一个告诉我们标记的当前位置。标准库实际上定义了两对 *seek* 和 *tell* 函数（表 A-6 给出对它们的描述），一对由输入流使用，另一对由输出流使用。输入和输出版本由后缀 *g* 和 *p* 区分，*g* 版本指出正在“获取”（读）数据，*p* 函数指出正在“放置”（写）数据。

837

表 A-6 seek 和 tell 函数	
seekg	重新定位输入流中的标记
tellg	返回输入流中标记的当前位置
seekp	重新定位输出流中的标记
tellp	返回输出流中标记的当前位置

逻辑上，只能在 *istream* 类型或其派生类型 *ifstream* 或 *istringstream* 之上使用 *g* 版本，并且只能在 *ostream* 类型或其派生类型 *ofstream* 或 *ostringstream* 之上使用 *p* 版本。*iostream* 对象、*fstream* 对象或 *stringstream* 对象对相关流既能读又能写，可以在这些类型的对象上使用 *g* 或 *p* 版本的任何一个。

2. 只有一个标记

标准库区分 *seek* 函数和 *tell* 函数的“放置”和“获取”版本，这一事实可能会令人误解。虽然标准库进行这种区分，但它只在文件中维持一个标记——没有可区分的读标记和写标记。

处理只输入或只输出的流的时候，区别并不明显。在这样的流上，只能使用 *g* 版本或只能使用 *p* 版本。如果试图在 *ifstream* 对象上调用 *tellp*，编译器将会给出错误提示。类似地，编译器不允许在 *ostringstream* 对象上调用 *seekg*。

使用既能读又能写的 *fstream* 类型和 *stringstream* 类型的时候，只有一个保存数据的缓冲区和一个表示缓冲区中当前位置的标记，标准库将 *g* 位置和 *p* 位置都映射到这个标记。



因为只有一个标记，所以，在读和写之间切换时必须进行 *seek* 来重新定位标记。

3. 普通 *iostream* 对象一般不允许随机访问

seek 函数和 *tell* 函数是为所有流类型定义的，它们是否完成有用的工作取决于流所绑定的对象的类别。在大多数系统上，绑定到 *cin*、*cout*、*cerr* 和 *clog* 的流不支持随机访问——直接写 *cout* 的时候，回跳 10 个位置会意味着什么？可以调用 *seek* 函数和 *tell* 函数，但这些函数将在运行时失败，使得流处于无效状态。

838



因为 *istream* 和 *ostream* 类型一般不支持随机访问，所以，应该认为本节其余部分只能应用于 *fstream* 和 *sstream* 类型。

4. 重新定位标记

seekg 函数和 *seekp* 函数用于改变文件或 *string* 对象中的读写位置。调用 *seekg* 函数之后，

改变流中的读位置，seekp 函数调用将位置置于下一个写将发生的地方。
seek 函数有两个版本：一个移动到文件中的一个“绝对”地址，另一个移动到给定位置的字节偏移量处：

```
// set the indicated marker a fixed position within a file or string
seekg(new_position); // set read marker
seekp(new_position); // set write marker

// offset some distance from the indicated position
seekg(offset, dir); // set read marker
seekp(offset, dir); // set write marker
```

第一个版本将当前位置置于给定地点，第二个版本接受一个偏移量以及从何处偏移的指示器。偏移量的可能值在表 A-7 中列出。

表 A-7 seek 实参的偏移量	
beg	流的开头
cur	流的当前位置
end	流的末尾

这些函数的实参类型和返回类型是与机器相关的类型，在 istream 和 ostream 中定义。名为 pos_type 和 off_type 的类型分别表示文件位置和从该位置的偏移量。off_type 类型的值可以为正也可以为负，在文件中可以进行前向或后向 seek。

5. 访问标记

当前位置由 tellg 或 tellp 返回，取决于正在查找读位置还是写位置。像前面一样，p 指出放置（写）而 g 指出获取（读）。tell 函数一般用于记住一个位置，以便随后可以通过 seek 回到那里：

839

```
// remember current write position in mark
ostringstream writeStr; // output stringstream
ostringstream::pos_type mark = writeStr.tellp();
// ...
if (cancelEntry)
    // return to marked position
    writeStr.seekp(mark);
```

tell 函数返回一个值，指出相关流中的位置。像 string 对象或 vector 对象的 size_type 一样，我们不知道从 tellg 或 tellp 返回的对象的实际类型，而是使用适当流类的 pos_type 成员来代替。

A.3.9 读写同一文件

看一个程序示例，假定给定一个文件来读，我们将在文件的末尾写一个新行，该行包含每一行开头的相对位置。例如，给定下面的文件，

```
abcd
efg
hi
```

j

这段程序应产生修改过的文件如下：

```
abcd
efg
hi
j
5 9 12 14
```

注意，程序不必写第一行的偏移量——它总是出现在位置 0，程序应该显示对应于文件数据部分末尾的偏移量，也就是说，它应该记录输入末尾之后的位置，以便我们知道原始数据在何处结束以及我们的输出从何处开始。

通过编写一次读一行的循环可以编写这个程序：

```
int main()
{
    // open for input and output and pre-position file pointers to end of file
    fstream inOut("copyOut",
                  fstream::ate | fstream::in | fstream::out);
    if (!inOut) {
        cerr << "Unable to open file!" << endl;
        return EXIT_FAILURE;
    }
    // inOut is opened in ate mode, so it starts out positioned at the end,
    // which we must remember as it is the original end-of-file position
    ifstream::pos_type end_mark = inOut.tellg();
    inOut.seekg(0, fstream::beg); // reposition to start of the file
    int cnt = 0; // accumulator for byte count
    string line; // hold each line of input
    // while we haven't hit an error and are still reading the original data
    // and successfully read another line from the file
    while (inOut && inOut.tellg() != end_mark
           && getline(inOut, line))
    {
        cnt += line.size() + 1; // add 1 to account for the newline
        // remember current read marker
        ifstream::pos_type mark = inOut.tellg();
        inOut.seekp(0, fstream::end); // set write marker to end
        inOut << cnt; // write the accumulated length
        // print separator if this is not the last line
        if (mark != end_mark) inOut << " ";
        inOut.seekg(mark); // restore read position
    }
    inOut.clear(); // clear flags in case we hit an error
    inOut.seekp(0, fstream::end); // seek to end
    inOut << "\n"; // write a newline at end of file
    return 0;
}
```

840

这个程序使用 in、out 和 ate 模式打开 fstream。前两种模式指出，我们打算对同一文件进行读写；通过用 ate 模式打开，文件首先定位于末端。照常，要检查打开是否成功，如果不成功就退出。

1. 初始设置

这段程序的核心部分将循环通过文件，一次一行，循环过程中记录每一行的相对位置。循环应该读文件的内容，直到但不包含正在增加的保存行偏移量的那一行。因为将对文件进行写，所以不是在遇到文件结束符时停止循环，相反，应在到达原始输入结束处结束循环。要做到这一点，必须首先记住原来文件结束符的位置。

因为是以 `ate` 模式打开文件，所以文件已经定位在末端。将原来的末端位置存储在 `end_mark` 中。当然，记住了结束位置之后，在试图读任意数据之前，必须将读标记重新定位于文件开头。

2. 主处理循环

`while` 循环有三部分条件。

首先检查流是否有效。假定第一个测试 `inOut` 成功，接着检查是否已经耗尽了原始输入，通过将 `tellg` 返回的当前读位置与 `end_mark` 中记录的位置相比较，进行这个检查。最后，假定两个测试都成功，就调用 `getline` 来读下一行输入，如果 `getline` 成功，就执行循环体。

841

`while` 所做的工作是将计数器增量，以确定下一行开始处的偏移量，并将那个标记写至文件末尾。注意，每通过一次循环都向前推进文件的末尾。

首先将当前位置记在 `mark` 中，需要保存那个值，是因为要写下一个相对偏移量必须重新定位文件。`seekp` 的调用进行这个重新定位，将文件指针重置到文件末尾。写计数器值然后将文件位置恢复为 `mark` 中所记录的值，效果是将标记返回到最后一次读之后的同一地方。恢复了标记之后，就准备重复检测 `while` 中的条件。

3. 完成文件

一旦退出了循环，就已经读过了每一行并计算了所有行开头的偏移量。剩下的是显示最后一行的偏移量。像对其他写操作一样，调用 `seekp` 将文件定位在末尾，并写 `cnt` 的值。唯一复杂的部分是记得对流进行 `clear`。可能因为文件结束符或其他输入错误而退出循环，如果是这样，

842

`inOut` 将处于错误状态，而 `seekp` 和输出表达式都将失败。

索引

索引中的页码为英文原书的页码，与书中边栏的页码一致。另外，粗体数字指的是第一次定义该术语的页码，斜体数字指的是各章后“术语”部分定义该术语的页码。

- ... (ellipsis parameter) (省略符形参), 244
- /* */ (block comment) (块注释), 10, 28
- // (single-line comment) (单行注释), 10, 28
- _ _DATE_ _ , 221
- _ _FILE_ _ , 221
- _ _LINE_ _ , 221
- _ _TIME_ _ , 221
- _ _cplusplus , 803
- \0 (null character) (空字符), 40
- \Xnnn (hexadecimal escape sequence) (十六进制转义字符), 40
- \n (newline character) (换行符), 40
- \t (tab character) (制表符), 40
- { } (curly brace) (花括号), 3, 28
- #define, 71
- #ifdef, 71
- #ifndef, 71
- #include, 7
- ~classname, ~类名, 见 destructor.
- L'c'(wchar_t literal) (wchar_t 字面值), 40
- ctrl-d (Unix end-of-file) (Unix 文件结束符), 20
- ctrl-z (Windows end-of-file) (Windows 文件结束符), 20
- ;(semicolon) (分号), 3
 - class definition (类定义), 440
- ++ (increment) (自增), 13, 30, 146, 190
 - and dereference (和解引用), 163
 - iterator (迭代器), 98, 108, 312
 - overloaded operator (重载操作符), 527
 - pointer (指针), 114
 - prefix yields lvalue (前缀生成左值), 162
 - reverse iterator (反向迭代器), 412
- (decrement) (自减), 190
 - and dereference (和解引用), 163
 - iterator (迭代器), 312
 - overloaded operator (重载操作符), 527
 - prefix yields lvalue (前缀生成左值), 162
 - reverse iterator (反向迭代器), 412
- * (dereference) (解引用), 98, 108, 146, 389
 - iterator (迭代器), 98
 - on map yields pair (用于 map, 生成 pair), 362
- overloaded operator (重载操作符), 524
- pointer (指针), 119
 - yields lvalue (生成左值), 99, 120
- & (address-of) (取地址), 115, 146, 511
- > (arrow operator) (箭头操作符), 164
 - class member access (类成员访问), 445
 - overloaded operator (重载操作符), 525
- >* (pointer to member arrow) (成员指针箭头操作符), 783
- [] (subscript) (下标), 87, 108, 146
 - bitset, 105
 - deque, 325
 - map, 363
 - string, 87
 - vector, 94, 325
- and multi-dimensioned array (和多维数组), 142
- and pointer (和指针), 124
- array (数组), 113
- overloaded operator (重载操作符), 522
 - reference return (引用返回), 522
- valid subscript range (有效下标范围), 88
- yields lvalue (生成左值), 88
- () (call operator) (调用操作符), 25, 30, 226, 280
 - overloaded operator (重载操作符), 530
- :: (scope operator) (作用域操作符), 8, 30, 78, 108
- base class members (基类成员), 569
- class member (类成员), 85, 445
- container defined type (容器定义的类型), 317
- member function definition (成员函数定义), 262
- to override name lookup (覆盖名字查找), 449
- = (assignment) (赋值), 13, 30, 159
 - and conversion (和转换), 179
 - and equality (和相等), 161
 - class assignment operator (类赋值操作符), 476
- container (容器), 328
- overloaded operator (重载操作符), 483, 520
 - and copy constructor (和复制构造函数), 484
 - check for self-assignment (检查自身赋值), 490

- Message, 490
 - multiple inheritance (多重继承), 737
 - reference return (引用返回), 483, 521
 - rule of three (三法则), 485
 - use counting (使用计数), 495, 498
 - valuelike classes (值型类), 501
 - pointer (指针), 120
 - string, 86
 - to signed (指向 signed 的), 37
 - to unsigned (指向 unsigned 的), 37
 - yields lvalue (生成左值), 160
 - += (compound assignment) (复合赋值), 13, 30, 161
 - string, 86
 - iterator (迭代器), 313
 - overloaded operator (重载操作符), 511
 - Sales_item, 521
 - + (addition) (加), 150
 - string, 86
 - iterator (迭代器), 101, 313
 - pointer (指针), 123
 - Sales_item, 517
 - (subtraction) (减), 150
 - iterator (迭代器), 101, 313
 - pointer (指针), 123
 - * (multiplication) (乘), 150
 - / (division) (除), 150
 - % (modulus) (模), 151
 - == (equality) (相等), 30, 154
 - string, 85
 - algorithm (算法), 421
 - container (容器), 322
 - container adaptor (容器适配器), 350
 - iterator (迭代器), 98, 312
 - string, 347
 - != (inequality) (不相等), 30, 154
 - container (容器), 322
 - container adaptor (容器适配器), 350
 - iterator (迭代器), 98, 312
 - string, 347
 - < (less-than) (小于), 30, 153
 - overloaded and containers (重载与容器), 520
 - used by algorithm (算法使用的), 420
 - <= (less-than-or-equal) (小于或等于), 13, 30, 153
 - > (greater-than) (大于), 30, 153
 - >= (greater-than-or-equal) (大于或等于), 30, 153
 - >> (input operator) (输入操作符), 8, 30
 - Sales_item, 516
 - istream_iterator, 408
 - string, 81, 108
 - overloaded operator (重载操作符), 515
 - must be nonmember (必须为非成员), 514
 - precedence and associativity (优先级和结合性), 158
 - << (output operator) (输出操作符), 7, 30
 - bitset, 106
 - ostream_iterator, 408
 - string, 81, 108
 - formatting (格式化), 514
 - overloaded operator (重载操作符), 513
 - must be nonmember (必须为非成员), 514
 - precedence and associativity (优先级和结合性), 158
 - Sales_item, 514
 - >> (right-shift) (右移), 155, 190
 - << (left-shift) (左移), 155, 190
 - && (logical AND) (逻辑与), 152
 - operand order of evaluation (操作数求值顺序), 172
 - overloaded operator (重载操作符), 511
 - || (logical OR) (逻辑或), 152
 - operand order of evaluation (操作数求值顺序), 172
 - overloaded operator (重载操作符), 511
 - & (bitwise AND) (位与), 156, 189
 - Query, 610
 - ! (logical NOT) (逻辑非), 153
 - | (bitwise OR) (位或), 156, 190
 - example (例子), 290
 - Query, 610
 - ^ (bitwise XOR) (位异或), 156, 190
 - ~ (bitwise NOT) (位求反), 155, 189
 - Query, 610
 - , (comma operator) (逗号操作符), 168, 189
 - example (例子), 289
 - operand order of evaluation (操作数求值顺序), 172
 - overloaded operator (重载操作符), 511
 - ?: (conditional operator) (条件操作符), 165, 189
 - operand order of evaluation (操作数求值顺序), 172
 - + (unary plus) (一元加), 150
 - (unary minus) (一元减), 150
 - \nnn (octal escape sequence) (八进制转义序列), 40
 - ddd.ddd L or ddd.ddd l (long double literal) (long double 字面值), 39
 - numEnum or numenum (double literal), 39 (double 字面值)
 - numF or numf (float literal) (float 字面值), 39
 - numL or numl (long literal) (long 字面值), 39
 - numU or numu (unsigned literal) (unsigned 字面值), 39
 - class member: *constant expression* (类成员: 常量表达式), 见 bit-field
- ## A
- abnormal termination, stream buffers (异常终止, 流缓冲区), 292
 - abort, 692, 748

- absInt, 530
- abstract base class (抽象基类), 596, 621
 - example (例子), 609
- abstract data type (抽象数据类型), 78, 429, 473
- abstraction, data (抽象, 数据), 432, 474
- access control (访问控制), 65
 - in base and derived classes (在基类和派生类中), 570
 - local class (局部类), 796
 - nested class (嵌套类), 787
 - using declarations to adjust (using 声明来调整), 573
- access label (访问标号), 65, 73, 432, 473
 - private, 65, 432
 - protected, 562, 622
 - public, 65, 432
- Account, 468
- accumulate, 396, 823
- Action, 783
- adaptor (适配器), 348, 353
 - container (容器), 348
 - function (函数), 533, 535, 553
 - iterator (迭代器), 399
- addition (+) (加), 150
 - string, 86
 - iterator (迭代器), 101, 313
 - pointer (指针), 123
 - Sales_item, 517
- address (地址), 35, 73
- address-of (&) (取地址), 115, 146
 - overloaded operator (重载操作符), 511
- adjacent_difference, 824
- adjacent_find, 813
- algorithm (算法), 392, 424
 - _copy versions (_copy 版本), 400, 421
 - _if versions (_if 版本), 421
 - element type constraints (元素类型约束), 394
 - independent of container (独立于容器), 393
 - iterator argument constraints (迭代器实参约束), 397, 415
 - iterator category and (迭代器种类和), 416, 418
 - naming convention (命名规范), 420-421
 - overloaded versions (重载版本), 420
 - parameter pattern (形参模式), 419-420
 - passing comparison function (传递比较函数), 403
 - read-only (只读), 396
 - structure (结构), 419
 - that reorders elements (对元素进行排序的算法), 421
 - that writes elements (写元素的算法), 398
 - type independence (类型独立性), 394, 396
 - using function object as argument (用函数对象作为实参), 531
 - with two input ranges (带两个输入范围), 420
- algorithm header (algorithm 头文件), 395
- alias, namespace (别名, 命名空间), 720, 750
- allocator, 755, 755-759, 805
 - allocate, 759
 - compared to operator new (与 operator new 比较), 761
 - construct, 755, 758
 - compared to placement new (与定位 new 表达式比较), 762
 - deallocate, 759
 - compared to operator delete (与 operator delete 比较), 761
 - destroy, 755, 759
 - compared to calling destructor (与调用析构函数比较), 763
 - operations (操作), 756
- alternative operator name (可选择的操作符名字), 46
- ambiguous (二义性)
 - conversion (转换), 541-544
 - multiple inheritance (多重继承), 734
 - function call (函数调用), 269, 272, 280
 - multiple base classes (多个基类), 738
 - overloaded operator (重载操作符), 550
- AndQuery, 609
 - definition (定义), 618
 - eval function (eval 函数), 619
- anonymous union (匿名联合), 795, 805
- app (file mode) (文件模式), 296
- append, string, 342
- argc, 244
- argument (实参), 25, 28, 226, 227, 280
 - array type (数组类型), 238
 - C-style string (C 风格字符串), 242
 - const reference type (const 引用类型), 235
 - conversion (转换), 229
 - with class type conversion (用类类型转换), 541
 - copied (复制的), 230
 - uses copy constructor (使用复制构造函数), 478
 - default (默认), 253
 - iterator (迭代器), 238, 242
 - multi-dimensioned array (多维数组), 241
 - passing (传递), 229
 - pointer to const (指向 const 对象的指针), 231
 - pointer to nonconst (指向非 const 对象的指针), 231
 - reference parameter (引用形参数), 233
 - template (模板), 见 template argument
 - to main (main 的), 243
 - to member function (成员函数的), 260
 - nonconst reference parameter (非 const 引用参数), 235
 - type checking (类型检查), 229
 - ellipsis (省略符), 244
 - of array type (数组类型的), 239
 - of reference to array (数组引用的), 240

- with class type conversion (用类类型转换的), 541
- argument deduction, template (模板实参推断), 637
- argument list (实参表), 226
- argv, 244
- arithmetic (算术)
 - iterator (迭代器), 100, 107, 312, 313
 - pointer (指针), 123, 146
- arithmetic operator (算术操作符)
 - and compound assignment (和复合赋值), 162
 - function object (函数对象), 533
 - overloaded operator (重载操作符), 517
- arithmetic type (算术类型), 34, 73
 - conversion (转换), 180, 188
 - from bool (从 bool 转换), 182
 - signed to unsigned (signed 到 unsigned), 36
 - conversion to bool (转换到 bool), 182
- array (数组), 40, 73, 110-114
 - and pointer (和指针), 122
 - argument (实参), 238
 - as initializer of vector (作为 vector 的初始化式), 140
 - assignment (赋值), 112
 - associative (关联数组), 388
 - conversion to pointer (转换到指针), 122, 181
 - and template argument (和模板实参), 639
 - copy (复制), 112
 - default initialization (默认初始化), 111
 - uses copy constructor (使用复制构造函数), 478
 - uses default constructor (使用默认构造函数), 460
 - definition (定义), 110
 - elements and destructor (元素与析构函数), 485
 - function returning (函数返回), 228
 - initialization (初始化), 111
 - multi-dimensioned (多维数组), 141-144
 - and pointer (和指针), 143
 - definition (定义), 142
 - initialization (初始化), 142
 - parameter (形参), 241
 - subscript operator (下标操作符), 142
 - of char initialization [char (数组) 的初始化], 112
 - parameter (形参), 238-244
 - buffer overflow (缓冲区溢出), 242
 - convention (规范), 241-243
 - reference type (引用类型), 240
 - size calculation (大小计算), 309
 - and sizeof (和 sizeof), 167
 - subscript operator (下标操作符), 113
- arrow operator (->) (箭头操作符), 164
 - auto_ptr, 704
 - class member access (类成员访问), 445
 - generic handle (泛型句柄), 670
 - overloaded operator (重载操作符), 525
- assert preprocessor macro (assert 预处理宏), 221, 223
- assign
 - container (容器), 328
 - string, 340
- assignment (赋值),
 - vs. initialization (与初始化), 49
 - memberwise (逐个成员), 483, 503
- assignment (=) (赋值), 13, 30, 159, 502
 - and conversion (和转换), 179
 - and copy constructor (和复制构造函数), 484
 - check for self-assignment (检查自身赋值), 490
 - container (容器), 328
 - for derived class (派生类的), 586
 - Message, 490
 - multiple inheritance (多重继承), 737
 - overloaded operator (重载操作符), 476, 483, 520
 - reference return (引用返回), 483, 521
 - pointer (指针), 120
 - rule of three (三法则), 485
 - exception for virtual destructors (虚析构函数的异常), 588
 - string, 86
 - synthesized (合成赋值), 483, 503
 - to base from derived [从派生类到基类 (赋值)], 578
 - to signed [给 signed (赋值)], 37
 - to unsigned [给 unsigned (赋值)], 37
 - use counting (使用计数), 495, 498
 - usually not virtual (通常不为虚), 588
 - valuelike classes (值型类), 501
 - yields lvalue (生成左值), 160
- associative array (关联数组), 见 map, 388
- associative container (关联容器), 356, 388
 - assignment (=) (赋值), 328
 - begin, 369
 - clear, 359
 - constructors (构造函数), 360
 - count, 377
 - element type constraints (元素类型约束), 309, 323
 - empty, 359
 - equal_range, 379
 - erase, 359
 - find, 377
 - insert, 364
 - key type constraints (键类型约束), 360
 - lower_bound, 377
 - operations (操作), 358
 - overriding the default comparison (覆盖默认比较), 604
 - rbegin, 412
 - rend, 412
 - returning an (返回一个关联容器), 381

reverse_iterator, 412
 size, 359
 supports relational operators (支持关系操作符), 359
 swap, 329
 types defined by (关联容器定义的类型), 361
 upper_bound, 377

associativity (结合性), 149, 170, 188
 overloaded operator (重载操作符), 507

at
 deque, 325
 vector, 325

ate (file mode) (ate 文件模式), 296

auto_ptr, 702, 748
 constructor (构造函数), 703
 copy and assignment (复制与赋值), 704
 default constructor (默认构造函数), 705
 get member (get 成员), 705
 operations (操作), 703
 pitfalls (缺陷), 707

reset member (reset 成员), 706
 self-assignment (自身赋值), 705
 automatic object (自动对象), 255, 280
 另见 local variable
 另见 parameter
 and destructor (和析构函数), 485

B

back
 queue, 352
 sequential container (顺序容器), 324

back_inserter, 399, 406, 424

bad, 289

bad_alloc, 175, 219

bad_cast, 219, 774

bad_typeid, 776

badbit, 289

base, 414

base class (基类), 285, 302, 558, 621

另见 virtual function

abstract (抽象), 596, 621

example (例子), 609

access control (访问控制), 561, 571

assignment operator, usually not virtual (赋值操作符, 通常不为虚), 588

can be a derived class (可以是派生类), 566

constructor (构造函数), 581

calls virtual function (调用虚函数), 589

not virtual (不为虚), 588

conversion from derived (从派生类转换), 567

access control (访问控制), 579

definition (定义), 560

destructor (析构函数)

calls virtual function (调用虚函数), 589

usually virtual (通常不为虚), 587

friendship not inherited (友元关系不能继承), 575

handle class (句柄类), 599

member operator delete (成员 operator delete), 764

member hidden by derived [被派生(类)隐藏的成员], 593

multiple (多个基类), 见 multiple base class

must be complete type (必须是完全类型), 566

no conversion to derived (不能转化为派生类), 580

object initialized or assigned from derived (用派生对象初始化或赋值的对象), 578

scope (作用域), 590

static members (static 成员), 576

user (用户), 563

virtual (虚), 751

见 virtual base class

Basket, 605

total function (total 函数), 606

Bear, 731

as virtual base [作为虚基(类)], 741

begin, 353

map, 369

set, 372

vector, 97

container (容器), 317

best match (最佳匹配), 269, 280

另见 function matching

bidirectional iterator (双向迭代器), 417, 424

list, 417

map, 417

set, 417

binary (file mode) (binary 文件模式), 296

binary function object (二元函数对象), 533

binary operator (二元操作符), 148, 188

binary_search, 814

BinaryQuery, 609

definition (定义), 617

bind1st, 535

bind2nd, 535

binder (绑定器), 535, 552

binding, dynamic (动态绑定), 559, 621

requirements for (要求), 566

bit-field (位域), 798, 805

access to (访问), 798

bitset, 101, 101-106, 107

any, 104

count, 104

flip, 105

- compared to bitwise NOT (与位求反比较), 155
 - none, 104
 - reset, 105
 - set, 105
 - size, 104
 - test, 105
 - to_ulong, 105
- compared to bitwise operator (与位操作符比较), 156
- constructor (构造函数), 101-103
 - header (头文件), 101
 - output operator (输出操作符), 106
 - subscript operator (下标操作符), 105
- bitwise AND (&) (位与), 156, 189
 - example (例子), 610
- bitwise exclusive or (^) (位异或), 156, 190
- bitwise NOT (~) (位求反), 155, 189
 - example (例子), 610
- bitwise operator (位操作符), 154-159
 - and compound assignment (和复合赋值), 162
 - compared to bitset (与 bitset 比较), 156
 - compound assignment (复合赋值), 157
 - example (例子), 290
 - operand (操作数), 155
- bitwise OR (|) (位或), 156, 190
 - example (例子), 290, 610
- block (块), 3, 13, 28, 193, 223
 - try, 216, 217, 224, 750
 - as target of if (作为 if 的目标), 196
 - function (函数), 227
- block scope (块作用域), 193
- body, function (体, 函数), 3, 29, 226, 281
- book finding program (书籍查找程序)
 - using equal_range (使用 equal_range), 379
 - using find (使用 find), 377
 - using upper_bound (使用 upper_bound), 378
- bookstore program (书店程序), 26
 - exception classes (异常类), 698
- bool, 35
 - and equality operator (和相等操作符), 154
 - conversion to arithmetic type (转换为算术类型), 182
 - literal (字面值), 39
- boolalpha manipulator (boolalpha 操纵符), 826
- brace, curly (花括号), 3, 28
- break statement (break 语句), 212, 223
 - and switch (和 switch), 201-203
- buffer (缓冲区), 8, 28
 - flushing (刷新), 290
- buffer overflow (缓冲区溢出), 114
 - and C-style string (和 C 风格字符串), 132
 - array parameter (数组形参), 242

- built-in type (内置类型), 3, 28, 34-37
 - class member default initialization (类成员默认初始化), 264
 - conversion (转换), 179
 - initialization of (初始化), 51
- Bulk_item,
 - class definition (类定义), 564
 - constructor (构造函数), 581
 - constructor using default arguments (使用默认实参的构造函数), 582
 - derived from Disc_item (从 Disc_item 派生), 584
 - interface (接口), 558
 - member functions (成员函数), 559
- byte (字节), 35, 73

C

- C++,
 - calling C function from C++ (在 C++ 中调用 C 函数), 801
 - compiling C and C++ (编译 C 和 C++), 803
 - using C linkage (使用 C 链接), 802
- .C file (.C 文件), 4
- .cc file (.cc 文件), 4
- .cpp file (.cpp 文件), 4
- .cp file (.cp 文件), 4
- C library header (C 库头文件), 90
- C with classes (带类的 C), 430
- C-style cast (C 风格强制类型转换), 186
- C-style string (C 风格字符串), 112, 130, 130-134, 145
 - and char* (和 char*), 131
 - and string literal (和字符串字面值), 140
 - compared to string (与 string 比较), 134, 138
 - definition (定义), 130
 - dynamically allocated (动态分配), 138
 - initialization (初始化), 130
 - parameter (形参), 242
 - pitfalls with generic programs (泛型程序中的缺陷), 671
- c_str, 140
 - example (例子), 294
- CachedObj
 - add_to_freelist, 771
 - operator delete, 770
 - operator new, 769
 - allocation explained (分配说明), 769
 - definition (定义), 767
 - definition of static members (static 成员定义), 771
 - design (设计), 766
 - illustration (举例), 767
 - inheriting from (由~继承), 768
- call operator (()) (调用操作符), 25, 30, 226, 280
 - execution flow (执行流), 226
 - overloaded operator (重载操作符), 530

- calling C function from C++ (在 C++ 中调用 C 函数), 801
- candidate function (候选函数), 270, 280
 - and function templates (和函数模板), 679
 - namespaces (命名空间), 728
 - overloaded operator (重载操作符), 549
- capacity
 - string, 336
 - vector, 331
- case label (case 标号), 201, 201-204, 223
 - default, 203
- cassert header (cassert 头文件), 221
- cast (强制类型转换), 183, 188
 - checked (带检查), 见 dynamic_cast
 - old-style (旧式的), 186
- catch clause (catch 子句), 216, 217, 223, 693, 749
 - catch(...), 696, 748
 - example (例子), 217
 - exception specifier (异常说明符), 693
 - matching (匹配), 693
 - ordering of (顺序), 694
 - parameter (形参), 693
- category, iterator (迭代器种类), 425
- cctype, 88-89, 107
 - header (头文件), 88
- cerr, 6, 28
- char, 34
 - literal (字面值), 40
- char string literal (char 字符串字面值), 见 string literal
- character (字符)
 - newline (\n) (换行符), 40
 - nonprintable (非打印字符), 40, 75
 - null (\0) (空字符), 40
 - printable (可打印的字符), 88
 - tab (\t) (制表符), 40
- checked cast (带检查的强制类型转换), 见 dynamic_cast
- CheckedPtr, 526
- children's story program (儿童故事程序), 400
 - revisited (再谈), 531
- cin, 6, 28
 - by default tied to cout (默认与 cout 绑在一起), 292
- cl, 4
- class (类), 20, 28, 63, 73, 473
 - static member (静态成员), 474
 - abstract base (抽象基类), 621
 - example (例子), 609
 - access labels (访问标号), 65, 432
 - as friend (作为友元), 465
 - base (基类), 285, 302, 见 base class, 621
 - concrete (具体类), 433
 - conversion (转换), 552
 - multiple conversions lead to ambiguities (多个转换导致二义性), 546
- conversion constructor (转换构造函数), 461
 - function matching (函数匹配), 547
 - with standard conversion (标准转换), 540
- data member (数据成员), 64, 74
 - const vs. mutable (const 与 mutable), 443
 - const, initialization (const 初始化), 455
 - constraints on type (类型约束), 438
 - definition (定义), 435
 - initialization (初始化), 454
 - mutable, 443
 - reference, initialization (引用, 初始化), 455
 - static, 469
- data member definition (数据成员定义), 65
 - default access label (默认访问标号), 433
 - default inheritance access label (默认继承访问标号), 574
 - definition (定义), 64, 430-440
 - and header (和头文件), 264, 437
 - derived (派生), 285, 302, 见 derived class, 621
- destructor definition (析构造函数定义), 486
- direct base (直接基数据成员定义), 见 immediate base class, 621
- explicit constructor (explicit 构造函数), 462
- forward declaration (前向声明), 438
- generic handle (泛型句柄), 667, 683
- handle (句柄), 见 handle class, 599, 622
- immediate base [直接基 (类)], 566, 622
- indirect base [间接基 (类)], 566, 622
- local (局部), 见 local class, 806
- member (成员), 64, 73, 430
- member access (成员访问), 445
- member function (成员函数), 25, 29, 见 member function
- member: *constant expression* (成员: 常量表达式), 见 bit-field
- multiple inheritance (多重继承, 多个基类), 见 multiple base class
- nested (嵌套的), 见 nested class, 806
- nonvirtual function, calls resolved at compile time (在编译时确定非虚函数调用), 569
- operator delete, 见 member operator
- operator new, 见 member operator new
- pointer member (指针成员)
 - copy control (复制控制), 492-501
 - copy control strategies (复制控制策略), 499
 - default copy behavior (默认复制行为), 493
- pointer to member (指向成员的指针), 780
 - definition (定义), 781
- pointer to member function, definition (指向成员函数的指针, 定义), 782
- preventing copies (禁止复制), 481

- private member (private 成员), 431
 - inheritance (继承), 561
- private member (私有成员), 75
- protected member (protected 成员), 562
- public member (public 成员), 75, 431
 - inheritance (继承), 561
- static member (static 成员), 467
 - as default argument (作为默认实参), 471
 - data member as constant expression (作为常量表达式的数
据成员), 471
 - example (例子), 468
 - inheritance (继承), 576
- template member (模板成员), 见 member template
- type member (类型成员), 435
- undefined member (未定义成员), 482
- user (用户), 433, 563
- virtual base (虚基类), 751
- virtual function, calls resolved at run time (在运行时确定的虚
函数调用), 569
- class, keyword (class, 关键字), 64
 - compared to typename (与 typename 比较), 631
 - in template parameter (在模板形参中), 630
 - in variable definition (在变量定义中), 440
- class keyword (class 关键字), 473
- class declaration (类声明), 438, 473
 - of derived class (派生类的), 566
- class derivation list (类派生列表), 563, 621
 - access control (访问控制), 570
 - default access label (默认访问标号), 574
 - multiple base classes (多个基类), 731
 - virtual base (虚基类), 742
- class scope (类作用域), 65, 444, 473
 - friend declaration (友元声明), 466
 - inheritance (继承), 590-595
 - member definition (成员定义), 445
 - name lookup (名字查找), 447
 - static members (static 成员), 470
 - virtual functions (虚函数), 594
- class template (类模板), 90, 107, 627, 683
 - 另见 template parameter
 - 另见 template argument
 - 另见 instantiation
 - compiler error detection (编译器错误检测), 634
 - declaration (声明), 629
 - definition (定义), 627
 - error detection (错误检测), 635
 - explicit template argument (显式模板实参), 636
 - export, 645
 - friend (友元)
 - declaration dependencies (声明依赖性), 658
 - explicit template instantiation (显示模板实例化), 657
 - nontemplate class or function (非模板类或函数), 656
 - template class or function (模板类或函数), 656
- member function (成员函数), 653
 - defined outside class body (在类定义体之外定义的), 651
 - instantiation (实例化), 653
- member specialization (成员特化), 677
- member template (成员模板), 见 member template
- nontype template parameter (非类型模板形参), 655
- static member (static 成员), 665
 - accessed through an instantiation (通过实例化访问), 666
 - definition (定义), 666
- type includes template argument(s) (包含模板实参的类型),
628, 637
- type-dependent code (类型相关的代码), 634
- uses of template parameter (模板形参的使用), 649
- class template specialization (类模板特化)
 - definition (定义), 675
 - member, declaration (成员, 声明), 677
 - member defined outside class body (类定义体外定义的成
员), 676
 - namespaces (命名空间), 730
- class type (类类型), 20, 28, 65
 - class member default initialization (类成员默认初始化), 264
 - conversion (转换), 183
 - initialization of (初始化), 52
 - object definition (对象定义), 439
 - parameter and overloaded operator (形参与重载操作符), 507
 - variable vs. function declaration (变量与函数声明), 460
- cleanup, object (清除, 对象), 见 destructor
- clear, 289, 290
 - associative container (关联容器), 359
 - example (例子), 290, 295
 - sequential container (顺序容器), 327
- clog, 6, 28
- close, 294
- comma operator (,) (逗号操作符) 168, 189
 - example (例子), 289
 - operand order of evaluation (操作数求值顺序), 172
 - overloaded operator (重载操作符), 511
- comment (注释), 10, 28
 - block (/* */) (块注释), 10, 28
 - single-line (//) (单行注释), 10, 28
- compare
 - plain function (普通函数), 624
 - string, 347
 - template version (模板版本), 625
 - instantiated with pointer (用指针实例化), 671
 - specialization (特化), 672
- compilation (编译)

- and header (和头文件), 67
- conditional (条件), 220
- inclusion model for templates (模板的包含模型), 644
- needed when class changes (类改变时需要), 434
- needed when inline function changes (内联函数改变时需要), 258
- separate (分别), 67, 76
 - of templates (模板的), 643
- separate model for templates (模板的分别编译模型), 644
- compiler (编译器)
 - extension (扩展), 112
 - flag for inclusion compilation model (包含编译模型的标志), 645
 - GNU, 4
 - Microsoft (微软), 4
 - template errors diagnosed at link time (链接时模板错误诊断), 635
- compiler extension (编译器扩展), 145
- compiling C and C++ (编译 C 和 C++), 803
- composition vs. inheritance (组合与继承), 573
- compound assignment (e.g., +=) (复合赋值), 13, 30, 161
 - string, 86
- bitwise operator (位操作符), 157
- iterator (迭代器), 313
- overloaded operator (重载操作符), 511, 518
- Sales_item, 521
- compound expression (复合表达式), 168, 188
- compound statement (复合语句), 193, 223
- compound type (复合类型), 58, 73, 145
- compute, 542
 - overloaded version (重载版本), 545
- concatenation (连接)
 - Screen operations (screen 操作), 441
 - string, 86
 - string literal (字符串字面值), 41
- concrete class (具体类), 433
 - initialization (初始化), 464
- condition (条件), 13, 28
 - and conversion (和转换), 179
 - assignment in [赋值 (在条件中)], 161
 - in do while statement (do while 语句中的), 211
 - in for statement (for 语句中的), 15, 207
 - in if statement (if 语句中的), 18, 195
 - in logical operator (逻辑操作符中的), 152
 - in while statement (while 语句中的), 205
 - stream type as [流类型作为 (条件)], 19, 183, 288
 - string input operation as (string 输入操作作为条件), 82
- condition state (条件状态), 287, 302
- conditional compilation (条件编译), 220
- conditional operator (?:) (条件操作符), 165, 189
 - operand order of evaluation (操作数求值顺序), 172
- console window (控制窗口), 6
- const, 57
 - and dynamically allocated array (和动态分配数组), 136
 - conversion to (转换为), 182, 231
 - and template argument (和模板实参), 639
 - iterator vs. const_iterator (迭代器与 const 迭代器), 100
 - object scope (对象作用域), 57, 69
 - overloading and (重载与), 267, 275
 - parameter (形参), 231
 - pointer (指针), 128
 - reference (引用), 59
 - initialization (初始化), 60
- const object, constructor (const 对象, 构造函数), 453
- const data member (const 数据成员)
 - static data member (static 数据成员), 470
 - compared to mutable (与 mutable 比较), 443
 - initialization (初始化), 455
- const member function (const 成员函数), 261, 262, 280, 431, 473
 - overloaded (重载), 442
 - reference return (引用返回), 442
 - static, 469
- const pointer (const 指针向 const 的指针), 另见 pointer to const
 - conversion from nonconst (从非 const 转换), 182
- const reference (const 引用)
 - argument (实参), 235
 - conversion from nonconst (从非 const 转换), 182
 - parameter (形参), 235
 - overloading (重载), 275
 - return type (返回类型), 249
- const void*, 127, 145
- const_cast, 183, 184
- const_iterator, 99, 415
 - compared to const iterator (与 const 迭代器比较), 100
- container (容器), 316
- const_reference, 317
- const_reverse_iterator, 412
 - container (容器), 316
- constant expression (常量表达式), 62, 74
 - and header file (和头文件), 69
 - array index (数组下标), 110
 - bit-field (位域), 798
 - enumerator (枚举成员), 62
 - nontype template parameter (非类型模板形参), 633
 - static data member (static 数据成员), 471
- construction, order of (构造, 顺序), 456, 749

- derived objects (派生对象), 581, 582
- multiple base classes (多个基类), 732
- virtual base classes (虚基类), 746
- constructor (构造函数), 49, 74, 262, 281, 431
 - const objects (const 对象), 453
 - conversion (转换), 461, 474
 - function matching (函数匹配), 547
 - with standard conversion (标准转换), 540
 - copy (复制), 476-482, 502
 - base from derived (从派生类到基类), 578
 - multiple inheritance (多重继承), 737
 - default (默认), 52, 74, 262, 281, 458-461, 474
 - default argument in (默认实参), 458
 - derived class (派生类), 581
 - initializes immediate base class (初始化直接基类), 583
 - initializes virtual base (初始化虚基类), 744
 - execution flow (执行流), 454
 - explicit, 462, 474
 - copy-initialization (复制初始化), 477
 - for associative container (用于关联容器), 360
 - for sequential container (用于顺序容器), 307-309
 - function try block (函数 try 块), 696
 - in constructor initializer list (构造函数初始化列表中的), 457
 - inheritance (继承), 581
 - initializer (初始化式), 452
 - may not be virtual (不能为虚), 588
 - object creation (对象创建), 452
 - order of construction (构造顺序), 456
 - derived objects (派生对象), 581, 582
 - multiple base classes (多个基类), 732
 - virtual base classes (虚基类), 746
 - overloaded (重载), 452
 - pair, 357
 - resource allocation (资源分配), 700
 - synthesized copy (合成的复制), 479, 503
 - synthesized default (合成的默认), 264, 281, 459, 474
 - virtual inheritance (虚继承), 744
- constructor initializer list (构造函数初始化列表), 263, 281, 431, 453-458, 474
 - compared to assignment (与赋值比较), 454
 - derived classes (派生类), 582
 - function try block (函数 try 块), 696
 - initializers (初始化式), 457
 - multiple base classes (多个基类), 733
 - sometimes required (有时需要), 455
 - virtual base class (虚基类), 745
- container (容器), 90, 107, 306, 353
 - 另见 sequential container
 - 另见 associative container
 - and generic algorithms (和泛型算法), 393
 - as element type (作为元素类型), 311
 - assignment (=) (赋值), 328
 - associative (关联容器), 356, 388
 - begin, 317
 - clear, 327
 - const_iterator, 316
 - const_reference, 317
 - const_reverse_iterator, 316
 - element type constraints (元素类型约束), 309, 323
 - elements and destructor (元素与析构函数), 485
 - elements are copies (元素被复制), 318
 - empty, 323
 - end, 317
 - erase, 402
 - has bidirectional iterator (具有双向迭代器), 417
 - inheritance (继承), 597
 - insert, 319
 - iterator, 316
 - rbegin, 317, 412
 - reference, 317
 - rend, 317, 412
 - returning a (返回一个容器), 381
 - reverse_iterator, 316, 412
 - sequential (顺序容器), 306, 354
 - size, 323
 - size_type, 316
 - supports relational operators (支持关系操作符), 321
 - swap, 329
 - types defined by (容器定义的类型), 316
- continue statement (continue 语句), 214, 223
 - example (例子), 290
- control, flow of (控制流), 192, 224
- conversion (转换), 178, 188
 - ambiguous (二义的), 541-544
 - and assignment (和赋值), 159
 - argument (实参), 229
 - with class type conversion (类类型转换), 541
 - arithmetic type (算术类型), 180, 188
 - array to pointer (指针数组), 122, 238
 - and template argument (和模板实参), 639
- class type (类类型), 183, 535, 552
 - design considerations (设计考虑), 544
 - example (例子), 537
 - multiple conversions lead to ambiguities (多个转换导致二义性), 546
 - operator (操作符), 537, 537-540, 552
 - operator and function matching (操作符与函数匹配), 545
 - used implicitly (隐式使用), 538
 - with standard conversion (标准转换), 539
- constructor (构造函数), 461

- function matching (函数匹配), 547
 - with standard conversion (标准转换), 540
 - derived to base (从派生类到基类), 567, 580
 - access control (访问控制), 579
 - enumeration type to integer (枚举类型到整数), 182
 - from istream (从 istream), 183
 - function matching of template and nontemplate functions (模板与非模板函数的函数匹配), 681
 - function to pointer (函数到指针), 277
 - and template argument (和模板实参), 639
 - implicit (隐式的), 189
 - inheritance (继承), 577
 - integral promotion (整型提升), 180
 - multi-dimensioned array to pointer (多维数组到指针), 143
 - multiple inheritance (多重继承), 734
 - nontemplate type argument (非模板类型实参), 640
 - of return value (返回值的), 246
 - rank for function matching (函数匹配排序), 272
 - rank of class type conversions (类类型转换的分类), 545
 - signed type (signed 类型), 180
 - signed to unsigned (signed 到 unsigned), 36
 - template argument (模板实参), 638
 - to const pointer (到 const 指针), 127
 - to const (到 const), 182
 - and template argument (和模板实参), 639
 - parameter matching (形参匹配), 231
 - virtual base (虚基类), 743
 - conversion constructor (转换构造函数), 474
 - copy, 815
 - copy constructor (复制构造函数), 476, 476—482, 502
 - and assignment operator (和赋值操作符), 484
 - argument passing (实参传递), 478
 - base from derived (从派生类到基类), 578
 - for derived class (派生类的), 586
 - initialization (初始化), 478
 - Message, 489
 - parameter (形参), 480
 - pointer members (指针成员), 480
 - rule of three (三法则), 485
 - exception for virtual destructors (虚析构函数的例外), 588
 - synthesized (合成的), 479, 503
 - use counting (使用计数), 495, 497
 - valuelike classes (值型类), 500
 - copy control (复制控制), 476, 502
 - handle class (句柄类), 601
 - inheritance (继承), 584—590
 - message handling example (消息处理实例), 489
 - multiple inheritance (多重继承), 737
 - of pointer members (指针成员的), 499
 - copy-initialization (复制初始化), 48
 - using constructor (使用构造函数), 477
 - copy_backward, 816
 - count, use (计数, 使用), 495, 503
 - count, 812
 - book finding program (书籍查找程序), 377
 - map, 367
 - multimap, 377
 - multiset, 377
 - set, 372
 - count_if, 404, 812
 - with function object argument (带函数对象实参), 532
 - cout, 6, 28
 - by default tied to cin (默认与 cin 绑在一起), 292
 - cstdint header (cstdint 头文件), 104, 123
 - cstdlib header (cstdlib 头文件), 247
 - cstring header (cstring 头文件), 132
 - curly brace (花括号), 3, 28
- ## D
- dangling else (悬垂 else), 198, 223
 - dangling pointer (悬垂指针), 176, 188
 - returning pointer to local variable (返回局部变量指针), 249
 - synthesized copy control (合成的复制控制), 494
 - data abstraction (数据抽象), 432, 474
 - advantages (优点), 434
 - data hiding (数据隐藏), 434
 - data structure (数据结构), 20, 28
 - data type, abstract (抽象数据类型), 473
 - dec manipulator (dec 操纵符), 827
 - decimal literal (十进制字面值), 38
 - declaration (声明), 52, 74
 - class (类), 438, 473
 - class template member specialization (类模板成员特化), 677
 - dependencies and template friends (依赖性与模板友元), 658
 - derived class (派生类), 566
 - export, 645
 - forward (前向), 438, 474
 - function (函数), 251
 - exception specification (异常说明), 708
 - function template specialization (函数模板特化), 672, 673
 - member template (成员模板), 661
 - template (模板), 629
 - using, 78, 108, 720, 750
 - access control (访问控制), 573
 - class member access (类成员访问), 574
 - overloaded inherited functions (重载继承函数), 593
 - declaration statement (声明语句), 193, 224
 - decrement (--) (自减), 190
 - iterator (迭代器), 312
 - overloaded operator (重载操作符), 526

- prefix yields lvalue (前缀生成左值), 162
- reverse iterator (反向迭代器), 412
- deduction, template argument (模板实参推断), 637
- default argument (默认实参), 253
 - and header file (和头文件), 254
 - function matching (函数匹配), 270
 - in constructor (构造函数中的), 458
 - initializer (初始化式), 254
 - overloaded function (重载函数), 267
 - virtual functions (虚函数), 570
- default case label (default case 标号), 203, 224
- default constructor (默认构造函数), 52, 74, 262, 281, 458—461, 474
 - Sales_item, 263
 - string, 52, 81
- default argument (默认实参), 458
- synthesized (合成的), 264, 281, 459, 474
- used implicitly (隐式使用的), 459
- variable definition (变量定义), 460
- definition (定义), 52, 74
 - array (数组), 110
 - base class (基类), 560
 - C-style string (C 风格字符串), 130
 - class (类), 64, 430—440
 - class data member (类数据成员), 65, 435
 - class static member (类 static 成员), 469
 - class template (类模板), 627
 - static member (static 成员), 666
 - class template specialization (类模板特化), 675
 - member defined outside class body (在类定义体之外定义的成员), 676
 - class type object (类类型对象), 439
 - derived class (派生类), 563
 - destructor (析构函数), 486
 - dynamically allocated array (动态分配数组), 135
 - dynamically allocated object (动态分配对象), 174
 - function (函数), 3
 - inside an if condition (在 if 条件中), 196
 - inside a switch expression (在 switch 表达式中), 203
 - inside a while condition (在 while 条件中), 205
 - map, 360, 373
 - multi-dimensioned array (多维数组), 142
 - namespace (命名空间), 712
 - can be discontinuous (可以是不连续的), 714
 - member (成员), 716
 - of variable after case label (case 标号后变量的), 204
 - overloaded operator (重载操作符), 482
 - pair, 356
 - pointer (指针), 115
 - pointer to function (指向函数的指针), 276
 - static data member (static 数据成员), 470
 - variable (变量), 48
- delete, 145, 176, 188, 806
 - compared to operator delete (与 operator delete 比较), 760
 - const object (const 对象), 178
 - execution flow (执行流), 760
 - member operator (成员 operator), 806
 - member operator (成员 operator)
 - and inheritance (和继承), 764
 - interface (接口), 764
 - memory leak (内存泄漏), 177, 485
 - null pointer (空指针), 176
 - runs destructor (运行析构函数), 485
- delete [], 135
 - and dynamically allocated array (和动态分配数组), 137
- deque, 353
 - as element type (作为元素类型), 311
 - assign, 328
 - assignment (=) (赋值), 328
 - at, 325
 - back, 324
 - begin, 317
 - clear, 327
 - const_iterator, 316
 - const_reference, 317
 - const_reverse_iterator, 316
 - constructor from element count, uses copy constructor (使用元素计数的构造函数, 使用复制构造函数), 478
 - constructors (构造函数), 307—309
 - difference_type, 316
 - element type constraints (元素类型约束), 309, 323
 - empty, 323
 - end, 317
 - erase, 326
 - invalidates iterator (使迭代器失效), 326
 - front, 324
 - insert, 319
 - invalidates iterator (使迭代器失效), 320
 - iterator, 316
 - iterator supports arithmetic (迭代器支持算术), 312
 - performance characteristics (性能特征), 334
 - pop_back, 326
 - pop_front, 326
 - push_back, 318
 - invalidates iterator (使迭代器失效), 321
 - push_front, 318
 - invalidates iterator (使迭代器失效), 321
 - random-access iterator (随机访问迭代器), 417
 - rbegin, 317, 412

- reference, 317
- relational operators (关系操作符), 321
- rend, 317, 412
- resize, 323
- reverse_iterator, 316, 412
- size, 323
- size_type, 316
- subscript ([]) (下标), 325
- supports relational operators (支持关系操作符), 313
- swap, 329
- types defined by (定义的类型), 316
- value_type, 317
- dereference (*) (解引用), 98, 108, 146, 389
 - and increment (和自增), 163
 - auto_ptr, 704
 - iterator (迭代器), 98
 - on map iterator yields pair (用于 map 迭代器生成 pair), 362
 - overloaded operator (重载操作符), 524
 - pointer (指针), 119
 - yields lvalue (生成左值), 99, 120
- derivation list, class (派生列表, 类), 563, 621
 - access control (访问控制), 570
 - default access label (默认访问标记), 574
- derived class (派生类), 285, 302, 558, 621
 - 另见 virtual function
 - access control (访问控制), 561, 572
 - as base class (作为基类), 566
 - assigned or copied to base object (赋值或赋值到基类对象), 578
 - assignment (=), (赋值) 586
 - constructor (构造函数), 581
 - calls virtual function (调用虚函数), 589
 - for remote virtual base (远程虚基类的), 744
 - initializes immediate base class (初始化直接基类), 583
 - constructor initializer list (构造函数初始化列表), 582
 - conversion to base (到基类的转换), 567
 - access control (访问控制), 579
 - copy constructor (复制构造函数), 586
 - default derivation label (默认派生标号), 574
 - definition (定义), 563
 - destructor (析构函数), 587
 - calls virtual function (调用虚函数), 589
 - friendship not inherited (友元关系不能继承), 576
 - handle class (句柄类), 599
 - member operator delete (成员 operator delete), 764
 - member hides member in base (成员隐藏基类中的成员), 593
 - multiple base classes (多个基类), 731
 - no conversion from base (不能从基类转换), 580
 - scope (作用域), 590
 - scope (::) to access base class member (作用域操作符访问基类成员), 569
 - static members (static 成员), 576
 - using declaration (using 声明)
 - inherited functions (继承的函数), 593
 - member access (成员访问), 574
 - with remote virtual base (具有祖先虚基类的), 742
- derived object (派生对象)
 - contains base part (包含基类部分), 565
 - multiple base classes, contains base part for each (多个基类, 包含每个基类部分), 732
- design (设计)
 - CachedObj, 766
 - class member access control (类成员访问控制), 563
 - class type conversions (类类型转换), 544
 - consistent definitions of equality and relational operators (相等与关系操作符的一致定义), 520
 - is-a relationship (“是一种”关系), 573
 - Message class (Message 类), 486
 - namespace (命名空间), 714
 - of handle classes (句柄类的), 599
 - of header files (头文件的), 67
 - export, 646
 - inclusion compilation model (包含编译模型), 644
 - separate compilation model (分别编译模型), 645
 - optimizing new and delete (优化 new 和 delete), 764
 - using freelist (使用自由列表), 766
 - overloaded operator (重载操作符), 510—513
 - overview of use counting (使用计数概述), 495
 - Query classes (Query 类), 609—611
 - Queue, 647
 - resource allocation is initialization (资源分配即初始化), 700—701
 - Sales_item handle class (Sales_item 句柄类), 599
 - TextQuery class (TextQuery 类), 380
 - vector memory allocation strategy (vector 内存分配策略), 756
 - writing generic code (编写泛型代码), 634
 - pointer template argument (指针模板实参), 671
- destruction, order of (顺序析构), 749
 - derived objects (派生对象), 587
 - multiple base classes (多个基类), 733
 - virtual base classes (虚基类), 747
- destructor (析构函数), 476, 484, 502
 - called during exception handling (异常处理期间调用的), 691
 - container elements (容器元素), 485
 - definition (定义), 486
 - derived class (派生类), 587
 - explicit call to (显式调用), 762

implicitly called (隐式调用的), 484
 library classes (库类), 709
 Message, 491
 multiple inheritance (多重继承), 737
 order of destruction (析构顺序), 485
 derived objects (派生对象), 587
 multiple base classes (多个基类), 733
 virtual base classes (虚基类), 747
 resource deallocation (资源分配), 700
 rule of three (三法则), 485
 exception for virtual destructors (虚析构函数的例外), 588
 should not throw exception (不应抛出异常), 692
 synthesized (合成的), 485, 486
 use counting (使用计数), 495, 497
 valuelike classes (值型类), 500
 virtual, multiple inheritance (虚的, 多重继承), 736
 virtual in base class (基类中的虚析构造函数) 587
 development environment, integrated (开发环境, 集成的), 3
 difference_type, 101, 107, 316
 dimension (维), 110, 145
 direct base class (直接基类), immediate base class, 621
 direct-initialization (直接初始化), 48
 using constructor (使用构造函数), 477
 directive, using (指示, using), 721, 751
 pitfalls (缺陷), 724
 Disc_item, 583
 class definition (类定义), 583
 discriminant (判别式), 794, 806
 divides<T>, 534
 division (/) (除), 150
 do while statement (do while 语句), 210
 condition in (中的条件), 211
 domain_error, 219
 dot operator (.) (点操作符), 25, 30
 class member access (类成员访问), 445
 double, 37
 literal (*numEnum* or *numenum*) (字面值), 39
 long double, 37
 notation output format control (记数法输出格式控制), 830
 output format control (输出格式控制), 828
 duplicate word program (复制单词程序), 400-404
 revisited (再谈), 531
 dynamic binding (动态绑定), 559, 621
 in C++ (在 C++ 中), 569
 requirements for (要求), 566
 dynamic type (动态类型), 568, 622
 dynamic_cast, 183, 773, 806
 example (例子), 773
 throws bad_cast (抛出 bad_cast), 774
 to pointer (到指针), 773

to reference (到引用), 774
 dynamically allocated (动态分配的), 145
 const object (const 对象), 177
 array (数组), 134, 134-139
 delete, 137
 definition (定义), 135
 initialization (初始化), 136
 of const (const 的), 136
 C-style string (C 风格型字符串), 138
 memory and object construction (内存与对象构造), 754
 object (对象), 174
 auto_ptr, 702
 constructor (构造函数), 453
 destructor (析构函数), 485
 exception (异常), 700

E

edit-compile-debug (编辑-编译-调试), 16, 28
 errors at link time (链接时错误), 635
 else, 见 if statement
 dangling (悬垂的), 198, 223
 empty
 string 83, 107
 vector 93, 107
 associative container (关联容器), 359
 container (容器), 323
 priority_queue, 352
 queue, 352
 stack, 351
 encapsulation (封装), 432, 474
 advantages (优点), 434
 end, 353
 map, 369
 set, 372
 vector, 97
 container (容器), 317
 end-of-file (文件结束符), 19, 28, 835
 entering from keyboard (从键盘输入), 20
 Endangered, 731
 endl, 8
 manipulator flushes the buffer (操纵符刷新缓冲区), 291
 ends, manipulator flushes the buffer (操纵符刷新缓冲区), 291
 enum keyword (enum 关键字), 62
 enumeration (枚举), 62, 74
 conversion to integer (转换为整数), 182
 function matching (函数匹配), 274
 enumerator (枚举成员), 62, 74
 conversion to integer (转换到整数), 182
 environment, integrated development (环境, 集成开发), 3
 eof, 289

- eofbit, 289
 - equal, 814
 - equal member function (equal 成员函数), 778
 - equal_range, 814
 - associative container (关联容器), 379
 - book finding program (书籍查找程序), 379
 - equal_to<T>, 534
 - equality (==) (等于), 30, 154
 - string, 85
 - algorithm (算法), 421
 - and assignment (和赋值), 161
 - container (容器), 322
 - container adaptor (容器适配器), 350
 - iterator (迭代器), 98, 312
 - overloaded operator (重载操作符), 512, 518
 - consistent with equality (与等于一致), 520
 - string, 347
 - erase
 - associative container (关联容器), 359
 - container (容器), 402
 - invalidates iterator (使迭代器失效), 326
 - map, 368
 - multimap, 376
 - multiset, 376
 - sequential container (顺序容器), 326
 - set, 372
 - string, 340
 - error, standard (标准错误), 6
 - escape sequence (转义序列), 40, 74
 - hexadecimal (\Xnnn) (十六进制的), 40
 - octal (\nnn) (八进制的), 40
 - evaluation (求值)
 - order of (顺序), 149, 189
 - short-circuit (短路求值), 152
 - exception, raise (异常, 引发), 见 throw
 - exception
 - class (类), 216, 224
 - class hierarchy (类层次), 698
 - constructor (构造函数), 220
 - extending the hierarchy (扩展层次), 697
 - header (头文件), 219
 - what member (what 成员), 218, 697
 - exception handling (异常处理), 215-220, 749
 - 另见 throw
 - 另见 catch clause
 - and terminate (和 terminate), 219
 - compared to assert (与 assert 比较), 221
 - exception in destructor (析构函数中的异常), 692
 - finding a catch clause (查找 catch 子句), 693
 - function try block (函数 try 块), 696, 749
 - handler (处理代码), 见 catch clause
 - library class destructors (库类析构函数), 709
 - local objects destroyed (撤销的局部对象), 691
 - specifier (说明符), 217, 224, 693, 749
 - nonreference (非引用), 693
 - reference (引用), 694
 - types related by inheritance (通过继承相关的类型), 694
 - stack unwinding (栈展开), 691
 - uncaught exception (未捕获的异常), 692
 - unhandled exception (未处理的异常), 219
 - exception object (异常对象), 690, 749
 - array or function (数组或函数), 689
 - initializes catch parameter (初始化 catch 形参), 693
 - must be copyable (必须为可复制的), 689
 - pointer to local object (局部对象的指针), 690
 - rethrow (重新抛出), 695
 - exception safety (异常安全), 700, 749
 - exception specification (异常说明), 706, 749
 - throw(), 708
 - function pointers (函数指针), 711
 - unexpected, 708
 - violation (违反), 708
 - virtual functions (虚函数), 710
 - executable file (可执行文件), 4
 - EXIT_FAILURE, 247
 - EXIT_SUCCESS, 247
 - explicit constructor (explicit 构造函数), 462, 474
 - copy-initialization (复制初始化), 477
 - export, 645
 - and header design (和头文件设计), 646
 - keyword (关键字), 645, 683
 - exporting C++ to C (导出 C++ 到 C), 802
 - expression (表达式), 7, 28, 148, 189
 - and operand conversion (和操作数转换), 179
 - compound (复合), 168, 188
 - constant (常量), 62, 74
 - throw, 689, 750
 - expression statement (表达式语句), 192, 224
 - extended_compute, 542
 - extension, compiler (编译器扩展), 145
 - extern, 53
 - extern 'C', 见 linkage directive
 - extern const, 57
- ## F
- factorial program (factorial 程序), 250
 - fail, 289
 - failbit, 289
 - file (文件)
 - executable (可执行的), 4

- object (目标), 68
- source (源), 4, 29
- file mode (文件模式), 296, 302
 - combinations (组合), 298
 - example (例子), 299
- file static (文件 static), 719, 749
- fill, 816
- fill_n, 815
- find, 392, 812
 - book finding program (书籍查找程序), 377
 - map, 368
 - multimap, 377
 - multiset, 377
 - set, 372
 - string, 344
- find last word program (查找最后一个单词程序), 414
- find_first_of, 812
- find_first_not_of, string, 346
- find_end, 812
- find_first_of, 396, 812
 - string, 345
- find_if, 421, 812
- find_last_not_of, string, 346
- find_last_of, string, 346
- find_val program (final_val 程序), 234
- fixed manipulator (fixed 操纵符), 830
- float, 37
 - literal (*numF* or *numf*) (字面值), 39
- floating point (浮点)
 - notation output format control (记数法输出格式控制), 830
 - output format control (输出格式控制), 828
- floating point literal (浮点字面值), 见 double literal
- flow of control (控制流), 192, 224
- flush, manipulator flushes the buffer (flush, 操纵符刷新缓冲区), 291
- Folder, 见 Message
- for statement (for 语句), 29, 207
 - condition in (条件), 207
 - execution flow (执行流), 208
 - expression (表达式), 207
 - for header (for 语句头), 207
 - initialization statement (初始化语句), 207
 - scope (作用域), 15
- for statement (for 语句), 14
- for_each, 813
- format state (格式状态), 825
- forward declaration of class type (类类型的前向声明), 438
- forward iterator (前向迭代器), 417, 424
- fp_compute, 542
- free store (自由存储区), 135, 145
- freelist (自由列表), 766, 806
- friend (友元), 465, 474
 - class (类), 465
 - class template (类模板)
 - explicit template instantiation (显式模板实例化), 657
 - nontemplate class or function (非模板类或函数), 656
 - template class or function (模板类或函数), 656
 - function template, example (函数模板, 例子), 659
- inheritance (继承), 575
- member function (成员函数), 466
- overloaded function (重载函数), 467
- overloaded operator (重载操作符), 509
- scope considerations (作用域考虑), 466
- namespaces (命名空间), 727
- template example (模板例子), 658
- friend keyword (friend 关键字), 465
- front
 - queue, 352
 - sequential container (顺序容器), 324
- front_inserter, 406, 424
 - compared to inserter (与 inserter 比较), 406
- fstream, 285, 293-299, 302
 - 另见 istream
 - 另见 ostream
 - close, 294
 - constructor (构造函数), 293
 - file marker (文件标记), 838
 - file mode (文件模式), 296
 - combinations (组合), 298
 - example (例子), 299
 - file random access (文件随机访问), 838
 - header (头文件), 285, 293
 - off_type, 839
 - open, 293
 - pos_type, 839
 - random IO sample program (随机 IO 示例程序), 840
 - seek and tell members (seek 和 tell 成员), 838-842
- function (函数), 2, 29, 225, 281
 - equal member (equal 成员), 778
 - inline, 257, 281
 - candidate (候选), 270, 280
 - conversion to pointer (转换到指针), 277
 - and template argument (和模板实参), 639
 - function returning (函数返回), 228
 - inline and header (inline 与头文件), 257
 - member (成员), 25, 29, 见 member function, 474
- nonvirtual, calls resolved at compile time (在编译时确定的非虚调用), 569
- overloaded (重载), 265, 281
 - compared to redeclaration (与重复声明比较), 266

- friend declaration (友元声明), 467
- scope (作用域), 268
- virtual (虚), 593
- pure virtual (纯虚), 596, 622
 - example (例子), 609
- recursive (递归), 249, 281
- viable (可行的), 270, 282
- virtual (虚的), 559, 566-570, 622
 - assignment operator (赋值操作符), 588
 - calls resolved at run time (运行时确定的调用), 568
 - compared to run-time type identification (与运行时类型识别比较), 777
 - default argument (默认实参), 570
 - derived classes (派生类), 564
 - destructor (析构函数), 587
 - destructor and multiple inheritance (析构函数与多重继承), 736
 - exception specifications (异常说明), 710
 - in constructors (在构造函数中), 589
 - in destructor (在析构函数中), 589
 - introduction (介绍), 561
 - multiple inheritance (多重继承), 735
 - no virtual constructor (没有虚析构函数), 588
 - overloaded (重载), 593
 - overloaded operator (重载操作符), 615
 - overriding run-time binding (覆盖运行时绑定), 570
 - return type (返回类型), 564
 - run-time type identification (运行时类型识别), 772
 - scope (作用域), 594
 - to copy unknown type (复制未知类型), 602
 - type-sensitive equality (类型敏感相等), 778
- function adaptor (函数适配器), 533, 535, 553
 - bind1st, 535
 - bind2nd, 535
 - binder (绑定器), 535
 - negator (求反器), 535
 - not1, 535
 - not2, 535
- function body (函数体), 3, 29, 226, 281
- function call (函数调用)
 - ambiguous (二义性), 269, 272
 - execution flow (执行流), 226
 - overhead (开销), 257
 - through pointer to function (通过函数指针), 278
 - through pointer to member (通过成员指针), 784
 - to overloaded operator (到重载操作符), 509
 - to overloaded postfix operator (到重载前缀操作符), 529
 - using default argument (使用默认实参的), 253
- function declaration (函数声明), 251
 - and header file (和头文件), 252
- exception specification (异常说明), 708
- function definition (函数定义), 3
- function matching (函数匹配), 269, 281
 - and overloaded function templates (和重载函数模板), 679-682
 - examples (例子), 680
- argument conversion (实参转换), 269
- conversion operator (转换操作符), 545
- conversion rank (转换分类), 272
 - class type conversions (类类型转换), 545
- enumeration parameter (枚举形参), 274
- integral promotion (整型提升), 273
- multiple parameters (多个形参), 272
- namespaces (命名空间), 727
- of member functions (成员函数的), 436
- overloaded operator (重载操作符), 547-551
- function name (函数名), 3, 29
- function object (函数对象), 531, 553
 - algorithms (算法), 531
 - example (例子), 534
 - binary (二元的), 533
 - library defined (库定义的), 533
 - unary (一元的), 533
- function pointer (函数指针), 276-279
 - and template argument deduction (和模板实参推断), 640
 - definition (定义), 276
 - exception specifications (异常说明), 711
 - function returning (函数返回的), 228
 - initialization (初始化), 277
 - overloaded functions (重载函数), 279
 - parameter (形参), 278
 - return type (返回类型), 278
 - typedef, 276
- function prototype (函数原型), 251, 281
- function return type (函数返回类型), 226, 227, 281
 - const reference (const 引用), 249
 - no implicit return type (非隐式返回类型), 228
 - nonreference (非引用), 247
 - uses copy constructor (使用复制构造函数), 478
 - reference (引用), 248
 - reference yields lvalue (引用生成左值), 249
 - void, 245
- function scope (函数作用域), 227
- function table (函数表), 785
 - pointer to member (指向成员的指针), 785
- function template (函数模板), 625, 683
 - 另见 template parameter
 - 另见 template argument
 - 另见 instantiation
 - as friend (作为友元), 659

compiler error detection (编译器错误检测), 634
 declaration (声明), 629
 error detection (错误检测), 635
 explicit template argument (显式模板实参), 642
 and function pointer (和函数指针), 643
 specifying (指定), 642
 export, 645
 inline, 626
 instantiation (实例化), 637
 template argument deduction (模板实参推断), 637
 type-dependent code (类型相关代码), 634
 function template specialization (函数模板特化)
 compared to overloaded function (与重载函数比较), 673
 declaration (声明), 672, 673
 example (例子), 672
 namespaces (命名空间), 730
 scope (作用域), 674
 function try block (函数 try 块), 696, 749

G

g++, 4
 gcd program (gcd 程序), 226
 generate, 816
 generate_n, 815
 generic algorithm (泛型算法), 见 algorithm
 generic handle class (泛型句柄类), 667, 683
 generic memory management (泛型内存管理), 见 CachedObj
 generic programming (泛型编程), 95, 624
 and pointer template argument (和指针模板实参), 671
 type-independent code (独立于类型的代码), 634
 getline, 82, 107
 example (例子), 300, 386
 global namespace (全局命名空间), 716, 750
 global scope (全局作用域), 54, 74
 global variable, lifetime (全局变量, 生命期), 254
 GNU compiler (GNU 编译器), 4
 good, 289
 goto statement (goto 语句), 214, 224
 greater-than (>) (大于), 30, 153
 greater-than-or-equal (>=) (大于或等于), 30, 153
 greater<T>, 534
 greater_equal<T>, 534
 GT6 program (GT6 程序), 403
 GT_cls, 532
 guard header (保护头文件), 71, 74

H

.h file (.h 文件), 21
 Handle, 667
 int instantiation (int 实例化), 668

operations (操作), 668
 Sales_item instantiation (Sales_item 实例化), 669
 handle class (句柄类), 599, 622
 copy control (复制控制), 601
 copying unknown type (复制未知类型), 602
 design (设计), 599
 generic (泛型), 667, 683
 that hides inheritance hierarchy (隐藏继承层次), 610
 using a (使用一个句柄类), 603
 handler (处理代码), 见 catch clause
 has-a relationship (“有一个”关系), 573
 HasPtr
 as a smart pointer (作为一个智能指针), 495
 using synthesized copy control (使用合成复制控制), 493
 with value semantics (带值语义的), 499
 header (头文件), 7, 29, 67, 74
 bitset, 101
 cctype, 88, 107
 cstddef, 104
 iomanip, 829
 string, 80
 vector, 90
 algorithm, 395
 and constant expression (和常量表达式), 69
 and library names (和库名), 810
 C library (C 库), 90
 cassert, 221
 class definition (类定义), 264, 437
 cstddef, 123
 cstdlib, 247
 cstring, 132
 default argument (默认实参), 254
 deque, 307
 design (设计), 67
 export, 646
 inclusion compilation model (包含编译模型), 644
 namespace members (命名空间成员), 714
 separate compilation model (分别编译模型), 645
 exception, 219
 fstream, 285, 293
 function declaration (函数声明), 252
 inline function (inline 函数), 257
 inline member function definition (inline 成员函数定义), 437
 iostream, 285
 iterator, 399
 list, 307
 map, 360, 375
 new, 219
 numeric, 395

programmer-defined (程序员定义的), 67-72
 queue, 349
 Sales_item, 21, 67, 264
 set, 373, 375
 sstream, 285, 300
 stack, 349
 stdexcept, 217, 219
 type_info, 219
 using declaration (using 声明), 80
 utility, 356
 vector, 307
 header file, naming convention (头文件, 命名规范), 264
 header guard (头文件保护符), 71, 74
 heap (堆), 135, 145
 hex manipulator (hex 操纵符), 827
 hexadecimal, literal (0Xnum or 0xnum) (十六进制的, 字面值), 38
 hexadecimal escape sequence (\Xnnn) (十六进制的转义序列), 40
 hides, names in base hidden by names in derived (隐藏, 派生类中的名字隐藏基类中名字), 592
 hierarchy, inheritance (继承层次), 558, 566, 622
 high-order bits (高阶位), 102, 107

I

IDE, 3
 identification, run-time type (识别, 运行时类型), 772-780, 807
 identifier (标识符), 46, 74
 naming convention (命名规范), 47
 reserved (保留的), 47
 if statement, else branch (if 语句, else 分支), 224
 if statement (if 语句), 17, 29, 195, 224
 compared to switch (与 switch 比较), 199
 dangling else (悬垂 else), 198
 else branch (else 分支), 18, 197
 ifstream, 285, 293-299
 另见 istream
 close, 294
 constructor (构造函数), 293
 file marker (文件标记), 838
 file mode (文件模式), 296
 combinations (组合), 298
 example (例子), 299
 file random access (文件随机访问), 838
 off_type, 839
 open, 293
 pos_type, 839
 random IO sample program (随机 IO 示例程序), 840
 seek and tell members (seek 和 tell 成员), 838-842
 immediate base class (直接基类), 566, 622

implementation (实现), 63, 74, 432
 implementation inheritance (实现继承), 573
 implicit this pointer, overloaded operator (隐式 this 指针, 重载操作符), 508
 implicit conversion (隐式转换), 见 conversion, 189
 implicit return (隐式 return), 245
 from main allowed (允许从 main 的), 247
 implicit this pointer (隐式 this 指针), 260, 281, 431, 440
 in and overloaded operator (与重载操作符), 483
 static member functions (static 成员函数), 469
 in (file mode) (in 文件模式), 296
 include, 见 #include
 includes, 822
 inclusion compilation model (包含编译模型), 644, 683
 incomplete type (不完全类型), 438, 474
 restriction on use (使用限制), 438, 566, 693
 increment (++) (自增), 13, 30, 146, 190
 reverse iterator (反向迭代器), 412
 and dereference (和解引用), 163
 iterator (迭代器), 98, 108, 312
 overloaded operator (重载函数), 526
 pointer (指针), 114
 prefix yields lvalue (前缀生成左值), 162
 indentation (缩排), 14, 197
 index (下标), 87, 107
 indirect base class (间接基类), 566, 622
 inequality (!=) (不等于), 30, 154
 iterator (迭代器), 98
 container (容器), 322
 container adaptor (容器适配器), 350
 iterator (迭代器), 312
 overloaded operator (重载操作符), 512, 519
 string, 347
 inheritance (继承), 284, 302
 containers (容器), 597
 conversions (转换), 577
 default access label (默认访问标号), 574
 friends (友元), 575
 handle class (句柄类), 599
 implementation (实现), 573
 interface (接口), 573
 iostream diagram (iostream 图), 285
 multiple (多个), 见 multiple base class, 731
 private (私有), 622
 static members (static 成员), 576
 virtual (虚), 741, 751
 inheritance hierarchy (继承层次), 558, 566, 622
 inheritance vs. composition (继承与组成), 573
 initialization (初始化), 9
 vs. assignment (与赋值), 49

- array (数组), 111
- array of char (char 数组), 112
- built-in type (内置类型), 51
- C-style string (C 风格字符串), 130
- class data member (类数据成员), 454
- class member of built-in type (内置类型的类成员), 264
- class member of class type (类类型的类成员), 264
- class type (类类型), 52, 452
- const static data member (const static 数据成员), 470
- dynamically allocated array (动态分配数组), 136
- dynamically allocated object (动态分配对象), 174
- map, 373
- memberwise (逐个成员), 479, 503
- multi-dimensioned array (多维数组), 142
- objects of concrete class type (具体类类型的对象), 464
- pair, 356
- parameter (形参), 229
- pointer (指针), 117-119
- pointer to function (指向函数的指针), 277
- return value (返回值), 247
- value (值), 92, 108
- variable (变量), 48, 50, 76
- initialization vs. assignment (初始化与赋值), 456
- initialized (已初始化的), 48, 75
- initializer list, constructor (初始化列表, 构造函数), 263, 281, 431, 453-458, 474
- inline function (inline 函数), 257, 281
 - and header (和头文件), 257
 - function template (函数模版), 626
 - member function (成员函数), 437
 - and header (和头文件) 437
- inner_product, 823
- inplace_merge, 816
- input, standard (标准输入), 6
- input(>>) (输入), 8, 30
 - Sales_item, 516
 - istream_iterator, 408
 - string, 81, 108
 - overloaded operator (重载操作符), 515
 - error handling (错误处理), 516-517
 - must be nonmember (必须为非成员), 514
 - precedence and associativity (优先级和结合性), 158
- input iterator (输入迭代器), 416, 424
- insert
 - inserter, 406
 - invalidates iterator (使迭代器失效), 320
 - map, 364
 - multimap, 376
 - multiset, 376
 - return type from set::insert (set::insert 返回类型), 373
 - sequential container (顺序容器), 319
 - set, 373
 - string, 340
- insert iterator (插入迭代器), 399, 405, 425
- inserter (插入器), 406
- inserter, 425
 - compared to front_inserter (与 front_inserter 比较), 406
- instantiation (实例化), 625, 684
 - class template (类模版), 628, 636, 654
 - member function (成员函数), 653
 - nontype parameter (非类型形参), 655
 - type (类型), 637
 - error detection (错误检测), 635
 - function template (函数模版), 637
 - from function pointer (来自函数指针的), 640
 - nontemplate argument conversion (非模板实参转换), 640
 - nontype template parameter (非类型模版转换), 633
 - template argument conversion (模板实参转换), 638
 - member template (成员模版), 663
 - nested class template (嵌套类模版), 788, 791
 - on use, 636
 - static class member (static 类成员), 665
- int, 34
 - literal (字面值), 38
- Integral, 539
- integral promotion (整型提升), 180, 189
 - function matching (函数匹配), 273
- integral type (整型), 34, 75
- integrated development environment (集成开发环境), 3
- interface (接口), 63, 75, 432
- interface inheritance (接口继承), 573
- internal manipulator (internal 操纵符), 832
- interval, left-inclusive (左闭合区间), 314, 354
- invalid_argument, 219
- invalidated iterator (无效迭代器), 315, 353
- IO stream (IO 流), 见 stream
- iomanip header (iomanip 头文件), 829
- iostate, 289
- iostream, 6, 29, 285
 - 另见 istream
 - 另见 ostream
 - header (头文件), 285
 - inheritance hierarchy (继承层次), 740
 - seek and tell members (seek 和 tell 成员), 838
- is-a relationship (“是一个”关系), 573
- isalnum, 88
- isalpha, 88

ISBN, 2

isbn_mismatch, 699

 destructor explained (析构函数解释的), 709

isctrl, 88

isdigit, 88

isgraph, 88

islower, 88

isprint, 88

ispunct, 88

isShorter program (isshorter 程序), 235, 403

isspace, 88

istream, 6, 29, 285

 另见 manipulator

 condition state (条件状态), 287

 flushing input buffer (刷新输入缓冲区), 290

 format state (格式化状态), 825

 gcount, 837

 get, 834

 multi-byte version (多字节版本), 836

 returns int (返回 int), 835, 836

 getline, 82, 836

 getline, example (getline, 例子), 300

 ignore, 837

 inheritance hierarchy (继承层次), 740

 input (>>) (输入), 8

 precedence and associativity (优先级和结合性), 158

 no containers of (没有容器的), 310

 no copy or assign (没有复制或赋值), 287

 peek, 834

 put, 834

 putback, 835

 read, 837

 seek and tell members (seek 和 tell 成员), 838

 unformatted operation (未格式化的操作), 834

 multi-byte (多字节), 837

 single-byte (单字节), 834

 unget, 835

 write, 837

istream_iterator, 407, 425

 and class type (和类类型), 410

 constructors (构造函数), 408

 input iterator (输入迭代器), 417

 input operator (>>) (输入操作符), 408

 limitations (限制), 411

 operations (操作), 409

 used with algorithms (与算法一起使用的), 411

istringstream, 285, 299-301

 另见 istream

 str, 301

 word per line processing (每行处理的单词), 300, 370, 386

isupper, 88

isxdigit, 88

Item_base

 class definition (类定义), 560

 constructor (构造函数), 580

 interface (接口), 558

 member functions (成员函数), 559

iter_swap, 816

iterator (迭代器), 95, 95-101, 107, 311-316, 354

 argument (实参), 238

 arrow (->) (箭头), 164

 bidirectional (双向), 417, 424

 compared to reverse iterator (与反向迭代器比较), 413, 414

 destination (目的), 399, 419

 equality (相等), 98, 312

 forward (前向的), 417, 424

 generic algorithms (泛型算法), 394

 inequality (不相等), 98, 312

 input (输入), 416, 424

 insert (插入), 399, 405, 425

 invalidated (无效的), 315, 353

 invalidated by (因……失效)

 assign, 328

 erase, 326

 insert, 321

 push_back, 321

 push_front, 321

 resize, 324

 off-the-end (超出末端的), 97, 394, 425

 operations (操作), 311

 output (输出), 416, 425

 parameter (形参), 238, 242

 random-access (随机访问), 417, 425

 relational operators (关系操作符), 313

 reverse (反向), 405, 412-415, 425

 stream (流), 425

iterator (迭代器), 362, 374

 container (容器), 316

iterator header (iterator 头文件), 399

iterator arithmetic (迭代器算术操作), 100, 107, 312, 313

 relational operators (关系操作符), 313

iterator category (迭代器种类), 416-418, 425

 algorithm and (算法和), 416, 418

 bidirectional iterator (双向迭代器), 417

 forward iterator (前向迭代器), 417

 hierarchy (层次), 417

 input iterator (输入迭代器), 416

 output iterator (输出迭代器), 416

 random-access iterator (随机访问迭代器), 417

iterator range (迭代器范围), 314, 314-316, 354

algorithms constraints on (的算法约束), 397, 415
 erase, 327
 generic algorithms (泛型算法), 394
 insert, 320

K

key_type, 388
 associative containers (关联容器), 362
 keyword (关键字)
 enum, 62
 friend, 465
 namespace, 712
 protected, 562
 template, 625
 try, 217
 union, 793
 virtual, 559
 export, 645
 keyword table (关键字表), 46
 Koenig lookup (Koenig 查找), 726

L

label (标号)
 case, 201, 201-204, 223
 access (访问), 65, 73, 432, 473
 statement (语句), 214
 labeled statement (带标号的语句), 214, 224
 left manipulator (left 操纵符), 832
 left-inclusive interval (左闭合区间), 314, 354
 left-shift (<<) (左移), 155, 190
 length_error, 219
 less-than (<) (小于), 30, 153
 overloaded and containers (重载和容器), 520
 used by algorithm (算法所用的), 420
 less-than-or-equal (<=) (小于或等于), 13, 30, 153
 less<T>, 534
 less_equal<T>, 534
 lexicographical_compare, 823
 library, standard (标准库), 5, 29
 library names to header table (头文件表的库名字), 810
 library type (库类型), 29
 lifetime, object (对象生命期), 254, 281
 link time errors from template (来自模板的链接时错误), 635
 linkage directive (链接指示), 801, 806
 C++ to C (C++到C), 802
 compound (复合), 802
 overloaded function (重载函数), 803
 parameter or return type (形参或返回类型), 804
 pointer to function (指向函数的指针), 803
 single (单个), 801

linking (链接), 68, 75
 list, 354
 as element type (作为元素类型), 311
 assign, 328
 assignment (=) (赋值), 328
 back, 324
 begin, 317
 bidirectional iterator (双向迭代器), 417
 clear, 327
 const_iterator, 316
 const_reference, 317
 const_reverse_iterator, 316
 constructor from element count, uses copy constructor (来自元素计数的构造函数, 使用复制构造函数), 478
 constructors (构造函数), 307-309
 element type constraints (元素类型约束), 309, 323
 empty, 323
 end, 317
 erase, 326
 front, 324
 insert, 319
 iterator, 316
 merge, 423
 performance characteristics (性能特征), 334
 pop_back, 326
 pop_front, 326
 push_back, 318
 push_front, 318
 rbegin, 317, 412
 reference, 317
 relational operators (关系操作符), 321
 remove, 423
 remove_if, 423
 rend, 317, 412
 resize, 323
 reverse, 423
 reverse_iterator, 316, 412
 size, 323
 size_type, 316
 specific algorithms (特有的算法), 421
 splice, 423
 swap, 329
 types defined by (定义的类型), 316
 unique, 423
 value_type, 317
 literal (字面值), 37, 37-42, 75
 bool, 39
 char, 40
 decimal (十进制的), 38
 double (numEnum or numenum), 39

float (*numF* or *numf*), 39
 hexadecimal (*0Xnum* or *0xnum*) (十六进制的), 38
 int, 38
 long (*numL* or *numl*), 38
 long double (*ddd.dddL* or *ddd.ddd1*), 39
 multi-line (多行), 42
 octal (*0num*) (八进制的), 38
 string (字符串), 9, 30, 40
 unsigned (*numU* or *numu*), 39
 wchar_t, 40
 local class (局部类), 796, 806
 access control (访问控制), 796
 name lookup (名字查找), 797
 nested class in (其中的嵌套类), 797
 restrictions on (限制), 796
 local scope (局部作用域), 54, 75
 local static object (局部 static 对象), 255, 281
 local variable (局部变量), 227, 281
 destructor (析构函数), 485
 lifetime (生命期), 254
 reference return type (引用返回类型), 248
 logic_error, 219
 logical AND (&&) (逻辑与), 152
 operand order of evaluation (操作数求值顺序), 172
 overloaded operator (重载操作符), 511
 logical NOT (!) (逻辑非), 153
 logical operator (逻辑操作符), 152
 function object (函数对象), 533
 logical OR (||) (逻辑或), 152
 operand order of evaluation (操作数求值顺序), 172
 overloaded operator (重载操作符), 511
 logical_and<T>, 534
 logical_not<T>, 534
 logical_or<T>, 534
 long, 34
 literal (*numL* or *numl*) (字面值), 38
 long double, 37
 long double, literal (*ddd.dddL* or *ddd.ddd1*) (long double, 字面值), 39
 lookup, name (名字查找), 447, 474
 and templates (和模板), 647
 before type checking (在类型检查之前), 269, 593
 multiple inheritance (多重继承), 738
 class member declaration (类成员声明), 447
 class member definition (类成员定义), 448, 450
 class member definition, examples (类成员定义, 例子), 449
 collisions under inheritance (继承下的冲突), 591
 depends on static type (依赖静态类型), 590
 multiple inheritance (多重继承), 735
 inheritance (继承), 590, 595

local class (局部类), 797
 multiple inheritance (多重继承), 737
 ambiguous names (有二义性的名字), 738
 namespace names (命名空间名字), 724
 argument-dependent lookup (实参相关的查找), 726
 nested class (嵌套类), 791
 overloaded virtual functions (重载虚函数), 593
 virtual inheritance (虚继承), 743
 low-order bits (低阶位), 102, 107
 lower_bound, 814
 associative container (关联容器), 377
 book finding program (书籍查找程序), 378
 lvalue (左值), 45, 75
 assignment (赋值), 160
 dereference (解引用), 99
 function reference return type (函数引用返回类型), 249
 prefix decrement (前自减), 162
 prefix increment (前自增), 162
 subscript (下标), 88

M

machine-dependent (机器相关的)
 bitfield layout (位域布局), 798
 char representation (char 表示), 36
 division and modulus result (除和求模结果), 151
 end-of-file character (文件结束符), 20
 iostate type (iostate 类型), 288
 linkage directive language (链接指示语言), 802
 nonzero return from main (从 main 返回的非零值), 247
 pre-compiled headers (预编译的头文件), 67
 random file access (随机文件访问), 837
 reinterpret_cast, 185
 representation of enum type (enum 类型的表示), 274
 return from exception what operation (从 exception what 操作返回), 220
 signed and out-of-range value (signed 和超出范围的值), 37
 signed types and bitwise operators (signed 类型和位操作符), 155
 size of arithmetic types (算术类型的长度), 34
 template compilation optimization (模板编译优化), 645
 terminate function (terminate 函数), 219
 type_info members (type_info 成员), 779
 vector memory allocation size (vector 内存分配大小), 331
 volatile implementation (volatile 实现), 799
 magic number (魔数), 56, 75
 main, 2, 29
 arguments to (实参), 243
 not recursive (非递归的), 251

- return type (返回类型), 3
- return value (返回值), 2-5, 247
- returns 0 by default (默认返回 0), 247
- make_pair, 358
- make_plural program (make_plural 程序), 248
- manip, 542
- manipulator (操纵符), 8, 29, 825
 - boolalpha, 826
 - dec, 827
 - fixed, 830
 - hex, 827
 - internal, 832
 - left, 832
 - noboolalpha, 827
 - noshowbase, 828
 - noshowpoint, 832
 - noskipws, 833
 - nouppercase, 828
 - oct, 827
 - right, 832
 - scientific, 830
 - setfill, 832
 - setprecision, 829
 - setw, 832
 - showbase, 827
 - showpoint, 831
 - skipws, 833
 - uppercase, 828
- boolalpha, 826
- change format state (改变格式状态), 825
- dec, 827
- endl flushes the buffer (endl 刷新缓冲区), 291
- ends flushes the buffer (ends 刷新缓冲区), 291
- fixed, 830
- flush flushes the buffer (flush 刷新缓冲区), 291
- hex, 827
- internal, 832
- left, 832
- noboolalpha, 827
- noshowbase, 828
- noshowpoint, 832
- noskipws, 833
- nouppercase, 828
- oct, 827
- right, 832
- scientific, 830
- setfill, 832
- setprecision, 829
- setw, 832
- showbase, 827
- showpoint, 831
- skipws, 833
- uppercase, 828
- unitbuf flushes the buffer (unitbuf 刷新缓冲区), 291
- uppercase, 828
- map, 356, 388
 - as element type (作为元素类型), 311
 - assignment (=) (赋值), 328
 - begin, 369
 - bidirectional iterator (双向迭代器), 417
 - clear, 359
 - constructors (构造函数), 360
 - count, 367
 - definition (定义), 360
 - dereference yields pair (解引用生成 pair), 362
 - element type constraints (元素类型约束), 309
 - empty, 359
 - end, 369
 - equal_range, 379
 - erase, 359, 368
 - find, 368
 - header, (头文件) 360
 - insert, 364
 - iterator, 362
 - key type constraints (键类型约束), 360
 - key_type, 362
 - lower_bound, 377
 - mapped_type, 362, 388
 - operations (操作), 358
 - overriding the default comparison (覆盖默认比较), 604
 - rbegin, 412
 - rend, 412
 - return type from insert (insert 的返回类型), 365
 - reverse_iterator (反向迭代器), 412
 - size, 359
 - subscript operator (下标操作符), 363
 - supports relational operators (支持关系操作符), 359
 - swap, 329
 - upper_bound, 377
 - value_type, 361
- mapped_type, map, multimap, 362
- match, best (最佳匹配), 269, 280
- max, 822
- max_element, 822
- member (成员), 另见 class member
 - mutable data (mutable 数据), 474
 - pointer to (指针), 780, 807
- member function (成员函数), 25, 29, 431, 474
 - const, 280
 - equal, 778
 - as friend (作为友元), 466

- base member hidden by derived (被派生类隐藏的基类成员), 593
- class template (类模板), 653
 - defined outside class body (在类定义体外部定义的), 651
 - instantiation (实例化), 653
- const, 261, 262
- defined outside class body (在类定义体外部定义的), 261, 431
- definition (定义), 258–262
 - in class scope (在类作用域中), 445
 - name lookup (名字查找), 448
 - name lookup, examples (名字查找, 例子), 449
- function template (函数模板), 见 member template
- implicitly inline (隐式 inline), 259
- inline (内联函数), 437
 - and header (和头文件), 437
- overloaded (重载), 436
- overloaded on const (const 上重载), 442
- overloaded operator (重载操作符), 483, 508
- pointer to, definition (指针指向, 定义), 782
- returning *this (返回*this), 442
- static, 467
 - this pointer (this 指针), 469
- undefined (未定义的), 482
- member operator delete (operator delete 成员), 764, 806
 - and inheritance (和继承), 764
 - example (例子), 769
 - CachedObj, 770
 - interface (接口), 764
- member operator delete[] (operator delete[] 成员), 765
- member operator new (operator new 成员), 764, 806
 - example (例子), 769
 - CachedObj, 769
 - interface (接口), 764
- member operator new[] (operator new[] 成员), 765
- member template (成员模板), 660, 684
 - declaration (声明), 661
 - defined outside class body (在类定义体之外定义的), 662
 - examples (例子), 660
 - instantiation (实例化), 663
 - template parameters (模板形参), 663
- memberwise assignment (逐个成员赋值), 483, 503
- memberwise initialization (逐个成员初始化), 479, 503
- memory exhaustion (内存耗尽), 175
- memory leak (内存泄漏), 177, 485
 - after exception (异常之后), 700
- memory management, generic(泛型内存管理), 见 CachedObj, 816
- merge, 816
 - list, 423
- Message, 486~491
 - assignment operator (赋值操作符), 490
 - class definition (类定义), 488
 - copy constructor (复制构造函数), 489
 - design (设计), 486
 - destructor (析构函数), 491
 - put_Msg_in_Folder, 489
 - remove_Msg_from_Folder, 491
- method (方法), 见 member function, 29
- Microsoft compiler (微软编译器), 4
- min, 822
- min_element, 822
- minus<T>, 534
- mismatch, 814
- mode, file (文件模式), 296, 302
- modulus (%) (求模), 151
- modulus<T>, 534
- multi-dimensioned array (多维数组), 141–144
 - and pointer (和指针), 143
 - conversion to pointer (到指针的转换), 143
 - definition (定义), 142
 - initialization (初始化), 142
 - parameter (形参), 241
 - subscript operator (下标操作符), 142
- multi-line literal (多行字面值), 42
- multimap, 375, 388
 - assignment (=) (赋值), 328
 - begin, 369
 - clear, 359
 - constructors (构造函数), 360
 - count, 377
 - dereference yields pair (解引用生成 pair), 362
 - element type constraints (元素类型约束), 309
 - empty, 359
 - equal_range, 379
 - erase, 359, 376
 - find, 377
 - has no subscript operator (没有下标操作符), 376
 - insert, 376
 - iterator, 362, 376
 - key type constraints (键类型约束), 360
 - key_type, 362
 - lower_bound, 377
 - mapped_type, 362
 - operations (操作), 358, 376
 - overriding the default comparison (覆盖默认比较), 604
 - rbegin, 412
 - rend, 412

return type from insert (从 insert 的返回类型), 365
 reverse_iterator, 412
 size, 359
 supports relational operators (支持关系操作符), 359
 swap, 329
 upper_bound, 377
 value_type, 361
 multiple base class (多个基类), 750
 另见 virtual base class
 ambiguities (二义性), 738
 ambiguous conversion (二义性转换), 734
 avoiding potential name ambiguities (避免潜在的名字二义性), 738
 conversions (转换), 734
 definition (定义), 731
 destructor usually virtual (析构函数通常为虚函数), 736
 name lookup (名字查找), 737
 object composition (对象组成), 732
 order of construction (构造顺序), 732
 scope (作用域), 737
 virtual functions (虚函数), 735
 multiple inheritance (多重继承), 见 multiple base class, 731
 multiplication (*) (乘), 150
 multiplies<T>, 534
 multiset, 375, 388
 assignment(=) (赋值), 328
 begin, 372
 clear, 359
 constructors (构造函数), 372
 count, 377
 element type constraints (元素类型约束), 309
 end, 372
 equal_range, 379
 erase, 359, 376
 find, 377
 insert, 376
 iterator, 376
 key type constraints (键类型约束), 360
 lower_bound, 377
 Sales_item, 605
 operations (操作), 358, 376
 overriding the default comparison (覆盖默认比较), 604
 rbegin, 412
 rend, 412
 return type from insert (从 insert 的返回类型), 373
 reverse_iterator, 412
 supports relational operators (支持关系操作符), 359
 swap, 329
 upper_bound, 377
 example (例子), 607

value_type, 372
 mutable data member (mutable 数据成员), 443, 474

N

\n(newline character) (换行符), 40
 name lookup (名字查找), 447, 474
 and templates (和模板), 647
 before type checking (类型检查之前), 269, 593
 multiple inheritance (多重继承), 738
 class member declaration (类成员声明), 447
 class member definition (类成员定义), 448, 450
 class member definition, examples (类成员定义, 例子), 449
 collisions under inheritance (继承下的冲突), 591
 depends on static type (依赖于静态类型), 590
 multiple inheritance (多重继承), 735
 inheritance (继承), 590, 595
 local class (局部类), 797
 multiple inheritance (多重继承), 737
 ambiguous names (有二义的名字/歧义名字), 738
 namespace names (命名空间的名字), 724
 argument-dependent lookup (实参相关的查找), 726
 nested class (嵌套类), 791
 overloaded virtual functions (重载虚函数), 593
 virtual inheritance (虚继承), 743
 name resolution (名字确定), 见 name lookup
 namespace (命名空间), 8, 29, 712, 750
 class friend declaration scope (类友元声明作用域), 727
 cplusplus_primer, 714
 definition (定义), 712
 design (设计), 714
 discontiguous definition (不连续定义), 714
 function matching (函数匹配), 727
 global (全局的), 716
 member (成员), 713
 member definition (成员定义), 716
 outside namespace (在命名空间外部), 716
 restrictions (限制), 716
 nested (嵌套的), 717
 scope (作用域), 713–717
 unnamed (未命名的), 718
 local to file (局部于文件), 718
 replace file static (置换文件 static), 719
 namespace keyword (namespace 关键字), 712
 namespace alias (命名空间别名), 720, 750
 namespace pollution (命名空间污染), 712, 750
 naming convention (命名约定)
 header file (头文件), 264
 source file (源文件), 264
 NDEBUG, 220
 negate<T>, 534

- negator (求反器), 535, 553
 - nested class (嵌套类), 786, 806
 - access control (访问控制), 787
 - class defined outside enclosing class (在外围类外部定义的类型), 789
 - in class template (在类模板中), 788
 - in local class (在局部类中), 797
 - member defined outside class body (在类定义体外部定义的成员), 788
 - name lookup (名字查找), 791
 - QueueItem example (QueueItem 例子), 787
 - relationship to enclosing class (与外围类的关系), 787, 790
 - scope (作用域), 786
 - static members (static 成员), 790
 - union (联合), 794
 - nested namespace (嵌套命名空间), 717
 - nested type (嵌套类型), 见 `nsted class`, 786
 - new, 145, 174, 189, 806
 - compared to operator new (与 operator new 比较), 760
 - execution flow (执行流), 760
 - header (头文件), 219
 - member operator (成员 operator), 806
 - member operator, interface (成员 operator, 接口), 764
 - placement (定位), 761, 807
 - compared to construct (与 construct 比较), 762
 - new [], 135
 - new failure (new 失败), 175
 - next_permutation, 821
 - noboolalpha manipulator (noboolalpha 操纵符), 827
 - NoDefault, 459
 - nonconst reference (非 const 引用), 60
 - parameter (形参), 232
 - limitations (限制), 235
 - nonportable (不可移植的), 42
 - nonprintable character (非打印字符), 40, 75
 - nonreference (非引用)
 - parameter (形参), 230
 - uses copy constructor (使用复制构造函数), 478
 - return type (返回类型), 247
 - uses copy constructor (使用复制构造函数), 478
 - nontype template parameter (非类型模板形参), 625, 628, 632, 684
 - 另见 `template parameter`
 - class template (类模板), 655
 - must be constant expression (必须是常量表达式), 633
 - nonvirtual function, calls resolved at compile time (在编译时确定的非虚函数调用), 569
 - noshowbase manipulator (noshowbase 操纵符), 828
 - noshowpoint manipulator (noshowpoint 操纵符), 832
 - noskipws manipulator (noskipws 操纵符), 833
 - not equal (不相等), 见 `inequality`
 - not1, 535
 - not2, 535
 - not_equal_to<T>, 534
 - NotQuery, 609
 - definition (定义), 616
 - eval function (eval 函数), 620
 - nouppercase manipulator (nouppercase 操纵符), 828
 - nth_element, 818
 - NULL, 118
 - null pointer (空指针), 118
 - delete of (的 delete), 176
 - null statement (空语句), 192, 224
 - null-terminated array (以空字符结束的数组), 见 `C-style string`
 - number, magic (魔数), 56, 75
 - numeric header (numeric 头文件), 395
 - numeric literal (数值型字面值)
 - float (*numF* or *numf*), 39
 - long (*numL* or *numl*), 39
 - long double (*ddd.dddL* or *ddd.dddl*), 39
 - unsigned (*numU* or *numu*), 39
- ## O
- object (对象), 46, 75
 - automatic (自动), 255, 280
 - function (函数), 553
 - is not polymorphic (不是多态的), 569
 - local static (局部 static), 255, 281
 - temporary (临时的), 247
 - object cleanup (对象清除), 见 `destructor`
 - object creation (对象创建)
 - constructor (构造函数), 452
 - order of construction (构造顺序), 456
 - derived objects (派生类对象), 581, 582
 - multiple base classes (多个基类), 732
 - virtual base classes (虚基类), 745
 - order of destruction (析构顺序), 485
 - derived objects (派生类对象), 587
 - multiple base classes (多个基类), 733
 - virtual base classes (虚基类), 747
 - object file (目标文件), 68
 - object lifetime (对象生命期), 254, 281
 - and destructor (和析构函数), 485
 - compared to scope (作用域与比较), 254
 - object-oriented programming (面向对象编程), 285, 302, 622
 - key ideas in (关键概念), 558—560
 - oct manipulator (oct 操纵符), 827
 - octal, literal (*0num*) (八进制字面值), 38
 - octal escape sequence (*\nnn*) (八进制转义序列 (*\nnn*)), 40
 - off-the-end iterator (超出末端迭代器), 97, 394, 425

- istream_iterator, 408
- off-the-end pointer (超出末端的指针), 125
- ofstream, 285, 293–299
 - 另见 ostream
 - close, 294
 - constructor (构造函数), 293
 - file marker (文件标记), 838
 - file mode (文件模式), 296
 - combinations (组合), 298
 - example (例子), 299
 - file random access (文件随机访问), 838
 - off_type, 839
 - open, 293
 - pos_type, 839
 - random IO sample program (随机 IO 示例程序), 840
 - seek and tell members (seek 和 tell 成员), 838–842
- open, 293
- open_file, example of (open_file, 例子), 370, 383
- open_file program (open_file 程序), 299
- operand (操作数), 148, 189
 - order of evaluation (求值顺序)
 - comma operator (逗号操作符), 172
 - conditional operator (条件操作符), 172
 - logical operator (逻辑操作符), 172
- operator (操作符), 148, 189
 - sizeof, 167
 - typeid, 775, 807
 - addition (+) (加), 150
 - string, 86
 - iterator (迭代器), 101, 313
 - pointer (指针), 123
 - address-of (&) (取地址), 115
 - arrow (->) (箭头), 164
 - class member access (类成员访问), 445
 - assignment (=) (赋值), 13, 30, 159
 - and conversion (和转换), 179
 - and equality (和相等), 161
 - container (容器), 328
 - multiple inheritance (多重继承), 737
 - pointer (指针), 120
 - string, 86
 - to signed (到 signed), 37
 - to unsigned (到 unsigned), 37
 - yields lvalue (生成左值), 160
 - binary (二元的), 148, 188
 - bitwise AND (&) (位与), 156
 - bitwise exclusive or (^) (位异或), 156
 - bitwise not (~) (位求反), 155
 - bitwise OR (|) (位或), 156
 - bitwise OR (|), example (位或, 例子), 290
 - call (()) (调用), 30, 226
 - comma (,) (逗号), 168
 - operand order of evaluation (操作数求值顺序), 172
 - comma (,), example (逗号, 例子), 289
 - compound assginment (e.g., +=), iterator [复合赋值 (如 +=), 迭代器], 313
 - compound assignment (e.g., +=) [复合赋值 (如 +=)], 13, 30, 161
 - string, 86
 - arithmetic (算术的), 162
 - bitwise (位), 162
 - conditional (?:) (条件), 165
 - operand order of evaluation (操作数求值顺序), 172
 - decrement (--) (自减)
 - iterator (迭代器), 312
 - prefix yields lvalue (前缀生成左值), 162
 - reverse iterator (反向迭代器), 412
 - dereference (*) (解引用), 98
 - and increment (和自增), 163
 - iterator (迭代器), 98
 - on map yields pair (用于 map 生成 pair), 362
 - pointer (指针), 119
 - yields lvalue (生成左值), 99, 120
 - division (/) (除), 150
 - dot (.) (点), 25, 30
 - class member access (类成员访问), 445
 - equality(==) (相等), 30, 154
 - string, 85
 - algorithm (算法), 421
 - container (容器), 322
 - container adaptor (容器适配器), 350
 - iterator (迭代器), 98, 312
 - string, 347
 - greater-than(>) (大于), 30, 153
 - greater-than-or-equal(>=) (大于或等于), 30, 153
 - increment(++), (自增) 13, 30
 - and dereference (和解引用), 163
 - iterator (迭代器), 98, 312
 - pointer (指针), 114
 - prefix yields lvalue (前缀生成左值), 162
 - reverse iterator (反向迭代器), 412
 - inequality(!=) (不相等), 30, 154
 - container (容器), 322
 - container adaptor (容器适配器), 350
 - iterator (迭代器), 98, 312
 - string, 347
 - input(>>) (输入), 8, 30
 - sales_item, 516
 - istream_iterator, 408
 - string (字符串), 81

- precedence and associativity (优先级与结合性), 158
- left-shift(<<) (左移), 155, 190
- less-than(<) (小于), 30, 153
 - used by algorithm (算法使用的), 420
- less-than-or-equal(<=) (小于或等于), 13, 30, 153
- logical AND(&&) (逻辑与), 152
 - operand order of evaluation (操作数求值顺序), 172
- logical NOT(!) (逻辑非), 153
- logical OR(||) (逻辑或), 152
 - operand order of evaluation (操作数求值顺序), 172
- modulus(%) (求模), 151
- multiplication(*) (乘), 150
- output(<<) (输出), 7, 30
 - bitset, 106
 - ostream_iterator, 408
 - string, 81
- precedence and associativity (优先级与结合性), 158
- overloaded (重载), 189, 482, 503
- pointer to member (成员指针)
 - arrow (->*) (箭头), 783
 - dot (.*) (点), 783
- right-shift(>>) (右移), 155, 190
- scope(::) (作用域), 8, 30, 78
 - class member (类成员), 85, 445
 - container defined type (定义类型的容器), 317
 - member function definition (成员函数定义), 262
 - to override name lookup (覆盖名字查找), 449
- shift (移位), 155, 190
- sizeof, 167
- subscript([]) (下标)
 - bitset, 105
 - deque, 325
 - map, 363
 - string, 87
 - vector, 94, 325
 - and multi-dimensioned array (和多维数组), 142
 - and pointer (和指针), 124
 - array (数组), 113
 - valid subscript range (有效下标范围), 88
 - yields lvalue (生成左值), 88
- subtraction(-) (减), 150
 - iterator (迭代器), 101, 313
 - pointer (指针), 123
- unary (一元的), 148, 189
- unary minus(-) (一元减), 150
- unary plus(+) (一元加), 150
- operator alternative name (操作符可选名字), 46
- operator delete function (operator delete 函数), 760, 806
 - compared to deallocate (与 deallocate 比较), 761
 - compared to delete expression (与 delete 表达式比较), 760
- operator delete member (operator delete 成员), 764
 - and inheritance (和继承), 764
 - example (例子), 769
 - CachedObj, 770
 - interface (接口), 764
- operator delete [] member (operator delete [] 成员), 765
- operator new function (operator new 函数), 760, 806
 - compared to allocate (与 allocate 比较), 761
 - compared to new expression (与 new 表达式比较), 760
- operator new member (operator new 成员), 764
 - example (例子), 769
 - CachedObj, 769
 - interface (接口), 764
- operator new [] member (operator new [] 成员), 765
- operator overloading (操作符重载), 见 overloaded operator
- options to main (main 的选项), 243
- order of construction (构造顺序), 456, 749
 - derived objects (派生类对象), 581, 582
 - multiple base classes (多个基类), 732
 - virtual base classes (虚基类), 746
- order of destruction (析构顺序), 485, 749
 - derived objects (派生类对象), 587
 - multiple base classes (多个基类), 733
 - virtual base classes (虚基类), 747
- order of evaluation (求值顺序), 149, 189
 - comma operator (逗号操作符), 172
 - conditional operator (条件操作符), 172
 - logical operator (逻辑操作符), 172
- ordering, strict weak (严格弱排序), 360, 389
- OrQuery, 609
 - definition (定义), 618
 - eval function (eval 函数), 619
- ostream, 6, 29, 285
 - 另见 manipulator
 - condition state (条件状态), 287
 - floatfield member (floatfield 成员), 831
 - flushing output buffer (刷新输出缓冲区), 290
 - format state (格式状态), 825
 - inheritance hierarchy (继承层次), 740
 - no containers of (没有容器), 310
 - no copy or assign (不可复制或求值), 287
 - not flushed if program crashes (若程序崩溃则不刷新), 292
 - output(<<) (输出), 7
 - precedence and associativity (优先级与结合性), 158
- precision member (precision 成员), 829
- seek and tell members (seek 和 tell 成员), 838
- tie member (tie 成员), 292

- unsetf member (unsetf 成员), 831
 - ostream_iterator, 407, 425
 - and class type (和类类型), 410
 - constructors (构造函数), 408
 - limitations (限制), 411
 - operations (操作), 410
 - output iterator (输出迭代器), 417
 - output operator (<<) (输出操作符), 408
 - used with algorithms (与算法一起使用的), 411
 - ostringstream, 285, 299–301
 - 另见 ostream
 - str, 301
 - out (file mode) (out 文件模式), 296
 - out_of_range, 219, 325
 - out_of__stock, 699
 - output, standard (标准输出), 6
 - output (<<) (输出), 7, 30
 - bitset, 106
 - ostream_iterator, 408
 - string, 81, 108
 - overloaded operator (重载操作符), 513
 - formatting (格式化), 514
 - must be nonmember (必须为非成员), 514
 - precedence and associativity (优先级与结合性), 158
 - Sales_item, 514
 - output iterator (输出迭代器), 416, 425
 - overflow (溢出), 150
 - overflow_error, 219
 - overload resolution (重载确定), 见 function matching
 - overloaded function (重载函数), 265, 281
 - using declarations (using 声明), 728
 - using directive (using 指示), 729
 - compared to redeclaration (与重复声明比较), 266
 - compared to template specialization (与模板特化比较), 673
 - friend declaration (友元声明), 467
 - linkage directive (链接指示), 803
 - namespaces (命名空间), 727
 - scope (作用域), 268
 - virtual (虚), 593
 - overloaded member function (重载成员函数), 436
 - on const (关于 const), 442
 - overloaded operator (重载操作符), 189, 482, 503
 - << (output operator) (输出操作符), 513
 - * (dereference) (解引用), 524
 - & (address-of) (取地址), 511
 - > (arrow operator) (箭头操作符), 525
 - [] (subscript) (下标操作符), 522
 - reference return (引用返回), 522
 - () (call operator) (调用操作符), 530
 - = (assignment) (赋值), 476, 483, 520
 - and copy constructor (和复制构造函数), 484
 - check for self-assignment (检查自身赋值), 490
 - Message, 490
 - reference return (引用返回), 483, 521
 - rule of three (三法则), 485
 - use counting (使用计数), 495, 498
 - valuelike classes (值型类), 501
 - >> (input operator) (输入操作符), 515
 - error handling (错误处理), 516–517
 - must be nonmember (必须是非成员), 514
 - << (output operator) (输出操作符)
 - formatting (格式化), 514
 - must be nonmember (必须是非成员), 514
 - Sales_item, 514
 - &&(logical AND) (逻辑与), 511
 - ||(logical OR) (逻辑或), 511
 - ,(comma operator) (逗号操作符), 511
 - addition (+), Sales_item[加(+), Sales_item], 517
 - ambiguous[二义性的], 550
 - arithmetic operators (算术操作符), 517
 - as virtual function (作为虚函数), 615
 - binary operator (二元操作符), 508
 - candidate functions (候选函数), 549
 - compound assignment (e.g., +=)[复合赋值 (如 +=)], 511
 - Sales_item, 521
 - consistency between relational and equality operators (关系操作符与相等操作符的一致性), 520
 - definition (定义), 482, 506
 - design (设计), 510–513
 - equality operators (相等操作符), 512, 518
 - explicit call to (显式调用), 509
 - explicit call to postfix operators (后缀操作符的显式调用), 529
 - function matching (函数匹配), 547–551
 - member and this pointer (成员与 this 指针), 483
 - member vs. nonmember function (成员函数与非成员函数), 508, 512
 - postfix increment (++) and decrement (--) operators (后自增与后自减操作符), 528
 - precedence and associativity (优先级与结合性), 507
 - prefix increment (++) and decrement (--) operators (前自增与前自减操作符), 527
 - relational operators (关系操作符), 511, 520
 - require class-type parameter (需要类类型形参), 507
 - unary operator (一元操作符), 508
- overloading (重载), 见 overloaded function
- operator (操作符), 见 overloaded operator

P

pair, 356, 388

- as return type from `map::insert`(作为 `map::insert` 的返回类型), 365
- as return type from `set::insert`(作为 `set::insert` 的返回类型), 373
- default constructor (默认构造函数), 357
- definition (定义), 356
- initialization (初始化), 356
- `make_pair`, 358
- operations (操作), 357
- public data members (公用数据成员), 357
- Panda, 731
 - virtual inheritance (虚继承), 741
- parameter (形参), 226, 227, 281
 - array and buffer overflow (数组与缓冲区溢出), 242
 - array type (数组类型), 238–244
 - C-style string (C 风格字符串), 242
 - `const`, 231
 - `const` reference (`const` 引用), 235
 - overloading (重载), 275
 - ellipsis (省略符), 244
 - function pointer (函数指针), 278
 - linkage directive (链接指示), 804
 - initialization of (初始化), 229
 - iterator (迭代器), 238, 242
 - library container (库容器), 237
 - lifetime (生命期), 255
 - local copy (局部复制), 230
 - matching (匹配), 229
 - ellipsis (省略符), 244
 - template specialization (模板特化), 673
 - with class type conversion (带类类型转换的), 541
 - multi-dimensioned array (多维数组), 241
 - nonconst reference (非 `const` 引用), 232
 - nonreference type (非引用类型), 230
 - uses copy constructor (使用复制构造函数), 478
 - of member function (成员函数的), 260
 - vector type (`vector` 类型), 237
 - passing (传递), 229
 - pointer to `const` (指向 `const` 的指针), 231
 - overloading (重载), 275
 - pointer to function (指向函数的指针), 278
 - linkage directive (链接指示), 804
 - pointer to nonconst (指向非 `const` 的指针), 231
 - pointer type (指针类型), 231, 239
 - reference (引用)
 - to array type (数组类型的), 240
 - to pointer (指针的), 236
 - template (模板), 见 `template parameter`
 - and `main` (和 `main`), 243
 - type checking (类型检查)
 - and `template argument` (和模板实参), 638
 - of reference to array (引用数组的), 240
- parameter list (形参列表), 3, 29, 226, 228
 - member function definition (成员函数定义), 446
- parentheses, override precedence (圆括号凌驾于优先级之上), 169
- partial specialization (部分特化), 678, 684
- `partial_sort`, 818
- `partial_sort_copy`, 818
- `partial_sum`, 824
- `partition`, 817
- placement new (定位 new), 761, 807
 - compared to `construct` (与 `construct` 比较), 762
- `plus<T>`, 534
- pointer (指针), 114, 114–126, 146
 - array (数组), 122
 - arrow (`->`) (箭头), 164
 - as initializer of `vector` (作为 `vector` 的初始化式), 140
 - as parameter (作为形参), 231
 - assignment (赋值), 120
 - `char*`, 见 C-style string
 - class member copy control (类成员复制控制), 492–501
 - copy constructor (复制构造函数), 480
 - destructor (析构函数), 485
 - strategies (策略), 499
 - compared to iterator (与迭代器比较), 114
 - compared to reference (与引用比较), 121
 - `const`, 128
 - `const` pointer to `const` (指向 `const` 的 `const` 指针), 129
 - container constructor from (容器构造函数), 308
 - conversion from derived to base (从派生类到基类的转换), 567
 - conversion from derived to multiple base (从派生类到多个基类的转换), 734
 - conversion to `bool` (到 `bool` 的转换), 182
 - conversion to `void` (到 `void` 的转换), 181
 - dangling (悬垂), 176, 188
 - synthesized copy control (合成的复制控制), 494
 - declaration style (声明风格), 116–117
 - definition (定义), 115
 - `delete`, 176
 - `dynamic_cast`, example (`dynamic_cast`, 例子), 773
 - function returning (函数返回), 228
 - implicit `this` (隐式 `this`), 260, 281
 - initialization (初始化), 117–119
 - is polymorphic (是多态的), 569
 - multi-dimensioned array (多维数组), 143
 - `new`, 174
 - `null`, 118
 - off-the-end (超出末端), 125

- pitfalls with generic programs (泛型程序的缺陷), 671
- reference parameter (引用形参), 236
- relational operator (关系操作符), 132
- return type and local variable (返回类型与局部变量), 249
- smart (智能), 495, 503, 553
 - handle class (句柄类), 599
 - overloaded `->` (arrow operator) and `*` (dereference)[重载`->` 箭头操作符和`*` (解引用)], 524
 - overloaded `++` and `*`[重载 `++` 和`*`], 526
- subscript operator (下标操作符), 124
- to pointer (指向指针的), 122
- typedef, 129
- typeid operator (typeid 操作符), 776
- uninitialized (未初始化的), 117
- volatile, 800
- pointer arithmetic (指针算术操作), 123, 146
- pointer to const (指向 const 的指针), 127
 - argument (实参), 231
 - conversion from nonconst (从非 const 转换), 127
 - parameter (形参), 231
 - overloading (重载的), 275
- pointer to function (指向函数的指针), 276—279
 - definition (定义), 276
 - exception specifications (异常说明), 711
 - function returning (函数返回的), 228
 - initialization (初始化), 277
 - linkage directive (链接指令), 803
 - overloaded functions (重载函数), 279
 - parameter (形参), 278
 - return type (返回类型), 278
 - typedef, 276
- pointer to member (成员指针), 780, 807
 - and typedef (和类型别名), 783
 - arrow (`->*`) (箭头), 783
 - definition (定义), 781
 - dot (`.*`) (点), 783
 - function pointer (函数指针), 782
 - function table (函数表), 785
- pointer to nonconst (指向非 const 的指针)
 - argument (实参), 231
 - parameter (形参), 231
- polymorphism (多态性), 558, 622
 - compile time polymorphism via templates (借助于模板的编译时多态性), 624
 - run time polymorphism in C++ (C++ 中的运行时多态性), 569
- pop
 - priority_queue, 352
 - queue, 352
 - stack, 351
- pop_back, sequential container (pop_back, 顺序容器), 326
- pop_front, sequential container (pop_front, 顺序容器), 326
- portable (可移植的), 797
- postfix decrement (`--`) (后自减)
 - overloaded operator (重载操作符), 528
 - yields rvalue (生成左值), 163
- postfix increment (`++`) (后自增)
 - and dereference (和解引用), 163
 - overloaded operator (重载操作符), 528
- precedence (优先级), 124, 146, 149, 168, 189
 - of assignment (赋值的), 160
 - of conditional (条件的), 166
 - of dot and dereference (点和解引用的), 164
 - of increment and dereference (自增和解引用的), 163
 - of IO operator (IO 操作符的), 158
 - of pointer to member and call operator (成员指针和调用操作符的), 782
 - overloaded operator (重载操作符), 507
 - pointer parameter declaration (指针形参声明), 241
- precedence table (优先级表), 170
- predicate (谓词), 402, 425
- prefix decrement (`--`) (前自减), 163
 - overloaded operator (重载操作符), 527
 - yields lvalue (生成左值), 162
- prefix increment (`++`) (前自增)
 - and dereference (和解引用), 163
 - overloaded operator (重载操作符), 527
 - yields lvalue (生成左值), 162
- preprocessor (预处理器), 70, 75
 - directive (指示), 7, 29
 - macro (宏), 221, 224
 - variable (变量), 71
- prev_permutation, 821
- preventing copies of class objects (禁止类对象的复制), 481
- print_total, 559
 - explained (解释), 568
- printable character (可打印字符), 88
- printValues program (printValues 程序), 240, 242, 243
- priority_queue, 348, 354
 - constructors (构造函数), 349
 - relational operator (关系操作符), 350
- private (私有的)
 - class (类), 496
 - copy constructor (复制构造函数), 481
 - inheritance (继承), 571
 - member (成员), 75, 474
- private access label (private 访问标号), 65, 432
 - inheritance (继承), 561
- private inheritance (私有继承), 622
- program (程序)

factorial, 250
 find_val, 234
 gcd, 226
 isShorter, 235
 make_plural, 248
 open_file, 299
 printValues, 240, 242, 243
 ptr_swap, 237
 rgcd, 250
 swap, 233, 245
 vector capacity (vector 容量), 331
 book finding (书籍查找)
 using equal_range (使用 equal_range), 379
 using find (使用 find), 377
 using upper_bound (使用 upper_bound), 378
 bookstore (书店), 26
 bookstore exception classes (书店异常类), 698
 CachedObj, 766
 duplicate words (复制单词), 400—404
 revisited (再谈), 531
 find last word (查找最后一个单词), 414
 GT6, 403
 Handle class (Handle 类), 667
 int instantiation (int 实例化), 668
 operations (操作), 668
 Sales_item instantiation (Sales_item 实例化), 669
 isShorter, 403
 message handling classes (消息处理类), 486
 Query
 design (设计), 609—611
 interface (接口), 610
 operations (操作), 607
 Queue, 648
 copy_elems member (copy_elems 成员), 652
 destroy member (destroy 成员), 651
 pop member (pop 成员), 651
 push member (push 成员), 652
 random IO example (随机 IO 例子), 840
 restricted word count (受限的单词计数), 374
 Sales_item handle class (Sales_item 句柄类), 599
 Screen class template (Screen 类模板), 655
 TextQuery, 383
 class definition (类定义), 382
 design (设计), 380
 interface (接口), 381
 vector capacity (vector 容量), 331
 vowel counting (元音统计), 200
 word count (单词计数), 363
 word transformation (单词转换), 370
 ZooAnimal class hierarchy (ZooAnimal 类层次), 731

programmer-defined header (程序员定义的头文件), 67—72
 programming (编程)
 generic (泛型编程), 95, 624
 object-oriented (面向对象编程), 285, 302, 622
 promotion, integral (整型提升), 180, 189
 protected, inheritance (受保护的继承), 571, 622
 protected access label (受保护的访问标号), 562, 622
 protected keyword (protected 关键字), 562
 prototype, function (函数原型), 251, 281
 ptr_swap program (ptr_swap 程序), 237
 ptrdiff_t, 123, 146
 public (公用的)
 inheritance (继承), 571, 622
 member (成员), 75, 474
 public access label (public 访问标号), 65, 432
 inheritance (继承), 561
 pure virtual function (纯虚函数), 596, 622
 example (例子), 609
 push
 priority_queue, 352
 queue, 352
 stack, 351
 push_back, 94, 108
 vector, 94
 back_inserter, 399
 sequential container (顺序容器), 318
 push_front
 front_inserter, 406
 sequential container (顺序容器), 318
 put_Msg_in_Folder, 489

Q

Query, 610
 & (bitwise AND (位与), 610
 definition (定义), 614
 ~ (bitwise NOT) (位求反), 610
 definition (定义), 614
 |(bitwise OR) (位或), 610
 definition (定义), 614
 <<(output operator) (输出操作符), 615
 definition (定义), 613
 design (设计), 609—611
 interface (接口), 610
 operations (操作), 607
 Query_base, 609
 definition (定义), 612
 member functions (成员函数), 609
 Queue
 <<(output operator) (输出操作符), 659
 assign, 662

copy_elems member (copy_elems 成员), 652, 662
 definition (定义), 648
 design (设计), 647
 destroy member (destroy 成员), 651
 final class definition (最终类定义), 664
 interface (接口), 627
 member template declarations (成员模板声明), 661
 operations (操作), 627
 pop member (pop 成员), 651
 push, specialized (push, 特化的), 677
 push member (push 成员), 652
 template version, char* (模板版本, char*), 675
 queue, 348, 354
 constructors (构造函数), 349
 relational operator (关系操作符), 350
 QueueItem, 648
 as nested class (作为嵌套类), 787
 constructor (构造函数), 789
 definition (定义), 788
 friendship (友元关系), 658
 CachedObj, 768
 allocation explained (分配解释的), 769

R

Raccoon as virtual base (作为虚基类的 Raccoon), 741
 RAII, 见 resource allocation is initialization
 raise (引发), 750
 raise exception (引发异常), 见 throw
 random file IO (随机文件 IO), 838
 random-access iterator (随机访问迭代器), 417, 425
 deque, 417
 string, 417
 vector, 417
 random_shuffle, 820
 range (范围)
 iterator (迭代器), 314, 314–316, 354
 left-inclusive (左闭合的), 314
 range_error, 219
 rbegin, container (rbegin, 容器), 317, 412
 rdstate, 290
 recursive function (递归函数), 249, 281
 refactoring (重构), 583, 622
 referece, 317
 reference (引用), 58, 75
 and pointer (和指针), 121
 const reference (const 引用), 59
 initialization (初始化), 60
 conversion from derived to base (从派生类到基类的转换), 567
 conversion from derived to multiple base (从派生类到多个基类的转换), 734
 dynamic_cast operator, example (dynamic_cast 操作符, 例子), 774
 is polymorphic (是多态的), 569
 nonconst reference (非 const 引用), 60
 parameter (形参), 232–237
 pointer parameter (指针形参), 236
 return type, is lvalue (返回类型, 是左值), 249
 return type and class object (返回类型与类对象), 440
 return type and local variable (返回类型与局部变量), 249
 return value (返回值), 248
 to array parameter (数组形参的), 240
 reference count (引用计数), 见 use count
 reference data member, initialization (引用数据成员, 初始化), 455
 reference to const (对 const 的引用), 见 const reference
 reinterpret_cast, 183, 185
 relational operator (关系操作符), 153
 string, 85
 associative container (关联容器), 359
 container (容器), 321
 container adaptor (容器适配器), 350
 function object (函数对象), 533
 overloaded operator (重载操作符), 511, 520
 consistent with equality (与相等一致的), 520
 pointer (指针), 132
 remove, 819
 list, 423
 remove_copy, 820
 remove_copy_if, 820
 remove_if, 819
 list, 423
 remove_Msg_from_Folder, 491
 rend, container (rend 容器), 317, 412
 replace, 400, 816
 string, 342
 replace_copy, 400, 815
 replace_copy_if, 815
 replace_if, 816
 reserve
 string, 336
 vector, 331
 reserved identifier (保留标识符), 47
 resize, sequential container (resize, 顺序容器), 323
 Resource, 700
 resource allocation is initialization (资源分配即初始化), 700–701
 auto_ptr, 702
 restricted word count program (受限的单词计数程序), 374
 result (结果), 148, 189

- rethrow (重新抛出), 695, 750
 - return, container (return, 容器), 381
 - return statement (return 语句), 245—251
 - from main (从 main 返回的), 247
 - implicit (隐式的), 245
 - local variable (局部变量), 247, 249
 - return type (返回类型), 3, 29, 226, 227
 - const reference (const 引用), 249
 - function (函数), 281
 - function pointer (函数指针), 278
 - linkage directive (链接指令), 804
 - member function definition (成员函数定义), 446
 - no implicit return type (没有隐式返回类型), 228
 - nonreference (非引用), 247
 - uses copy constructor (使用复制构造函数), 478
 - of virtual function (虚函数的), 564
 - pointer to function (指向函数的指针), 278
 - reference (引用), 248
 - reference yields lvalue (引用生成左值), 249
 - void, 245
 - return value (返回值)
 - conversion (转换), 246
 - copied (复制的), 247
 - reverse, 819
 - list, 423
 - reverse iterator (反向迭代器), 405, 412—415, 425
 - (decrement) (自减), 412
 - ++ (increment) (自增), 412
 - base, 414
 - compared to iterator (与迭代器比较), 413, 414
 - example (例子), 414
 - requires -- (decrement)[需要-- (自减)], 413
 - reverse_copy, 819
 - reverse_iterator, 412
 - container (容器), 316
 - rfind, string, 346
 - rgcd program (rgcd 程序), 250
 - right manipulator (right 操纵符), 832
 - right-shift (>>) (右移), 155, 190
 - rotate, 819
 - rotate_copy, 820
 - rule of three (三法则), 485, 503
 - exception for virtual destructors (虚析构函数的例外), 588
 - run time (运行时), 75
 - error (错误), 38
 - run-time type identification (运行时类型识别), 772—780, 807
 - classes with virtual functions (带虚函数的类), 772
 - compared to virtual functions (与虚函数比较), 777
 - dynamic_cast, 773
 - example (例子), 773
 - throws bad_cast (抛出 bad_cast), 774
 - to pointer (指针), 773
 - to reference (引用), 774
 - type-sensitive equality (类型敏感的相等), 778
 - typeid, 775
 - and virtual functions (和虚函数), 775
 - example (例子), 776
 - returns type_info (返回 type_info), 776
 - runtime_error, 217, 219
 - constructor from string (源自 string 的构造函数), 218
 - rvalue (右值), 45, 75
- ## S
- safety, exception (异常安全), 700
 - Sales_item, 21
 - addition (+) (加), 23, 517
 - throws exception (抛出异常), 217, 699
 - class definition (类定义), 64, 258-265
 - compare function (compare 函数), 604
 - compound assignment (e.g., +=)[复合赋值 (如 +=)], 521
 - conversion (转换), 461
 - default constructor (默认构造函数), 263
 - equality operators (==), (!=) (相等操作符), 519
 - explicit constructor (explicit 构造函数), 462
 - handle class (句柄类), 599
 - clone function (clone 函数), 602
 - constructor (构造函数), 601, 602
 - definition (定义), 600
 - design (设计), 599
 - multiset of Sales_item 的 multiset, 605
 - using generic Handle (使用泛型 Handle 类), 669
 - header (头文件), 21, 67, 264
 - input (>>) (输入), 516
 - istream constructor (istream 构造函数), 452
 - no relational operators (非关系操作符), 520
 - operations (操作), 21
 - output (<<) (输出), 514
 - avg_price definition (avg_price 定义), 261
 - same_isbn, 24, 258
 - string constructor (string 构造函数), 452
 - scientific manipulator (scientific 操纵符), 830
 - scope (作用域), 54, 75
 - const object (const 对象), 57, 69
 - block (块), 193
 - class (类), 65, 444, 473
 - compared to object lifetime (与对象生命期比较), 254
 - for statement (for 语句), 15
 - friend declaration (友元声明), 466
 - function (函数), 227
 - function template specialization (函数模板特化), 674

- global (全局的), 54, 74
- local (局部的), 54, 75
- multiple inheritance (多重继承), 737
- namespace (命名空间), 713–717
- statement (语句), 194
- template parameter (模板形参), 629
- using declaration (using 声明), 720
- using directive (using 指示), 721
 - example (例子), 722
 - name collisions (名字冲突), 723
- scope (::)[作用域(::)]
 - base class members (基类成员), 569
 - namespace member (命名空间成员), 750
- scope operator (::) (作用域操作符), 8, 30, 78, 108
 - class member (类成员), 85, 445
 - container defined type (容器定义的类型), 317
 - member function definition (成员函数定义), 262
 - namespace member (命名空间成员), 713
 - to override class-specific memory allocation (覆盖类特定的内存分配), 765
 - to override name lookup (覆盖名字查找), 449
- Screen, 435
 - class template (类模板), 655
 - concatenating operations (连接操作符), 441
 - display, 442
 - do_display, 442
 - friends (友元), 465
 - get definition (get 定义), 446
 - get members (get 成员), 436
 - get_cursor definition (get_cursor 定义), 446
 - CachedObj, 768
 - Menu function table (Menu 函数表), 785
 - move members (move 成员), 441
 - set members (set 成员), 441
 - simplified (简化的), 781
 - size_type, 435
- ScreenPtr, 523
 - arrow operator (->) (箭头操作符), 525
 - dereference (*) (解引用), 524
 - use counted (使用计数的), 523
- ScrPtr, 523
- search, 813
- search_n, 813
- self-assignment (自身赋值)
 - auto_ptr, 705
 - check (检查), 490
 - use counting (使用计数), 498
- semantics, value (语义, 值), 499, 503
- semicolon (;) (分号), 3
- semicolon (;), class definition (分号, 类定义), 440
- sentinel (哨兵), 97, 108
- separate compilation (分别编译), 67, 76
 - inclusion model for templates (模板的包含模型), 644
 - of templates (模板的), 643
 - separate compilation model for templates (模板的分别编译模型), 645, 684
- sequence, escape (转义序列), 74
- sequence (\Xnnn), hexadecimal escape (十六进制转义序列), 40
- sequential container (顺序容器), 306, 354
 - assign, 328
 - assignment (=) (赋值), 328
 - back, 324
 - clear, 327
 - const_iterator, 316
 - const_reverse_iterator, 316
 - constructor from element count, uses copy constructor (来自元素计数的构造函数, 使用复制构造函数), 478
 - uses element default constructor (使用元素默认构造函数), 460
 - constructors (构造函数), 307–309
 - deque, 306
 - element type constraints (元素类型约束), 309, 323
 - empty, 323
 - erase, 326
 - front, 324
 - insert, 319
 - iterator, 316
 - list, 306
 - operations (操作), 316–330
 - performance characteristics (性能特征), 333
 - pop_back, 326
 - pop_front, 326
 - priority_queue, 348
 - push_back, 318
 - push_front, 318
 - queue, 348
 - rbegin, 412
 - rend, 412
 - resize, 323
 - returning a (返回一个顺序容器), 381
 - reverse_iterator, 316, 412
 - size, 323
 - size_type, 316
 - stack, 348
 - supports relational operators (支持关系操作符), 321
 - swap, 329
 - types defined by (定义的类型), 316
 - value_type, 317
 - vector, 306

- set, 356, 388
 - as element type (作为元素类型), 311
 - assignment (=) (赋值), 328
 - begin, 372
 - bidirectional iterator (双向迭代器), 417
 - clear, 359
 - constructors (构造函数), 372
 - count, 372
 - element type constraints (元素类型约束), 309
 - empty, 359
 - end, 372
 - equal_range, 379
 - erase, 359, 372
 - find, 372
 - insert, 373
 - iterator, 374
 - key type constraints (键类型约束), 360
 - lower_bound, 377
 - operations (操作), 358
 - overriding the default comparison (覆盖默认比较), 604
 - rbegin, 412
 - rend, 412
 - return alternatives (返回可选值), 381
 - return type from insert (insert 的返回类型), 373
 - reverse_iterator, 412
 - size, 359
 - supports relational operators (支持关系操作符), 359
 - swap, 329
 - upper_bound, 377
 - value_type, 372
- set_difference, 822
- set_intersection, 619, 822
- set_symmetric_difference, 822
- set_union, 822
- setfill manipulator (setfill 操纵符), 832
- setprecision manipulator (setprecision 操纵符), 829
- setstate, 289, 290
- setw manipulator (setw 操纵符), 832
- shift operator (移位操作符), 155, 190
- short, 34
- short-circuit evaluation (短路求值), 152
 - overloaded operator (重载操作符), 508
- shorterString, 248
- showbase manipulator (showbase 操纵符), 827
- showpoint manipulator (showpoint 操纵符), 831
- signed, 35, 76
 - conversion to unsigned (转换到 unsigned), 36, 180
- size, 108
 - string, 83
 - vector, 93
- associative container (关联容器), 359
- priority_queue, 352
- queue, 352
- sequential container (顺序容器), 323
- stack, 351
- size_t, 104, 108, 146
 - and array (和数组), 113
- size_type, 84, 108
 - string, 84
 - vector, 93
 - container (容器), 316
- sizeof operator (sizeof 操作符), 167
- skipws manipulator (skipws 操纵符), 833
- sliced (切掉的), 579, 622
- SmallInt, 536, 550
 - conversion operator (转换操作符), 537
- smart pointer (智能指针), 495, 503, 553
 - handle class (句柄类), 599
 - overloaded -> (arrow operator) and * (dereference) [重载->(箭头操作符)与*(解引用)], 524
 - overloaded (++) and (*) [重载(++)与(*)], 526
- sort, 401, 817
- source file (源文件), 4, 29
 - naming convention (命名约定), 264
- specialization (特化)
 - class template (类模板)
 - definition (定义), 675
 - member defined outside class body (在类定义体外部定义的成员), 676
 - partial (部分), 678
 - partial specialization (部分特化), 684
- class template member (类模板成员), 677
 - declaration (声明), 677
- function template (函数模板)
 - compared to overloaded function (与重载函数比较), 673
 - declaration (声明), 672, 673
 - example (例子), 672
 - scope (作用域), 674
 - template, namespaces (模板, 命名空间), 730
- specifier, type (类型说明符), 48, 76
- splice, list, 423
- sstream
 - header (头文件), 285, 300
 - str, 301
- stable_partition, 817
- stable_sort, 403, 817
- stack, 348, 354
 - constructors (构造函数), 349
 - relational operator (关系操作符), 350
- stack unwinding (栈展开), 691, 750

- standard error (标准错误), 6, 29
- standard input (标准输入), 6, 29
- standard library (标准库), 5, 29
- standard output (标准输出), 6, 30
- state, condition (条件状态), 302
- statement (语句), 2, 30
 - break, 212, 223
 - continue, 214, 223
 - do while, 210
 - for, 29, 207
 - goto, 214, 224
 - if, 17, 29, 195, 224
 - return, 245–251
 - switch, 199, 224
 - while, 12, 30, 204, 224
 - compound (复合的), 193, 223
 - declaration (声明), 193, 224
 - expression (表达式), 192, 224
 - for statement (for 语句), 14
 - labeled (带标号的), 214, 224
 - null (空的), 192, 224
 - return, local variable (return, 局部变量), 247, 249
- statement block (语句块), 见 block
- statement label (语句标号), 214
- statement scope (语句作用域), 194
- statement, for (for 语句), 14
- static (file static) (文件静态), 719
- static class member (static 类成员), 467, 474
 - as default argument (作为默认实参), 471
- class template (类模板), 665
 - accessed through an instantiation (通过实例化访问), 666
 - definition (定义), 666
- const data member, initialization (const 数据成员, 初始化), 470
- const member function (const 成员函数), 469
- data member (数据成员), 469
 - as constant expression (作为常量表达式), 471
- inheritance (继承), 576
- member function (成员函数), 467
 - this pointer (this 指针), 469
- static object, local (局部 static 对象), 255, 281
- static type (静态类型), 568, 622
 - determines name lookup (确定名字查找), 590
 - multiple inheritance (多重继承), 735
- static type checking (静态类型检查), 44, 76
 - argument (实参), 229
 - function return value (函数返回值), 246
- static_cast, 183, 185
- std, 8, 30
- stdexcept header (stdexcept 头文件), 217, 219
- store, free (自由存储区), 135, 145
- str, 301
- strcat, 133
- strcmp, 133
- strcpy, 133
- stream 流
 - istream_iterator, 407
 - ostream_iterator, 407
 - flushing buffer (刷新缓冲区), 290
 - iterator (迭代器), 405, 407–412
 - and class type (和类类型), 410
 - limitations (限制), 411
 - used with algorithms (与算法一起使用), 411
 - not flushed if program crashes (如果程序崩溃了, 则不会刷新), 292
 - type as condition (类型作为条件), 19
- stream iterator (流迭代器), 425
- strict weak ordering (严格弱排序), 360, 389
- string, C-style (C 风格字符串), 见 C-style string
- string, 80–89
 - addition (加), 86
 - addition to string literal (加至字符串字面值), 87
 - and string literal (和字符串字面值), 81, 140
 - append, 342
 - are case sensitive (大小写敏感), 344
 - as sequential container (作为顺序容器), 335
 - assign, 340
 - assignment (=) (赋值), 86
 - c_str, 140
 - c_str, example (c_str, 例子), 294
 - capacity, 336
 - compare, 347
 - compared to C-style string (与 C 风格字符串比较), 134
 - compound assignment (复合赋值), 86
 - concatenation (连接), 86
 - constructor (构造函数), 80, 338–339
 - default constructor (默认构造函数), 52
 - empty, 83
 - equality (==) (相等), 85
 - equality operator (相等操作符), 347
 - erase, 340
 - find, 344
 - find_first_not_of, 346
 - find_first_of, 345
 - find_last_not_of, 346
 - find_last_of, 346
 - getline, 82
 - getline, example (getline, 例子), 300
 - header (头文件), 80
 - input operation as condition (输入操作作为条件), 82

- input operator (输入操作符), 81
 - insert, 340
 - output operator (输出操作符), 81
 - random-access iterator (随机访问迭代器), 417
 - relational operator (关系操作符), 85, 347
 - replace, 342
 - reserve, 336
 - rfind, 346
 - size, 83
 - size_type, 84
 - subscript operator (下标操作符), 87
 - substr, 342
 - string literal (字符串字面值), 9, 30, 40
 - addition to string (加到 string), 87
 - and string library type (和 string 库类型), 81, 140
 - and C-style string (和 C 风格字符串), 140
 - concatenation (连接), 41
 - stringstream, 285, 299-301, 302
 - 另见 istream
 - 另见 ostream
 - str, 301
 - strlen, 133
 - strncat, 133
 - strncpy, 133
 - struct (结构), 另见 class
 - default access label (默认访问标号), 433
 - default inheritance access label (默认继承访问标号), 574
 - struct, keyword (struct 关键字), 66, 76, 474
 - in variable definition (在变量定义中), 440
 - structure, data (数据结构), 20, 28
 - Studio, Visual, 4
 - subscript ([]) (下标), 87, 108, 146, 389
 - bitset, 105
 - deque, 325
 - map, 363
 - string, 87
 - vector, 94, 325
 - and multi-dimensioned array (和多维数组), 142
 - and pointer (和指针), 124
 - array (数组), 113
 - overloaded operator (重载操作符), 522
 - reference return (引用返回), 522
 - valid subscript range (有效下标范围), 88
 - yields lvalue (生成左值), 88
 - subscript range (下标范围)
 - string, 88
 - vector, 96
 - array, 114
 - substr, string, 342
 - subtraction (-) (减), 150
 - iterator (迭代器), 101, 313
 - pointer (指针), 123
 - swap, 329, 816
 - container (容器), 329
 - swap program (swap 程序), 233, 245
 - swap_ranges, 816
 - switch statement (switch 语句), 199, 224
 - default label (default 标号), 203
 - and break (和 break), 201-203
 - case label (case 标号), 201
 - compared to if (与 if 比较), 199
 - execution flow (执行流), 201
 - expression (表达式), 203
 - variable definition (变量定义), 204
 - synthesized assignment (=) (合成的赋值), 483, 503
 - multiple inheritance (多重继承), 737
 - pointer members (指针成员), 493
 - synthesized copy constructor (合成的复制构造函数), 479, 503
 - multiple inheritance (多重继承), 737
 - pointer members (指针成员), 493
 - virtual base class (虚基类), 747
 - synthesized copy control, volatile (合成的复制控制), 800
 - synthesized default constructor (合成的默认构造函数), 264, 281, 459, 474
 - inheritance (继承), 581
 - synthesized destructor (合成的析构函数), 485, 486
 - multiple inheritance (多重继承), 737
 - pointer members (指针成员), 493
- ## T
- \t(tab character) (制表符), 40
 - table of library name and header (库名与头文件表), 810
 - template (模板)
 - 另见 class template
 - 另见 function template, instantiation
 - class (类), 90, 107
 - class member (类成员), 见 member template
 - link time errors (链接时错误), 635
 - overview (概述), 624
 - template keyword (template 关键字), 625
 - template argument (模板实参), 625, 684
 - and function argument type checking (和函数实参类型检查), 638
 - class template (类模板), 628
 - conversion (转换), 638
 - deduction (推断), 684
 - from function pointer (从函数指针), 640
 - deduction for class template member function (类模板成员函数的推断), 653
 - deduction for function template (函数模板的推断), 637

- explicit and class template (显式的和类模板), 636
- explicit and function template (显式的和函数模板), 642
 - and function pointer (和函数指针), 643
 - specifying (指定), 642
- pointer (指针), 671
- template argument deduction (模板实参推断), 637
- template class (模板类), 见 class template
- template function (模板函数), 见 function template
- template parameter (模板形参), 625, 628–633, 684
 - and member templates (和成员模板), 663
 - name (名字), 628
 - restrictions on use (使用限制), 629
 - nontype parameter (非类型形参), 625, 628, 632, 684
 - class template (类模板), 655
 - must be constant expression (必须是常量表达式), 633
 - scope (作用域), 629
 - type parameter (类型形参), 625, 628, 630, 684
 - uses of inside class definition (类定义内部的使用), 649
- template parameter list (模板形参列表), 625, 684
- template specialization (模板特化), 672, 684
 - class member declaration (类成员声明), 677
 - compared to overloaded function (与重载函数比较), 673
 - definition (定义), 675
 - example (例子), 672
 - function declaration (函数声明), 672, 673
 - member defined outside class body (类定义体外部定义的成员), 676
 - member of class template (类模板的成员), 677
 - parameter matching (形参匹配), 673
 - partial specialization (部分特化), 678, 684
 - scope (作用域), 674
- template<>, 见 template specialization
- temporary object (临时对象), 247
- terminate, 219, 219, 224, 692, 750
- TextQuery
 - class definition (类定义), 382
 - main program using (main 程序使用), 383
 - program design (程序设计), 380
 - program interface (程序接口), 381
 - revisited (再谈), 609
- this pointer (this 指针)
 - implicit (隐式的), 260, 281
 - implicit parameter (隐式形参), 431, 440
 - in overloaded operator (在重载操作符中), 483
 - overloaded operator (重载操作符), 508
 - static member functions (static 成员函数), 469
- three, rule of (三法则), 485, 503
- throw, 216, 216, 224, 689, 750
 - example (例子), 217, 290
 - execution flow (执行流), 218, 691
 - pointer to local object (局部对象的指针), 690
 - rethrow (重新抛出), 695
- tolower, 88
- top
 - priority_queue, 352
 - stack, 351
- toupper, 88
- transform, 815
- transformation program, word (单词转换程序), 370
- translation unit (转换单元), 见 source file
- trunc (file mode) (文件模式), 296
- try block try 块, 216, 217, 224, 750
- try keyword (try 关键字), 217
- type (类型)
 - abstract data (抽象数据), 78, 473
 - arithmetic (算术), 34, 73
 - built-in (内置), 3, 28, 34–37
 - class (类), 20, 28, 65
 - compound (复合), 58, 73, 145
 - dynamic (动态), 568, 622
 - function return (函数返回), 281
 - incomplete (不完全的), 438, 474
 - integral (整型), 34, 75
 - library (库), 29
 - nested (嵌套), 见 nested class, 786
 - return (返回), 3, 29, 226, 227
 - static (静态), 568, 622
 - determines name lookup (确定名字查找), 590
 - name lookup and multiple inheritance (名字查找与多重继承), 735
- type checking (类型检查), 44
 - argument (实参), 229
 - with class type conversion (用类类型的转换), 541
- ellipsis parameter (省略形参), 244
- name lookup (名字查找), 269
- reference to array argument (数组实参的引用), 240
- type identification, run-time (运行时类型识别), 772–780, 807
- type specifier (类型说明符), 48, 76
- type template parameter (类型模板形参), 628, 630, 684
 - 另见 template parameter
- type_info, 807
 - header (头文件), 219
 - name member (name 成员), 780
 - no copy or assign (没有复制或赋值), 780
 - operations (操作), 779
 - returned from typeid (从 typeid 返回的), 776
- typedef
 - and pointer (和指针), 129
 - and pointer to member (和指向成员的指针), 783
 - pointer to function (指向函数的指针), 276

typedef, 61, 76
 typeid operator (typeid 操作符), 775, 807
 and virtual functions (和虚函数), 775
 example (例子), 776
 returns type_info (返回 type_info), 776
 typename, keyword (typename 关键字)
 compared to class (与类比较的), 631
 in template parameter (在模板形参中), 630
 inside template definition (在模板定义内部), 632

U

U_Ptr, 496
 unary function object (一元函数对象), 533
 unary minus (-) (一元减), 150
 unary operator (一元操作符), 148, 189
 unary plus (+) (一元加), 150
 uncaught exception (未捕获的异常), 692
 undefined behavior (未定义的行为), 41, 76
 dangling pointer (悬垂指针), 176
 synthesized copy control (合成的复制控制), 494
 invalidated iterator (无效的迭代器), 315
 uninitialized class data member (未初始化的类数据成员), 459
 uninitialized pointer (未初始化的指针), 117
 uninitialized variable (未初始化的变量), 51
 underflow_error, 219
 unexpected, 708, 750
 uninitialized (未初始化的), 8, 30, 51, 76
 uninitialized pointer (未初始化的指针), 117
 uninitialized_copy, 755, 759
 uninitialized_fill, 755
 union (联合), 792, 807
 anonymous (匿名的), 795, 805
 as nested type (作为嵌套类型), 794
 example (例子), 794
 limitations on (限制), 793
 union keyword (union 关键字), 793
 unique, 402, 819
 list, 423
 unique_copy, 412, 820
 unitbuf, manipulator flushes the buffer (unitbuf, 操纵符刷新缓冲区), 291
 unnamed namespace (未命名的命名空间), 718, 750
 local to file (局部于文件), 718
 replace file static (代替文件 static), 719
 unsigned, 35, 76
 conversion to signed (转换到 signed), 36, 180
 literal (numU or numu) (字面值), 39
 unsigned char, 36
 unwinding, stack (栈展开), 691, 750

upper_bound, 814
 associative container (关联容器), 377
 book finding program (书籍查找程序), 378
 example (例子), 607
 uppercase manipulator (uppercase 操纵符), 828
 use count (使用计数), 495, 503
 design overview (设计概述), 495
 generic class (泛型类), 667
 held in companion class (在伙伴类中保存), 496
 pointer to (指针指向), 600
 self-assignment check (自身赋值检查), 498
 user (用户), 433, 563
 using declaration (using 声明), 78, 108, 720, 750
 access control (访问控制), 573
 class member access (类成员访问), 574
 in header (在头文件中), 80
 overloaded function (重载函数), 728
 overloaded inherited functions (重载继承函数), 593
 scope (作用域), 720
 using directive (using 指示), 721, 751
 overloaded function (重载函数), 729
 pitfalls (缺陷), 724
 scope (作用域), 721
 example (例子), 722
 name collisions (名字冲突), 723
 utility header (utility 头文件), 356

V

value initialization (值初始化), 92, 108
 map subscript operator (map 下标操作符), 363
 vector, 92
 and dynamically allocated array (和动态分配的数组), 136
 deque deque (双端队列 deque), 309
 list list (列表 list), 309
 of dynamically allocated object (动态分配对象的), 175
 and resize (和 resize), 324
 sequential container (顺序容器), 309
 vector vector (向量 vector), 309
 value semantics (值语义), 499, 503
 value_type, 389
 map, multimap, 361
 sequential container (顺序容器), 317
 set, multiset, 372
 varargs, 244
 variable (变量), 8, 30, 43—55
 define before use (在使用之前定义), 44
 defined after case label (在 case 标号之后定义), 204
 definition (定义), 48
 definitions and goto (定义与 goto), 215
 initialization (初始化), 48, 50, 76

- constructor (构造函数), 452
- local (局部的), 227, 281
- scope (作用域), 55
- Vector, 757
 - capacity, 757
 - memory allocation strategy (内存分配策略), 757
 - push_back, 758
 - reallocate, 758
 - size, 757
 - using operator new and delete (使用 operator new 和 delete), 761
 - using explicit destructor call (使用显式析构函数调用), 763
 - using placement new (使用定位 new), 762
- vector, 90–95, 354
 - argument (实参), 237
 - as element type (作为元素类型), 311
 - assign, 328
 - assignment (=) (赋值), 328
 - at, 325
 - back, 324
 - begin, 97, 317
 - capacity, 331
 - clear, 327
 - const_iterator, 99, 316
 - const_reference, 317
 - const_reverse_iterator, 316
 - constructor from element count, uses copy constructor (来自元素计数的构造函数, 使用复制构造函数), 478
 - constructor taking iterators (接受迭代器的构造函数), 140
 - constructors (构造函数), 91–92, 307–309
 - difference_type, 316
 - element type constraints (元素类型约束), 309, 323
 - empty, 93, 323
 - end, 97, 317
 - erase, 326, 402
 - invalidates iterator (使迭代器失效), 326
 - front, 324
 - header (头文件), 90
 - initialization from pointer (用指针初始化), 140
 - insert, 319
 - invalidates iterator (使迭代器失效), 320
 - iterator, 97, 316
 - iterator supports arithmetic (迭代器支持运算), 312
 - memory allocation strategy (内存分配策略), 756
 - memory management strategy (内存管理策略), 330
 - parameter (形参), 237
 - performance characteristics (性能特征), 334
 - pop_back, 326
 - push_back, 94, 318
 - invalidates iterator (使迭代器失效), 321
 - random-access iterator (随机访问迭代器), 417
 - rbegin, 317, 412
 - reference, 317
 - relational operators (关系操作符), 321
 - rend, 317, 412
 - reserve, 331
 - resize, 323
 - reverse_iterator, 316, 412
 - size, 93, 323
 - size_type, 93, 316
 - subscript ([]) (下标), 325
 - subscript operator (下标操作符), 94
 - supports relational operators (支持关系操作符), 313
 - swap, 329
 - type (类型), 91
 - types defined by (定义的类型), 316
 - value_type, 317
- vector capacity program (vector 容量程序), 331
- viable function (可行函数), 270, 282
 - with class type conversion (带类类型转换的), 545
- virtual base class (虚基类), 741, 751
 - ambiguities (二义性), 743
 - conversion (转换), 743
 - defining base as (定义基类为), 742
 - derived class constructor (派生类构造函数), 744
 - name lookup (名字查找), 743
 - order of construction (构造顺序), 746
 - stream types (流类型), 741
- virtual function (虚函数), 559, 566–570, 622
 - assignment operator (赋值操作符), 588
 - calls resolved at run time (运行时确定的调用), 568
 - compared to run-time type identification (与运行时类型识别比较), 777
 - default argument (默认实参), 570
 - derived classes (派生类), 564
 - destructor (析构函数), 587
 - multiple inheritance (多重继承), 736
 - exception specifications (异常说明), 710
 - in constructors (在构造函数中), 589
 - in destructor (在析构函数中), 589
 - introduction (介绍), 561
 - multiple inheritance (多重继承), 735
 - no virtual constructor (非虚构造函数), 588
 - overloaded (重载), 593
 - overloaded operator (重载操作符), 615
 - overriding run-time binding (覆盖运行时绑定), 570
 - pure (纯虚函数), 596, 622
 - example (例子), 609
 - return type (返回类型), 564
 - run-time type identification (运行时类型识别), 772

scope (作用域), 594
static, 469
to copy unknown type (复制未知类型), 602
type-sensitive equality (区分敏感的相等), 778
virtual inheritance (虚继承), 741, 751
virtual keyword (virtual 关键字), 559
Visual Studio, 4
void, 34, 76
 return type (返回类型), 245
void*, 119, 146
 const void*, 127, 145
volatile, 800, 807
 pointer (指针), 800
 synthesized copy control (合成的复制控制), 800
vowel counting program (元音计数程序), 200

W

wcerr, 286
wchar_t, 34
 literal (字面值), 40
wchar_t streams (wchar_t 流), 286
wcin, 286
wcout, 286
weak ordering, strict (严格弱排序), 360, 389
wfstream, 286
what, 见 exception,
while statement (while 语句), 12, 30, 204, 224
 condition in (其中的条件), 205
whitespace (空白字符), 81
wide character streams (宽字符流), 286

wfstream, 286
window, console (控制窗口), 6
Window_Mgr, 465
wiostream, 286
wistream, 286
wistreamstream, 286
wofstream, 286
word (字), 35, 76
word count program (单词计数程序), 363
 restricted (受限的), 374
word per line processing (每行处理的单词)
 istreamstream, 386
 istreamstream, 370
istreamstream, 300
word transformation program (单词转换程序), 370
WordQuery, 609
 definition (定义), 616
wostream, 286
wostringstream, 286
wrap around, 38
wstringstream, 286

X

\xnnn(hexadecimal escape sequence) (十六进制转义序列), 40

Z

ZooAnimal, using virtual inheritance (ZooAnimal, 使用虚继承), 741
ZooAnimal class hierarchy (ZooAnimal 类层次), 731

[G e n e r a l I n f o r m a t i o n]

书名= c + + p r i m e r 中文版

作者= S t a n l e y B . L i p p m a n , J o s e e L a j o i e , B a r b a r a E . M o o 著 李师贤 [等] 译

S S 号= 1 1 6 1 6 8 2 4

页数= 7 4 5

丛书名= 图灵程序设计丛书

读秀号= 0 0 0 0 0 6 0 1 9 0 8 9

I S B N 号= 7 - 1 1 5 - 1 4 5 5 4 - 7

主题词= c 语言 (学科： 程序设计) c 语言 程序设计

参考文献格式= s t a n l e y b . l i p p m a n , j o s e e l a j o i e , b a r b a r a e . m o o 著 李师贤 [等] 译 . c + + p r i m e r 中文版 . 人民邮电出版社 2 0 0 6 .

封面
书名
版权
前言
目录

第1章 快速入门

- 1.1 编写简单的C++程序
- 1.2 初窥输入/输出
 - 1.2.1 标准输入与输出对象
 - 1.2.2 一个使用IO库的程序
- 1.3 关于注释
- 1.4 控制结构
 - 1.4.1 while语句
 - 1.4.2 for语句
 - 1.4.3 if语句
 - 1.4.4 读入未知数目的输入
- 1.5 类的简介
 - 1.5.1 Sales_item类
 - 1.5.2 初窥成员函数
- 1.6 C++程序

小结
术语

第一部分 基本语言

第2章 变量和基本类型

- 2.1 基本内置类型
 - 2.1.1 整型
 - 2.1.2 浮点型
- 2.2 字面值常量
- 2.3 变量
 - 2.3.1 什么是变量
 - 2.3.2 变量名
 - 2.3.3 定义对象
 - 2.3.4 变量初始化规则
 - 2.3.5 声明和定义
 - 2.3.6 名字的作用域
 - 2.3.7 在变量使用处定义变量
- 2.4 const限定符
- 2.5 引用
- 2.6 typedef名字
- 2.7 枚举
- 2.8 类类型
- 2.9 编写自己的头文件
 - 2.9.1 设计自己的头文件
 - 2.9.2 预处理器的简单介绍

小结
术语

第3章 标准库类型

- 3.1 命名空间的using声明
- 3.2 标准库string类型
 - 3.2.1 string对象的定义和初始化
 - 3.2.2 String对象的读写

- 3 . 2 . 3 s t r i n g 对象的操作
- 3 . 2 . 4 s t r i n g 对象中字符的处理
- 3 . 3 标准库 v e c t o r 类型
- 3 . 3 . 1 v e c t o r 对象的定义和初始化
- 3 . 3 . 2 v e c t o r 对象的操作
- 3 . 4 迭代器简介
- 3 . 5 标准库 b i t s e t 类型
- 3 . 5 . 1 b i t s e t 的定义和初始化
- 3 . 5 . 2 b i t s e t 对象上的操作

小结

术语

第4章 数组和指针

- 4 . 1 数组
- 4 . 1 . 1 数组的定义和初始化
- 4 . 1 . 2 数组操作
- 4 . 2 指针的引入
- 4 . 2 . 1 什么是指针
- 4 . 2 . 2 指针的定义和初始化
- 4 . 2 . 3 指针操作
- 4 . 2 . 4 使用指针访问数组元素
- 4 . 2 . 5 指针和 c o n s t 限定符
- 4 . 3 C 风格字符串
- 4 . 3 . 1 创建动态数组
- 4 . 3 . 2 新旧代码的兼容
- 4 . 4 多维数组

小结

术语

第5章 表达式

- 5 . 1 算术操作符
- 5 . 2 关系操作符和逻辑操作符
- 5 . 3 位操作符
- 5 . 3 . 1 b i t s e t 对象或整型值的使用
- 5 . 3 . 2 将移位操作符用于 I O
- 5 . 4 赋值操作符
- 5 . 4 . 1 赋值操作的右结合性
- 5 . 4 . 2 赋值操作具有低优先级
- 5 . 4 . 3 复合赋值操作符
- 5 . 5 自增和自减操作符
- 5 . 6 箭头操作符
- 5 . 7 条件操作符
- 5 . 8 s i z e o f 操作符
- 5 . 9 逗号操作符
- 5 . 1 0 复合表达式的求值
- 5 . 1 0 . 1 优先级
- 5 . 1 0 . 2 结合性
- 5 . 1 0 . 3 求值顺序
- 5 . 1 1 n e w 和 d e l e t e 表达式
- 5 . 1 2 类型转换
- 5 . 1 2 . 1 何时发生隐式类型转换
- 5 . 1 2 . 2 算术转换
- 5 . 1 2 . 3 其他隐式转换

- 5 . 1 2 . 4 显式转换
- 5 . 1 2 . 5 何时需要强制类型转换
- 5 . 1 2 . 6 命名的强制类型转换
- 5 . 1 2 . 7 旧式强制类型转换

小结

术语

第 6 章 语句

- 6 . 1 简单语句
- 6 . 2 声明语句
- 6 . 3 复合语句（语句块）
- 6 . 4 语句作用域
- 6 . 5 i f 语句
- 6 . 6 s w i t c h 语句
 - 6 . 6 . 1 使用 s w i t c h
 - 6 . 6 . 2 s w i t c h 中的控制流
 - 6 . 6 . 3 d e f a u l t 标号
 - 6 . 6 . 4 s w i t c h 表达式与 c a s e 标号
 - 6 . 6 . 5 s w i t c h 内部的变量定义
- 6 . 7 w h i l e 语句
- 6 . 8 f o r 循环语句
 - 6 . 8 . 1 省略 f o r 语句头的某些部分
 - 6 . 8 . 2 f o r 语句头中的多个定义
- 6 . 9 d o w h i l e 语句
- 6 . 1 0 b r e a k 语句
- 6 . 1 1 c o n t i n u e 语句
- 6 . 1 2 g o t o 语句
- 6 . 1 3 t r y 块和异常处理
 - 6 . 1 3 . 1 t h r o w 表达式
 - 6 . 1 3 . 2 t r y 块
 - 6 . 1 3 . 3 标准异常
- 6 . 1 4 使用预处理器进行调试

小结

术语

第 7 章 函数

- 7 . 1 函数的定义
 - 7 . 1 . 1 函数返回类型
 - 7 . 1 . 2 函数形参表
- 7 . 2 参数传递
 - 7 . 2 . 1 非引用形参
 - 7 . 2 . 2 引用形参
 - 7 . 2 . 3 v e c t o r 和其他容器类型的形参
 - 7 . 2 . 4 数组形参
 - 7 . 2 . 5 传递给函数的数组的处理
 - 7 . 2 . 6 m a i n ：处理命令行选项
 - 7 . 2 . 7 含有可变形参的函数
- 7 . 3 r e t u r n 语句
 - 7 . 3 . 1 没有返回值的函数
 - 7 . 3 . 2 具有返回值的函数
 - 7 . 3 . 3 递归
- 7 . 4 函数声明
- 7 . 5 局部对象

- 7 . 5 . 1 自动对象
- 7 . 5 . 2 静态局部对象
- 7 . 6 内联函数
- 7 . 7 类的成员函数
- 7 . 7 . 1 定义成员函数的函数体
- 7 . 7 . 2 在类外定义成员函数
- 7 . 7 . 3 编写 S a l e s _ i t e m 类的构造

函数

- 7 . 7 . 4 类代码文件的组织
- 7 . 8 重载函数
- 7 . 8 . 1 重载与作用域
- 7 . 8 . 2 函数匹配与实参转换
- 7 . 8 . 3 重载确定的三个步骤
- 7 . 8 . 4 实参类型转换
- 7 . 9 指向函数的指针

小结

术语

第 8 章 标准 I O 库

- 8 . 1 面向对象的标准库
- 8 . 2 条件状态
- 8 . 3 输出缓冲区的管理
- 8 . 4 文件的输入和输出
- 8 . 4 . 1 文件流对象的使用
- 8 . 4 . 2 文件模式
- 8 . 4 . 3 一个打开并检查输入文件的

程序

- 8 . 5 字符串流

小结

术语

第二部分 容器和算法

第 9 章 顺序容器

- 9 . 1 顺序容器的定义
- 9 . 1 . 1 容器元素的初始化
- 9 . 1 . 2 容器内元素的类型约束
- 9 . 2 迭代器和迭代器范围
- 9 . 2 . 1 迭代器范围
- 9 . 2 . 2 使迭代器失效的容器操作
- 9 . 3 顺序容器的操作
- 9 . 3 . 1 容器定义的类型别名
- 9 . 3 . 2 b e g i n 和 e n d 成员
- 9 . 3 . 3 在顺序容器中添加元素
- 9 . 3 . 4 关系操作符
- 9 . 3 . 5 容器大小的操作
- 9 . 3 . 6 访问元素
- 9 . 3 . 7 删除元素
- 9 . 3 . 8 赋值与 s w a p
- 9 . 4 v e c t o r 容器的自增长
- 9 . 5 容器的选用
- 9 . 6 再谈 s t r i n g 类型
- 9 . 6 . 1 构造 s t r i n g 对象的其他方法
- 9 . 6 . 2 修改 s t r i n g 对象的其他方法

- 9.6.3 只适用于 `string` 类型的操作
- 9.6.4 `string` 类型的查找操作
- 9.6.5 `string` 对象的比较
- 9.7 容器适配器
- 9.7.1 栈适配器
- 9.7.2 队列和优先级队列

小结

术语

第10章 关联容器

- 10.1 引言：`pair` 类型
- 10.2 关联容器
- 10.3 `map` 类型
 - 10.3.1 `map` 对象的定义
 - 10.3.2 `map` 定义的类型
 - 10.3.3 给 `map` 添加元素
 - 10.3.4 使用下标访问 `map` 对象
 - 10.3.5 `map::insert` 的使用
 - 10.3.6 查找并读取 `map` 中的元素
 - 10.3.7 从 `map` 对象中删除元素
 - 10.3.8 `map` 对象的迭代遍历
 - 10.3.9 “单词转换” `map` 对象
- 10.4 `set` 类型
 - 10.4.1 `set` 容器的定义和使用
 - 10.4.2 创建“单词排除”集
- 10.5 `multimap` 和 `multiset` 类型
 - 10.5.1 元素添加和删除
 - 10.5.2 在 `multimap` 和 `multiset` 中查找元素
- 10.6 容器的综合应用：文本查询程序
 - 10.6.1 查询程序的设计
 - 10.6.2 `TextQuery` 类
 - 10.6.3 `TextQuery` 类的使用
 - 10.6.4 编写成员函数

小结

术语

第11章 泛型算法

- 11.1 概述
- 11.2 初窥算法
 - 11.2.1 只读算法
 - 11.2.2 写容器元素的算法
 - 11.2.3 对容器元素重新排序的算法
- 11.3 迭代器回顾
 - 11.3.1 插入迭代器
 - 11.3.2 `istream` 迭代器
 - 11.3.3 反向迭代器
 - 11.3.4 `const` 迭代器
 - 11.3.5 五种迭代器
- 11.4 泛型算法的结构
 - 11.4.1 算法的形参模式
 - 11.4.2 算法的命名规范
- 11.5 容器特有的算法

小结

术语

第三部分 类和数据抽象

第 1 2 章 类

- 1 2 . 1 类的定义和声明
 - 1 2 . 1 . 1 类定义：扼要重述
 - 1 2 . 1 . 2 数据抽象和封装
 - 1 2 . 1 . 3 关于类定义的更多内容
 - 1 2 . 1 . 4 类声明与类定义
 - 1 2 . 1 . 5 类对象
- 1 2 . 2 隐含的 t h i s 指针
- 1 2 . 3 类作用域

类作用域中的名字查找

- 1 2 . 4 构造函数
 - 1 2 . 4 . 1 构造函数初始化式
 - 1 2 . 4 . 2 默认实参与构造函数
 - 1 2 . 4 . 3 默认构造函数
 - 1 2 . 4 . 4 隐式类—类型转换
 - 1 2 . 4 . 5 类成员的显式初始化
- 1 2 . 5 友元
- 1 2 . 6 s t a t i c 类成员
 - 1 2 . 6 . 1 s t a t i c 成员函数
 - 1 2 . 6 . 2 s t a t i c 数据成员

小结

术语

第 1 3 章 复制控制

- 1 3 . 1 复制构造函数
 - 1 3 . 1 . 1 合成的复制构造函数
 - 1 3 . 1 . 2 定义自己的复制构造函数
 - 1 3 . 1 . 3 禁止复制
- 1 3 . 2 赋值操作符
- 1 3 . 3 析构函数
- 1 3 . 4 消息处理示例
- 1 3 . 5 管理指针成员
 - 1 3 . 5 . 1 定义智能指针类
 - 1 3 . 5 . 2 定义值型类

小结

术语

第 1 4 章 重载操作符与转换

- 1 4 . 1 重载操作符的定义
- 1 4 . 2 输入和输出操作符
 - 1 4 . 2 . 1 输出操作符 < < 的重载
 - 1 4 . 2 . 2 输入操作符 > > 的重载
- 1 4 . 3 算术操作符和关系操作符
 - 1 4 . 3 . 1 相等操作符
 - 1 4 . 3 . 2 关系操作符
- 1 4 . 4 赋值操作符
- 1 4 . 5 下标操作符
- 1 4 . 6 成员访问操作符
- 1 4 . 7 自增操作符和自减操作符
- 1 4 . 8 调用操作符和函数对象
 - 1 4 . 8 . 1 将函数对象用于标准库算法

1 4 . 8 . 2	标准库定义的函数对象
1 4 . 8 . 3	函数对象的函数适配器
1 4 . 9	转换与类类型
1 4 . 9 . 1	转换为什么有用
1 4 . 9 . 2	转换操作符
1 4 . 9 . 3	实参匹配和转换
1 4 . 9 . 4	重载确定和类的实参
1 4 . 9 . 5	重载、转换和操作符
小结	
术语	
第四部分	面向对象编程与泛型编程
第 1 5 章	面向对象编程
1 5 . 1	面向对象编程：概述
1 5 . 2	定义基类和派生类
1 5 . 2 . 1	定义基类
1 5 . 2 . 2	p r o t e c t e d 成员
1 5 . 2 . 3	派生类
1 5 . 2 . 4	v i r t u a l 与其他成员函数
1 5 . 2 . 5	公用、私有和受保护的继承
1 5 . 2 . 6	友元关系与继承
1 5 . 2 . 7	继承与静态成员
1 5 . 3	转换与继承
1 5 . 3 . 1	派生类到基类的转换
1 5 . 3 . 2	基类到派生类的转换
1 5 . 4	构造函数和复制控制
1 5 . 4 . 1	基类构造函数和复制控制
1 5 . 4 . 2	派生类构造函数
1 5 . 4 . 3	复制控制和继承
1 5 . 4 . 4	虚析构函数
1 5 . 4 . 5	构造函数和析构函数中的虚函数
1 5 . 5	继承情况下的类作用域
1 5 . 5 . 1	名字查找在编译时发生
1 5 . 5 . 2	名字冲突与继承
1 5 . 5 . 3	作用域与成员函数
1 5 . 5 . 4	虚函数与作用域
1 5 . 6	纯虚函数
1 5 . 7	容器与继承
1 5 . 8	句柄类与继承
1 5 . 8 . 1	指针型句柄
1 5 . 8 . 2	复制未知类型
1 5 . 8 . 3	使用句柄
1 5 . 9	再谈文本查询示例
1 5 . 9 . 1	面向对象的解决方案
1 5 . 9 . 2	值型句柄
1 5 . 9 . 3	Q u e r y _ b a s e 类
1 5 . 9 . 4	Q u e r y 句柄类
1 5 . 9 . 5	派生类
1 5 . 9 . 6	e v a l 函数
小结	
术语	
第 1 6 章	模板与泛型编程

1 6 . 1	模板定义
1 6 . 1 . 1	定义函数模板
1 6 . 1 . 2	定义类模板
1 6 . 1 . 3	模板形参
1 6 . 1 . 4	模板类型形参
1 6 . 1 . 5	非类型模板形参
1 6 . 1 . 6	编写泛型程序
1 6 . 2	实例化
1 6 . 2 . 1	模板实参推断
1 6 . 2 . 2	函数模板的显式实参
1 6 . 3	模板编译模型
1 6 . 4	类模板成员
1 6 . 4 . 1	类模板成员函数
1 6 . 4 . 2	非类型形参的模板实参
1 6 . 4 . 3	类模板中的友元声明
1 6 . 4 . 4	Q u e u e 和 Q u e u e I t e m 的友元声明
1 6 . 4 . 5	成员模板
1 6 . 4 . 6	完整的 Q u e u e 类
1 6 . 4 . 7	类模板的 s t a t i c 成员
1 6 . 5	一个泛型句柄类
1 6 . 5 . 1	定义句柄类
1 6 . 5 . 2	使用句柄
1 6 . 6	模板特化
1 6 . 6 . 1	函数模板的特化
1 6 . 6 . 2	类模板的特化
1 6 . 6 . 3	特化成员而不特化类
1 6 . 6 . 4	类模板的部分特化
1 6 . 7	重载与函数模板
小结	
术语	
第五部分	高级主题
第 1 7 章	用于大型程序的工具
1 7 . 1	异常处理
1 7 . 1 . 1	抛出类类型的异常
1 7 . 1 . 2	栈展开
1 7 . 1 . 3	捕获异常
1 7 . 1 . 4	重新抛出
1 7 . 1 . 5	捕获所有异常的处理代码
1 7 . 1 . 6	函数测试块与构造函数
1 7 . 1 . 7	异常类层次
1 7 . 1 . 8	自动资源释放
1 7 . 1 . 9	a u t o _ p t r 类
1 7 . 1 . 1 0	异常说明
1 7 . 1 . 1 1	函数指针的异常说明
1 7 . 2	命名空间
1 7 . 2 . 1	命名空间的定义
1 7 . 2 . 2	嵌套命名空间
1 7 . 2 . 3	未命名的命名空间
1 7 . 2 . 4	命名空间成员的使用
1 7 . 2 . 5	类、命名空间和作用域

- 17.2.6 重载与命名空间
- 17.2.7 命名空间与模板
- 17.3 多重继承与虚继承
- 17.3.1 多重继承
- 17.3.2 转换与多个基类
- 17.3.3 多重继承派生类的复制控制
- 17.3.4 多重继承下的类作用域
- 17.3.5 虚继承
- 17.3.6 虚基类的声明
- 17.3.7 特殊的初始化语义

小结

术语

第18章 特殊工具与技术

- 18.1 优化内存分配
- 18.1.1 C++中的内存分配
- 18.1.2 `allocator`类
- 18.1.3 `operator new`函数和`operator delete`函数
- 18.1.4 定位`new`表达式
- 18.1.5 显式析构函数的调用
- 18.1.6 类特定的`new`和`delete`
- 18.1.7 一个内存分配器基类
- 18.2 运行时类型识别
- 18.2.1 `dynamic_cast`操作符
- 18.2.2 `typeid`操作符
- 18.2.3 RTTI的使用
- 18.2.4 `type_info`类
- 18.3 类成员的指针
- 18.3.1 声明成员指针
- 18.3.2 使用类成员的指针
- 18.4 嵌套类
- 18.4.1 嵌套类的实现
- 18.4.2 嵌套类作用域中的名字查找
- 18.5 联合：节省空间的类
- 18.6 局部类
- 18.7 固有的不可移植的特征
- 18.7.1 位域
- 18.7.2 `volatile`限定符
- 18.7.3 链接指示：`extern "C"`

小结

术语

附录 标准库

索引