# 第6章函数

## 6.2 参数传递

### 6.2.4 数组形参

需求：因为数组是以指针的形式传递给数组的，所以函数一开始不知道数组的确切尺寸，调用者应该提供一些额外的信息，管理指针形参有三种常用技术。

1. 使用编辑指定数组长度
2. 使用标准库规范

// function

void print(const int \*beg, const int \*end)

{

// print every element starting at beg up to but not including end

while (beg != end)

cout << \*beg++ << endl; // print the current element

// and advance the pointer

}

//call

int j[2] = {0, 1};

// j is converted to a pointer to the first element in j

// the second argument is a pointer to one past the end of j

print(begin(j), end(j));

1. 显示传递一个表示数组大小的形参

// const int ia[] is equivalent to const int\* ia

// size is passed explicitly and used to control access to elements of ia

void print(const int ia[], size\_t size)

{

for (size\_t i = 0; i != size; ++i)

{

cout << ia[i] << endl;

}

}

int j[] = { 0, 1 }; // int array of size 2

print(j, end(j) - begin(j));

**数组形参和const**

When a function does not need write access to the array elements, the array parameter should be a pointer to const (§ 2.4.2, p. 62). A parameter should be a plain pointer to a nonconst type only if the function needs to change element values.

**数组引用形参**

Note

The parentheses around &arr are necessary (§ 3.5.1, p. 114):

Click here to view code image

f(int &arr[10]) // error: declares arr as an array of references

C++ Primer, Fifth Edition

f(int (&arr)[10]) // ok: arr is a reference to an array of ten ints