CS 1502H 程序设计思想与方法 (C++)
Principles and Methodology in Programming
Chapter 6. 函数-1

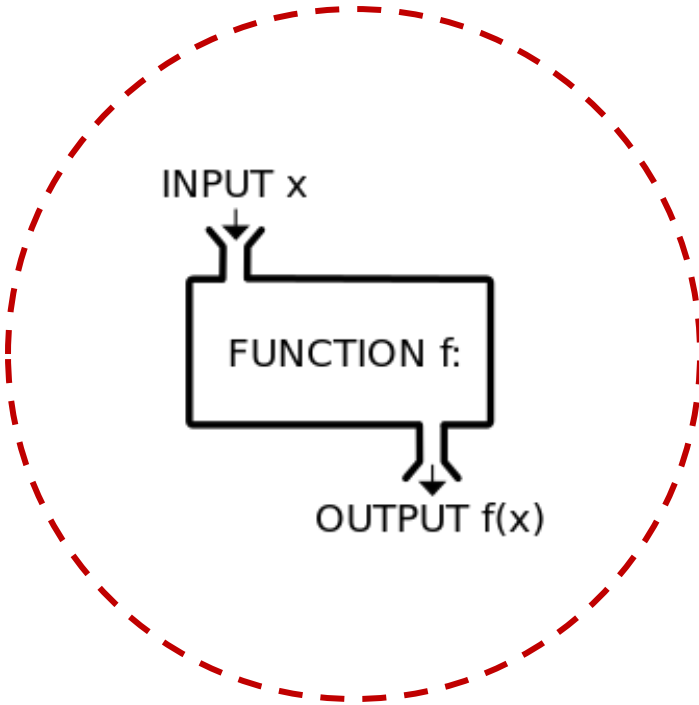
陈东尧
上海交通大学
2025年秋季

本章大纲

- 函数的用途和定义
- 函数的使用
- 变量的作用域与存储类别
- 函数参数—数组和默认值
- 内联、重载、模版、递归

什么是函数（function）

- 数学中的函数（function）：
  - 在数学中为两不为空集的集合间的一种对应关系：输入值集合中的每项元素皆能对应**唯一**一项输出值集合中的元素。



数学中的函数

- 程序设计中，函数（function）是子程序中的一种
  - 子程序：是一个大型程序中的某部分代码，由一个或多个语句块组成。它负责完成某项特定任务，而且相较于其他代码，具备相对的独立性。

函数的概述

- 函数是程序设计语言中最重要的部分之一，是模块化设计的主要工具。每一个C++程序都要用到函数。
- 函数
  - 输入、输出

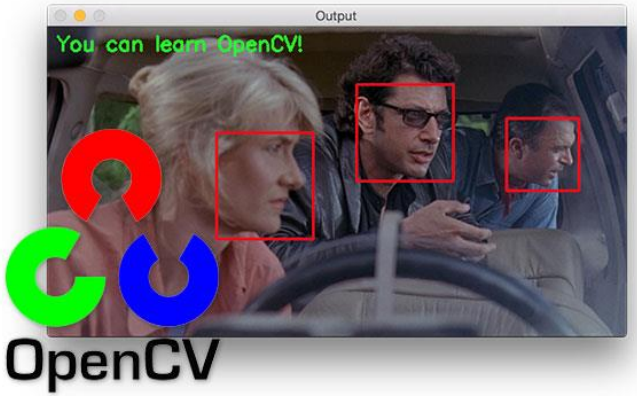
- 在每一个完整的C++程序中都必须有一个main() 函数。

函数编程的好处

- 函数的使用者，无需知道函数内部如何运作，只了解其与外界的接口（Interface）即可。
- 把函数内的具体实现细节对外界隐藏起来，只要对外接口不变，就不影响函数的使用，便于实现函数的复用和模块化编程

函数的分类

- 标准库函数
  - ANSI/ISO[1] C++定义的标准库函数
    - 使用时，必须在程序开头把定义该函数的头文件包含进来（例如：iostream, cstring, 等等）
  - 第三方库函数（third-party library）
    - 不在标准范围内，能扩充C/C++语言的功能，由其他厂商自行开发的C++语言函数库



车道线偏离检测

- 自定义函数
  - 用户自己定义的函数
    - 包装后，也可成为函数库，供别人使用

[1] ANSI stands for the American National Standards Institute, and ISO stands for the International Organization for Standardization

函数的定义

- 函数定义

```
类型名  函数名（形式参数表） //函数头
{
 变量定义部分 }        //函数体
 语句部分     }
}
```

- 函数的返回值：返回值类型应与定义中的类型名一致

```
return 返回值; 或  return（返回值）;
```

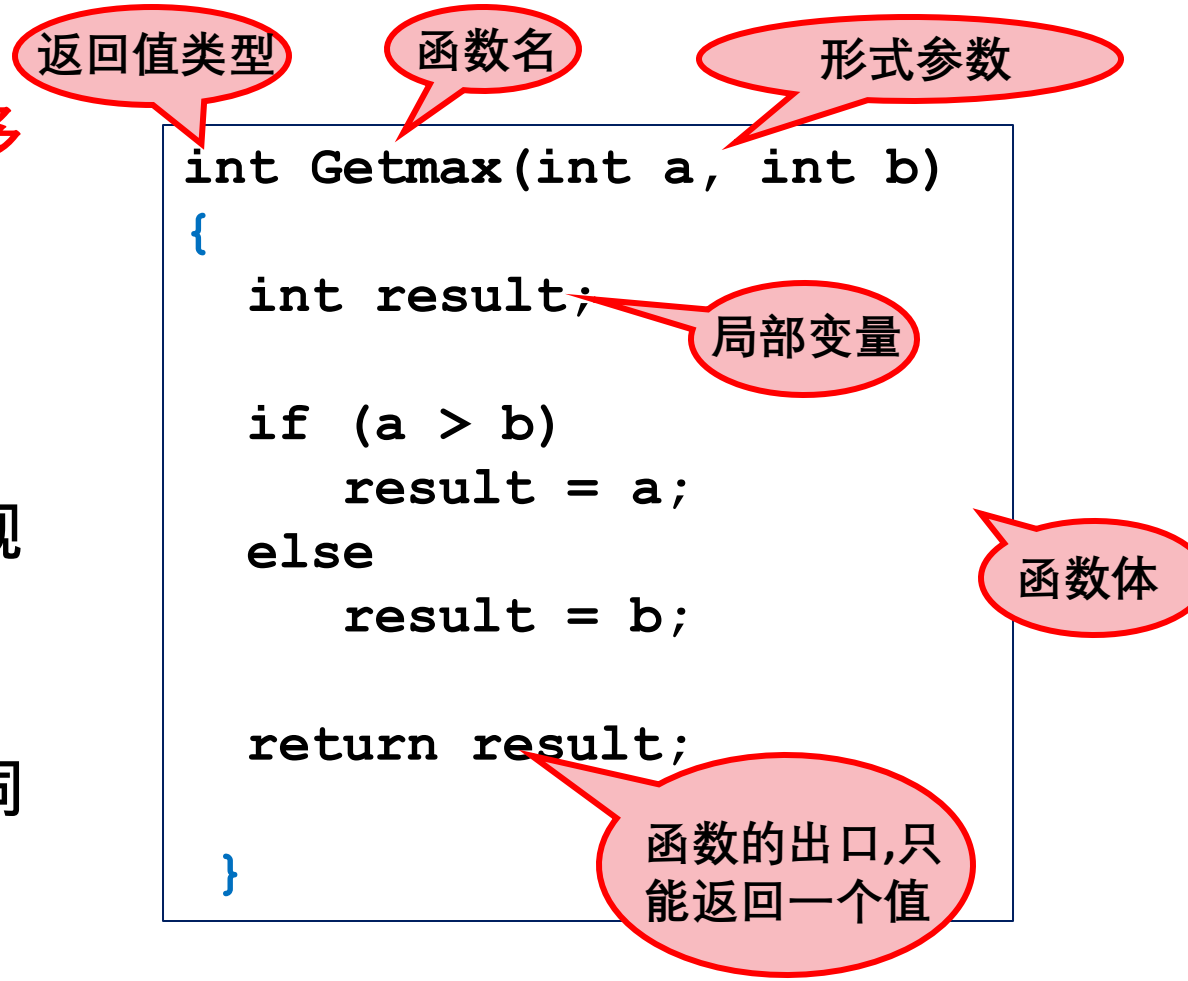
```
int max(int a, int b){
    if (a>b) return(a);
        else return(b);
}
```

- 表示一个函数没有返回值，类型标识符用void。没有返回值的函数也称为过程（void function）。


Void means completely empty

函数的定义规则

- C++的函数只能有一个返回值，可以有多条返回语句
- C++的返回值不能是数组
- 函数名是一个标识符，符合标识符命名规范。
- 函数名要有意义，函数名一般是一个动词短语，表示函数的行为



函数举例：无参数，无返回值

- 打印一个有五行的三角形

```
    *
   ***
  *****
 *******
*********
```

```
void  PrintStar( )
{
    cout << "    *\n";
    cout << "   ***\n";
    cout << "  *****\n";
    cout << " *******\n";
    cout << "*********\n";
}
```

函数举例：有参数，无返回值

- 打印一个由**n行**组成的三角形

```
void PrintStarN (int numOfLine)
{  int i , j;

   for (i = 1; i <= numOfLine; ++i)    {
        cout << endl;
        for (j = 1; j <= numOfLine - i; ++j)  cout << ' ';
        for (j = 1; j <= 2 * i - 1; ++j)    cout << "*";
   }
}
```

函数举例：无参数，有返回值

- 从终端获取一个1 – 10之间的整型数，即：若输入满足1-10之间，则输出

```
int getInput( )
{
    int num;
    while (true) {
          cin >> num;
         if (num >= 1 && num <= 10) return num;
    }
}
```

函数举例：有参数，有返回值

- 计算n！

```
int fact (int n)
{
    int s=1, i;
    if (n < 0){
        return(0);
    }
    for (i = 1; i <= n; ++i){
        s *= i;
    }
    return(s);
}
```

本章大纲

- 函数的用途和定义
- 函数的使用
- 变量的作用域与存储类别
- 函数参数—数组和默认值
- 内联、重载、模版、递归

函数的声明（declaration of a function）

- 所有函数在使用前必须被声明，以便让编译器知道用户的用法是否正确
- 函数声明包括三个内容(与函数头相似)：
  - 函数名
  - 参数的个数和类型
  - 函数的返回类型
- 函数的声明被称为函数的原型(prototype)，它的形式为：

  返回类型 函数名（参数表）；

  参数表中的每个参数说明可以是类型，也可以是类型后面再接一个参数名。如：

```
int  max(int, int);
int  max(int a, int b);
```

函数说明规则

- 库函数（library）在调用前需要 # include相应的头文件
- 自定义的函数在调用时需要进行函数原型说明 （define the prototype）
- 函数原型说明与函数头写法上需要保持完全一致，即函数类型、函数名、参数个数和参数顺序必须相同。
- 如果被调函数（callee）的定义在主调函数（caller）之前，可以不必加声明。

函数的调用（Function call）

- 函数调用形式

  函数名（实际参数表）；

注：实际参数（argument）

- 形式参数和实际参数的个数、排列次序、类型要完全相同。
- 实际参数可以是常量、变量、表达式，甚至是另一个函数调用。
- 传递方式：值传递 和 引用传递。
  - 值传递：被调函数获得主调函数传来的实际参数的拷贝。被调函数可以改变这些拷贝，但这对主调函数的环境没有影响。
  - 引用传递：指针这一章的重点！

注意：形式参数（形参，parameters）和实际参数（实参，arguments）的区别

形式参数 (parameters)	实际参数 (arguments)
当函数被声明和定义时所提及的参数被称为形式参数	当函数被调用时，随着调用被传递的参数被称为实际参数
The values which are defined at the time of the function prototype or definition of the function are called as parameters.	When a function is called, the values that are passed during the call are called as arguments.

函数的调用（Function call）

```
#include <iostream.h>

int Getmax(int a, int b)
{
  return a > b ? a : b;
}

int main()
{   int x, y;

    cin >> x >> y;
    cout << Getmax(x , y );
    return 0;
 }
```

形式参数表

实际参数表

函数调用

函数的调用（Function call）

```
#include <iostream.h>

int Getmax(int a, int b)
{
  return a > b ? a : b;
}

int main()
{  int x, y;

   cin >> x >> y;
   cout << Getmax(x , y );
   return 0;
 }
```

函数实现；如在主函数前则不需要原型声明

函数调用

```
#include <iostream.h>

int Getmax(int a, int b);

int main()
{     int x, y;
      cin >> x >> y;
      cout << Getmax(x , y );
      return 0;
 }

int Getmax(int a, int b)
{
  return a > b ? a : b;
}
```

函数原型声明

函数调用

函数实现

函数的调用（Function call）

- 调用方式

1. 作为语句：PrintStar();//打印一个三角形出来，后面会讲到

2. 作为表达式的一部分

如要计算阶乘（factorial）之和 5!+4!+7!

```
x=fact(5) + fact (4) + fact (7)
```

3. 作为函数的参数

```
PrintStarN ( fact(5) + fact (4) + fact (7) );
```

函数调用的基本方式 (1)

- **函数无返回值时，单独作为一个函数调用语句**

```
void  printstar()
{
 cout << "     *\n";
 cout << "    ***\n";
 cout << "   *****\n";
 cout << "  *******\n";
 cout << "*********\n";
}
```

```
int main()
{   …
   printstar();
    …
   return 0;
}
```

函数调用的基本方式 (2)

- **调用者通过函数名调用函数**
  - **有返回值时，可放到一个赋值表达式语句中，作为表达式的一部分**

```
#include <iostream.h>

int Getmax(int a, int b)
{
  return a > b ? a : b;
}
```

```
main()
{
    int x, y, max;

    cin >> x >> y;
    max = Getmax(x,y);
}
```

函数调用的基本方式 (3)

- **还可放到一个函数调用语句中，作为另一个函数的参数**

```
int Getmax(int a, int b)
{
  return a > b ? a : b;
}
int fact(int n) //求阶乘
{   int s=1, i;
    if (n < 0)   return(0);
    for (i = 1; i <= n; ++i)
          s *= i;
    return(s);
}
```

```
main()
{
    int x, y, max;

    cin >> x >> y;
    cout<< fact(Getmax(x,y));

}
```

判断下列陈述的对错

- 下列说法正确的是：
- A. C++语言的函数必须要有return语句
- B. C++语言程序中，要调用的函数必须在main()函数中定义
- C. C++语言程序中，只有int类型函数可以未经声明出现在调用之后
- D. C++语言程序中，main()函数必须放在程序开始的地方

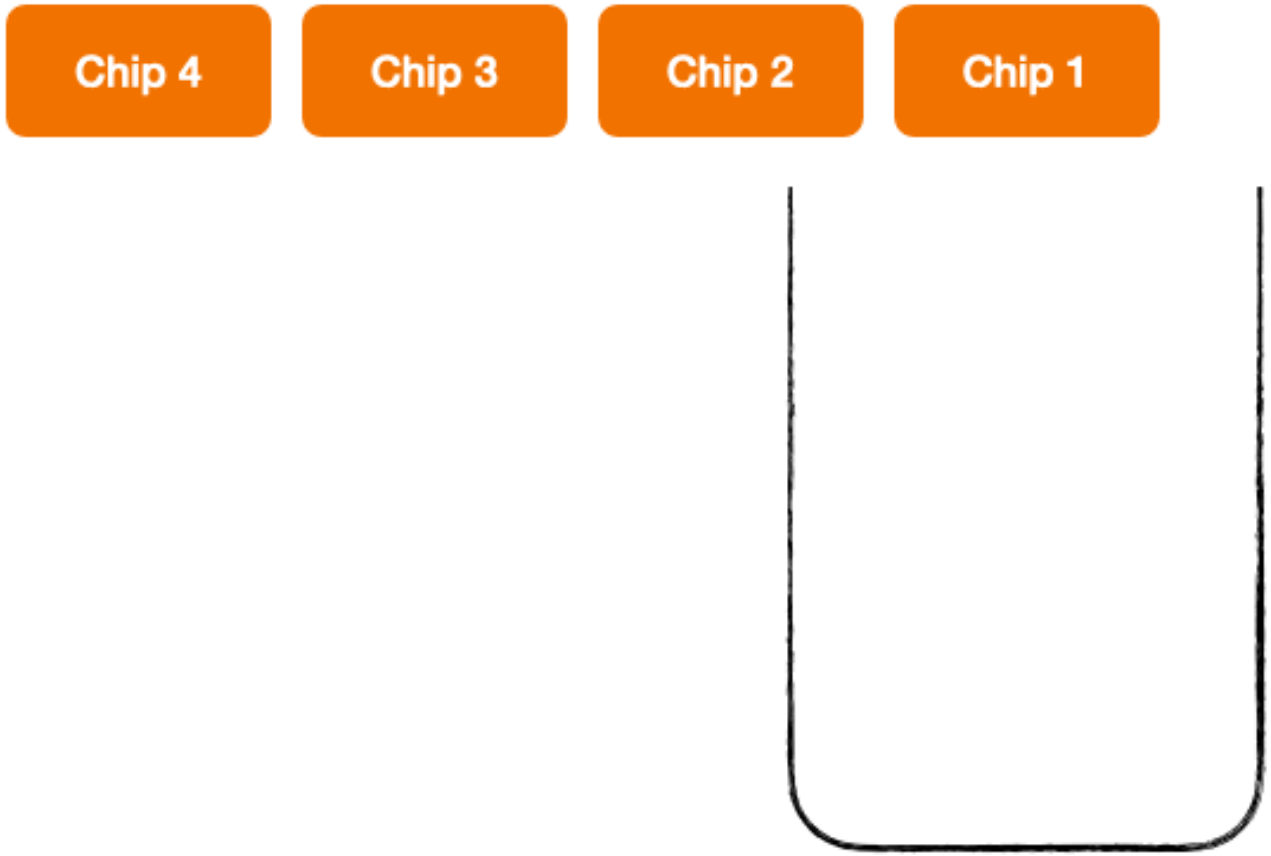
课堂练习题：编写一个求x的n次方的函数

```
#include <iostream>
using namespace std;

//计算x的n次方
double power(double x, int n) {
  double val = 1.0;
  while (n--) val *= x;
  return val;
}

int main() {
  cout << "5 to the power 2 is "
       << power(5, 2) << endl;
  return 0;
}
```

函数调用的过程 --- 总览

1. 在主程序中计算每个实际参数值；
2. 用实际参数值初始化形式参数，如类型不一致则自动转换；
3. 依次执行函数体的每个语句，直到遇见return语句或函数体结束；
4. 计算return后面的表达式的值，如类型不一致则自动转换；
5. 回到调用函数，用返回值置换函数调用，继续主程序的执行。

函数调用的过程---函数栈

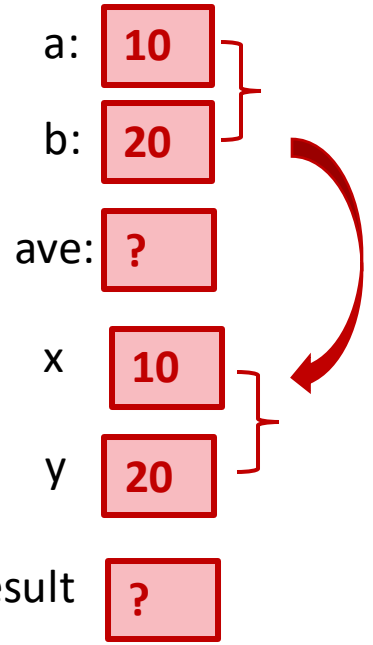


函数调用的过程---函数栈

- **执行函数调用（第1、2步）**
  - 现场保护、并为函数内的局部变量（包括形参）分配内存
  - 把实参值复制一份给形参，单向传值（实参→形参）
  - 实参与形参的数目、类型和顺序要一致

```
int main()
{
    int  a=10,  b=20, ave;
     …
    ave = Average(a, b);
     …
    return 0;

}
```

```
int Average(int x, int y)
{
    int result;

    result = (x + y) / 2;

    return result;
}
```



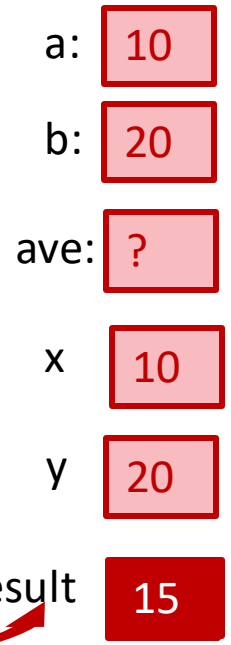
【扩展阅读】函数栈实现原理https://segmentfault.com/a/1190000017151354

函数调用的过程

- **程序控制权交给被调函数（第3、4步）**
  - 执行函数内的语句
  - 当执行到return语句或}时，从函数退出

```
int main()
{
    int  a=10,  b=20, ave;
     …
    ave = Average(a, b);
     …
    return 0;

}
```

```
int Average(int x, int y)
{
    int result;

    result = (x + y) / 2;

    return result;
}
```

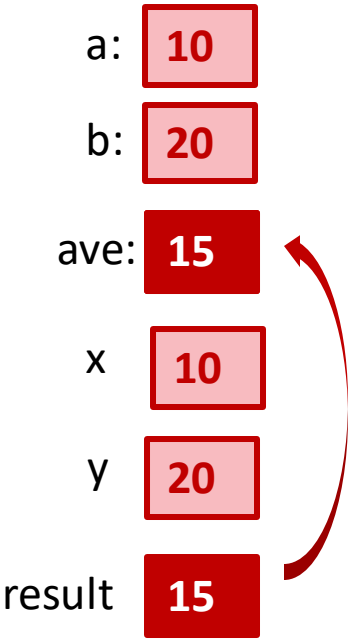


函数调用的过程

- **从函数退出时（第5步）**
  - 返回到本次函数调用的地方
  - 把函数值返回给主调函数，同时把控制权还给调用者
  - 收回分配给函数内所有变量（包括形参）的内存

```
int main()
{
    int  a=10,  b=20, ave;
     …
    ave = Average(a, b);
     …
    return 0;

}
```

```
int Average(int x, int y)
{
    int result;

    result = (x + y) / 2;

    return result;
}
```



讨论

- 请问程序输出结果为？

```
void swap(int x, int y);
int main()
{
    int a(5),b(8);
    cout << "a="<<a<<", b="<<b<< endl;
    swap(a,b);
    cout << "a="<<a<<", b="<<b<< endl;
    return 0;
}
```

```
void swap(int x, int y);
{
    int temp;
    temp=x;
    x=y;
    y=temp;
}
```

C⁺ main

TACpp 〉 test 〉 C⁺ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

test 〉 Thread 1 〉 0 main

L a = (int) 86053
L b = (int) 32766

(lldb)

C⁺ main

TACpp 〉 test 〉 C⁺ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

test 〉 Thread 1 〉 0 main

L **a** = (int) 5

L **b** = (int) 8

(lldb)

C+ main

TACpp 〉 test 〉 C+ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

test 〉 Thread 1 〉 0 swap(int, int)

A x = (int) 5
A y = (int) 8
L temp = (int) 1

a=5, b=8
(lldb)

C⁺ main

TACpp 〉 test 〉 C⁺ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

test 〉 Thread 1 〉 0 swap(int, int)

A **x** = (int) 5
A **y** = (int) 8
L **temp** = (int) 5

a=5, b=8
(lldb)

C⁺ main

TACpp 〉 test 〉 C⁺ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

test 〉 Thread 1 〉 0 swap(int, int)

A **x** = (int) 8
A **y** = (int) 8
L **temp** = (int) 5

a=5, b=8
(lldb)

C⁺ main

TACpp 〉 test 〉 C⁺ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

test 〉 Thread 1 〉 0 swap(int, int)

A **x** = (int) 8
A **y** = (int) 5
L **temp** = (int) 5

a=5, b=8
(lldb)

C⁺ main

TACpp 〉 test 〉 C⁺ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

无返回值，故无法置换main函数中变量！

test 〉 Thread 1 〉 0 main

L a = (int) 5
L b = (int) 8

a=5, b=8
(lldb)

C⁺ main

TACpp 〉 test 〉 C⁺ main 〉 No Selection

```
5
6   #include <iostream>
7
8   using namespace std;
9
10  void swap(int x, int y);
11
12  int main()
13  {
14      int a(5),b(8);
15      cout << "a="<<a<<", b="<<b<< endl;
16      swap(a,b);
17      cout << "a="<<a<<", b="<<b<< endl;
18      return 0;
19  }
20
21  void swap(int x, int y)
22  {
23      int temp;
24      temp=x;
25      x=y;
26      y=temp;
27  }
```

test 〉 Thread 1 〉 0 main

L a = (int) 5
L b = (int) 8

```
a=5, b=8
a=5, b=8
(lldb)
```

课堂习题

在C++程序中，如果实参是普通变量，调用函数时，以下说明正确的是：

- A.实参和形参可以共用存储单元
- B.实参和形参各自占用一个独立的存储单元
- C.可以让用户指定是否共用存储单元
- D.由计算机系统自动确定是否共用存储单元

预习工作

- 函数的用途和定义
- 函数的使用
- 变量的作用域与存储类别
- 函数参数—数组和默认值
- 内联、重载、模版、递归