《计算概论A》课程 程序设计部分
C++程序中的函数（2）

李　戈

北京大学 信息科学技术学院 软件研究所

lige@sei.pku.edu.cn

北京大学

数组与函数

北京大学

复习（1）

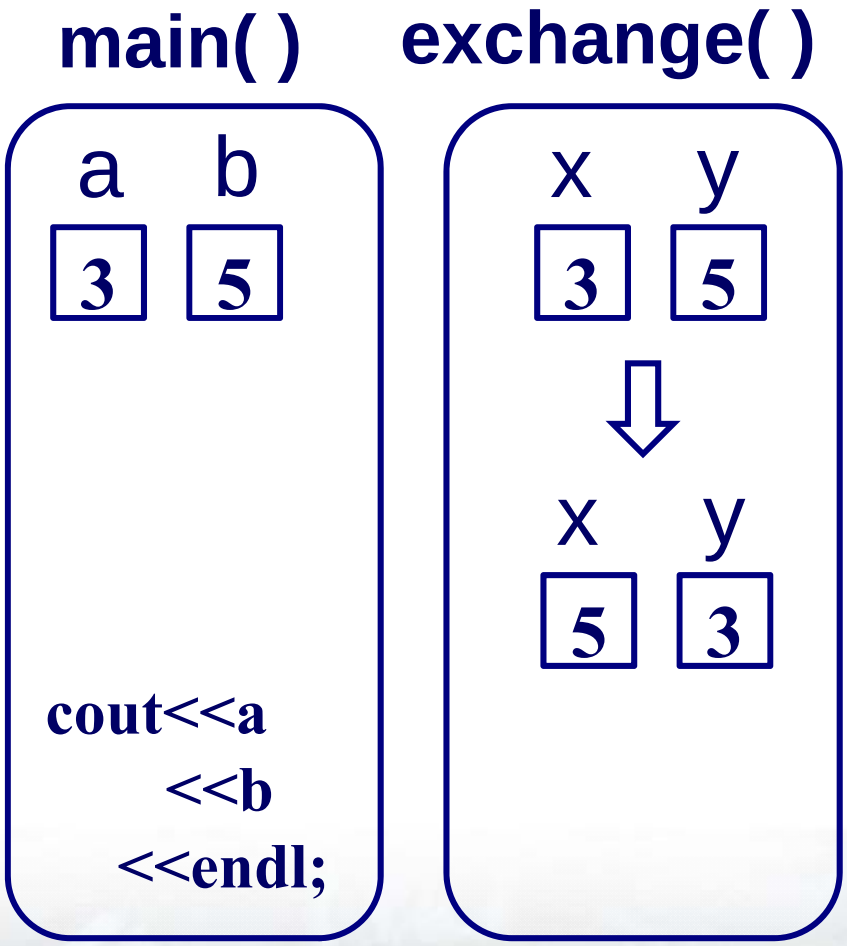
```
#include<iostream>
using namespace std;
void exchange( int x, int y)
{
    int p;
    if (x < y)
    { p = x; x = y; y = p; }
}
int main()
{
    int a = 3, b = 5;
    exchange(a, b);
    cout<<a<<" "<<b<<endl;
    return 0;
}
```

程序运行结果？

北京大学

复习（2）

```
#include<iostream>
using namespace std;
void exchange( int x, int y)
{
    int p;
    if (x < y)
    { p = x; x = y; y = p; }
}
int main()
{
    int a = 3, b = 5;
    exchange(a, b);
    cout<<a<<" "<<b<<endl;
    return 0;
}
```



数组元素做函数参数

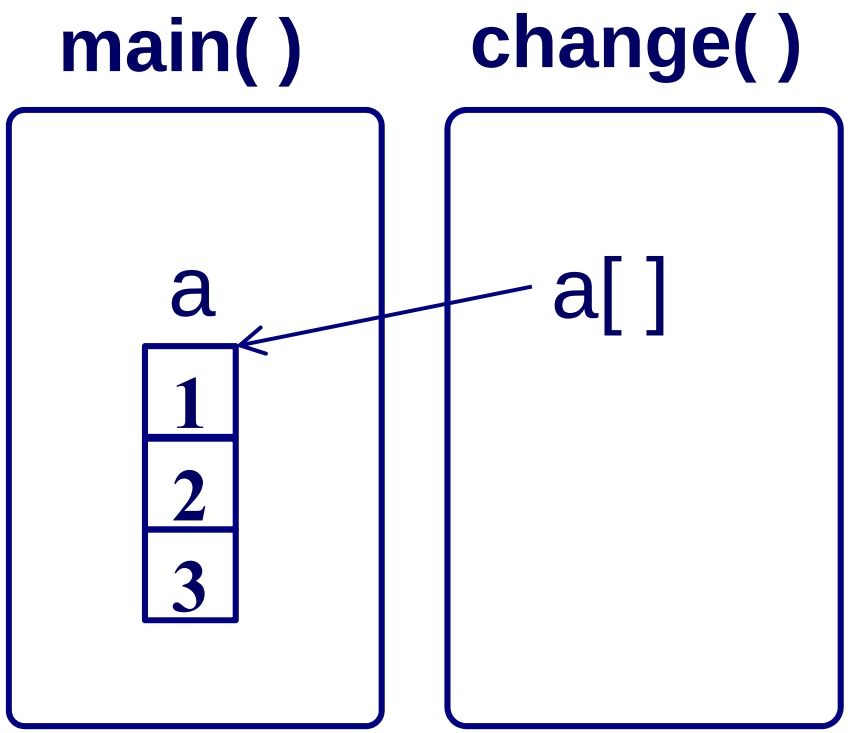
```
#include<iostream>
using namespace std;
void change(int a, int b, int c)
{
        a = 0;
        b = 0;
        c = 0;
}
int main()
{
        int a[3]={1, 2, 3};
        change(a[0], a[1], a[2]);
        cout<<a[0]<<" "<<a[1]<<" "<<a[2]<<endl;
        return 0;
}
```

数组名做函数参数

```
#include<iostream>
using namespace std;
void change(int a[ ])
{
        a[0] = 0;
        a[1] = 0;
        a[2] = 0;
}
int main()
{
        int a[3]={1, 2, 3};
        change(a);
        cout<<a[0]<<" "<<a[1]<<" "<<a[2]<<endl;
        return 0;
}
```

数组名做函数参数

```
#include<iostream>
using namespace std;
void change(int a[ ])
{
        a[0] = 0;
        a[1] = 0;
        a[2] = 0;
}
int main()
{
        int a[3]={1, 2, 3};
        change(a);
        cout<<a[0]<<" "<<a[1]<<" "<<a[2]<<endl;
        return 0;
}
```



思考题(1)

```
#include<iostream>
using namespace std;
void change(char a, char b)
{
        a = a ^ b;
        b = b ^ a;
        a = a ^ b;
}
int main()
{
        char a[3]={'A','B'};
        change(a[0], a[1]);
        cout<<a[0]<<" "<<a[1]<<endl;
        return 0;
}
```

思考题(2)

```
#include<iostream>
using namespace std;
void change(char a[ ])
{
        a[0] = a[0] ^ a[1];
        a[1] = a[1] ^ a[0];
        a[0] = a[0] ^ a[1];
}
int main()
{
        char a[3]={'A','B'};
        change(a);
         cout<<a[0]<<" "<<a[1]<<endl;
        return 0;
}
```

思考题(3)

```
#include<iostream>
using namespace std;
void change(char a[ ])
{
        a[0] = a[0] ^ a[1];
        a[1] = a[1] ^ a[0];
        a[0] = a[0] ^ a[1];
}
int main()
{
        char a[3]={'A','B'};
        change(a);
         cout<<a<<endl;
        return 0;
}
```

多维数组名做函数参数

例: 有一个3×4的矩阵，将所有元素置0。

```
#include<iostream>
using namespace std;
void reset(int array[][4])
{
    for(int i = 0; i < 3; i++)
        for(int j = 0; j < 4; j++)
            array[i][j] = 0;
}
void main()
{
    int main_array[3][4]={{1,3,5,7},{2,4,6,8},{5,7,4,2}};
    reset(main_array);
    for(int i = 0; i < 3; i++)
        for(int j = 0; j < 4; j++)
            cout<<main_array[i][j]<<" ";
}
```

结构体与函数

北京大学

什么是结构体

- 声明一个名为“学生”的结构体

```
struct student \\结构体的名字为“student”;
 {
        int     id;          \\声明学号为int型;
        char    name[20];    \\声明姓名为字符数组;
        char    sex;         \\声明性别为字符型;
        int     age;         \\声明年龄为整型;
        float   score;       \\声明成绩为实型;
        char    addr[30];    \\声明地址为字符数组
 };        \\注意大括号后的“;”
```

定义结构体类型的变量

- 定义结构体变量

  struct student student1, student2;

  （结构体类型名）（结构体变量名）;

  - 对比：

    int a;(struct student 相当于 int)

    float a; (struct student 相当于 float)

结构体变量的引用

- 引用结构体变量中成员的方式为

  结构体变量名.成员名

  - 如：student1.id=10010;

    student1.birthday.month = 10;

- 不能将一个结构体变量作为一个整体进行输入和输出
  - 不正确的引用：cout<<student1;　　cin>>student1;
- 只能对结构体变量中的各个成员分别进行输入和输出
  - 正确的引用：cin>>student1.id;　cout<<student1.id;

结构体变量的初始化

```
struct    date
{   int    month;
    int    day;
    int    year; };
struct    student
{   int    num;
    char    name[20];
    char    sex;
    int    age;
    struct     date birthday;
    char    addr[30];
}student1={121, “zhang", 'M', 20, {12, 30, 2000}, "PKU"},
 student2={122, “wang", 'M', 20, {12, 30, 2000}, "PKU"};
student student3
            = {123, “zhao", 'M', 20, {12, 30, 2000}, "PKU"};
```

- 结构体可以在定义时进行初始化

北京大学

结构体变量的赋值

- 类型相同的结构体变量可以进行赋值
  - 例如：

要将student1和student2互换

```
temp = student1;
student1 = student2;
student2 = temp;
```

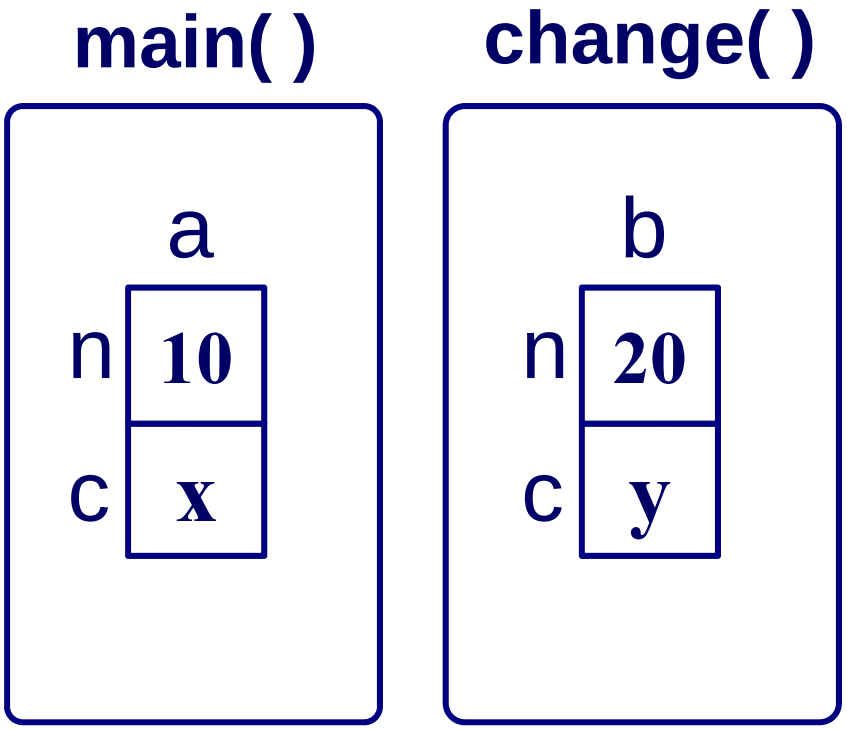
  - (temp也必须是相同类型结构体变量)

结构体做函数参数（1）

```
struct stru
{   int n;
    char c;
};
void change(stru b)
{   b.n=20;     b.c='y';}
int main()
{
    stru a = {10,'x'};
    change(a);
    cout<<a.n<<" "<<a.c<<endl;
    return 0;
}
```

结构体做函数参数（1）

```
struct stru
{  int n;
   char c;
};
void change(stru b)
{  b.n=20;    b.c='y';}
int main()
{
   stru a = {10,'x'};
   change(a);
   cout<<a.n<<" "<<a.c<<endl;
   return 0;
}
```



结构体做函数参数（2）

```
struct stru
{  int n;
   char c;
};
void change(stru b)
{  b.n=20;   b.c='y';}
int main()
{
   stru a = {10,'x'};
   change(a);
   cout<<a.n<<" "<<a.c<<endl;
   return 0;
}
```

- 结构体做参数时采用值传递的方式；
- 系统会构造一个结构体的副本给函数使用；

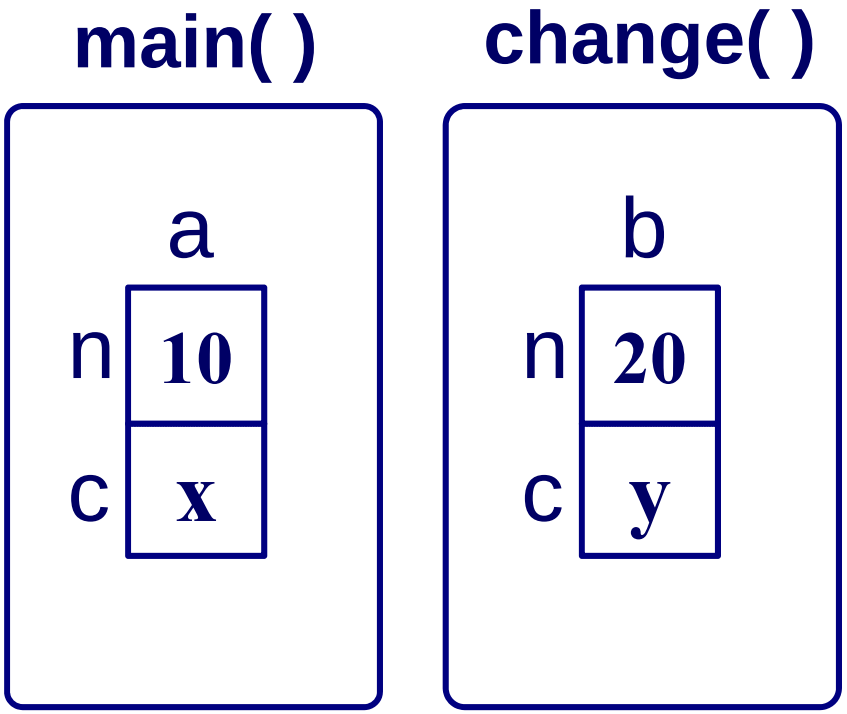
```
10 x
Press any key
```

结构体做函数返回值（1）

```
struct stru
{   int n;
    char c;
};
stru change(stru b)
{
    b.n=20;    b.c='y';
    return b;
}
int main()
{
    stru a = {10,'x'};
    a = change(a);
    cout<<a.n<<" "<<a.c<<endl;
    return 0;
}
```

结构体做函数返回值（2）

```
struct stru
{   int n;
    char c;
};
stru change(stru b)
{
    b.n=20;    b.c='y';
    return b;
}
int main()
{
    stru a = {10,'x'};
    a = change(a);
    cout<<a.n<<" "<<a.c<<endl;
    return 0;
}
```



结构体做函数返回值（3）

```
student GetStudent()
{  student t;
   cout <<“请输入学号”;
   cin >> t.No;
   cout<<“请输入学生姓名：”;
   cin.getline(t.name,20);
   cout<<“请输入数学、英语、C++成绩：”;
   cin>>t.score[0]>> t.score[1]>> t.score[2];
   return t;
}
int main( )
{  student stu[4];
   for(int i=0;i<4;i++)
   {
     stu[i] = GetStudent();
     print(stu[i]);
   }
}
```

- 一个函数可以返回一个结构体！

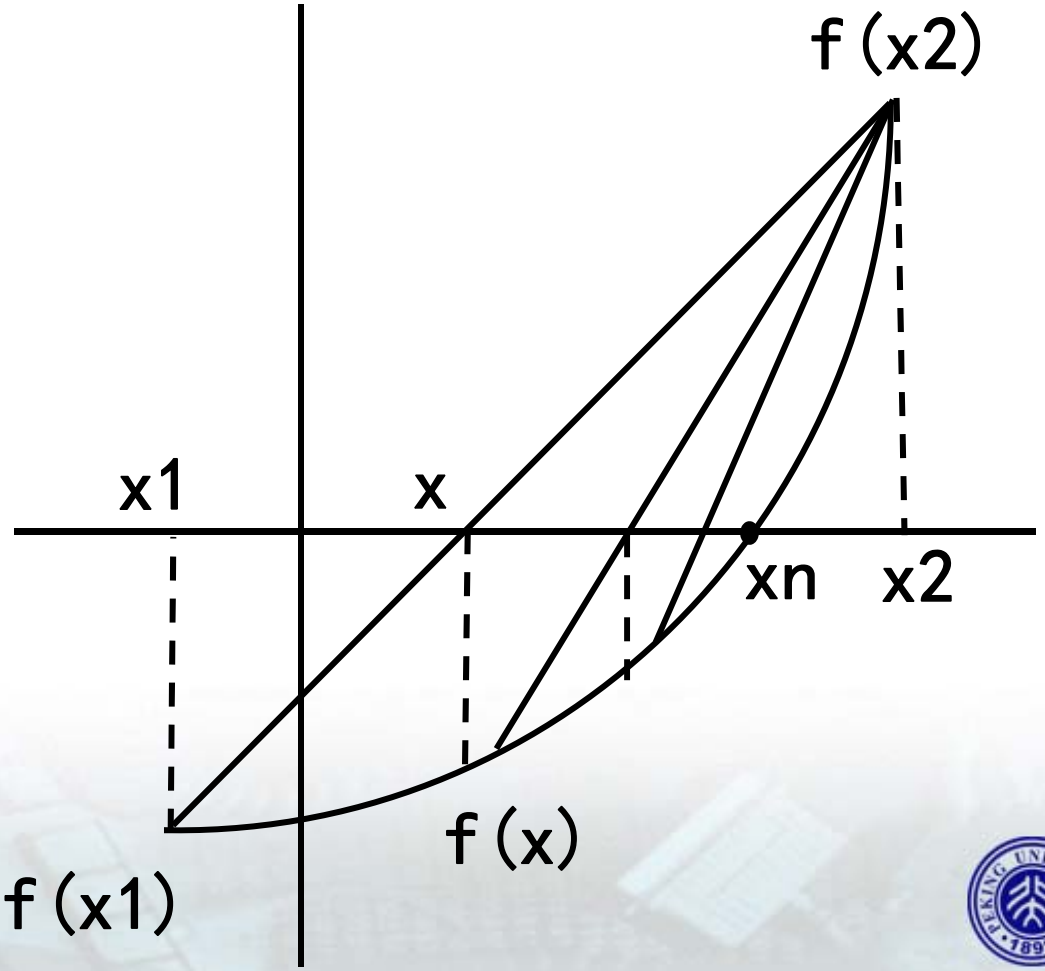
嵌套与递归

北京大学

函数嵌套调用程序实例

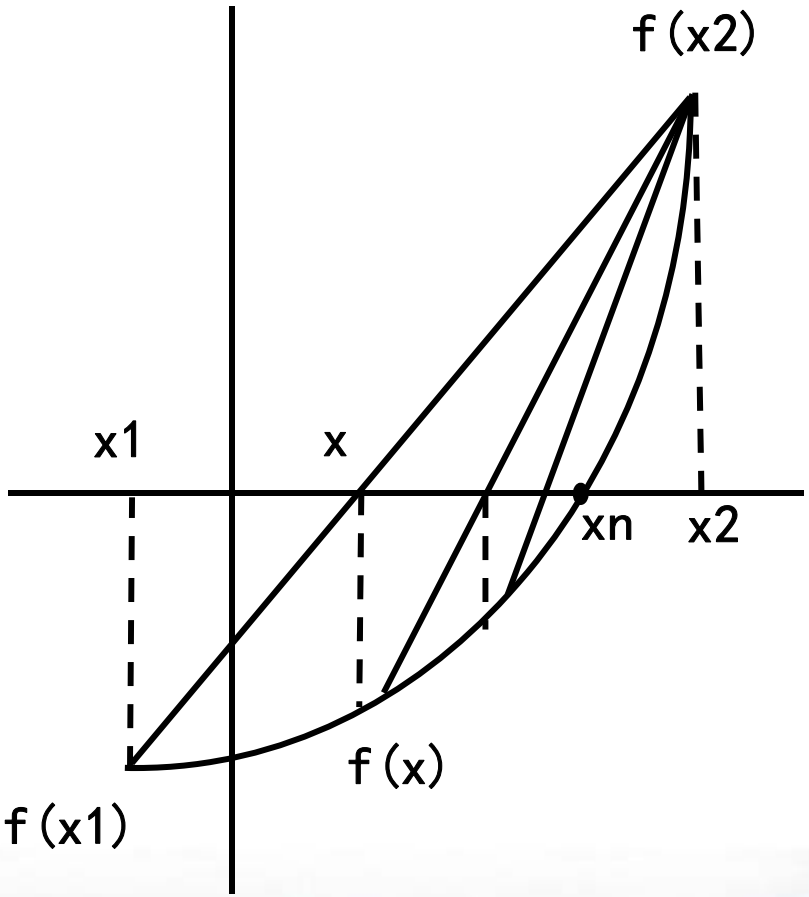
[例]用弦截法求方程的根

$x^3+5x^2+16x-80=0$



函数嵌套调用程序实例

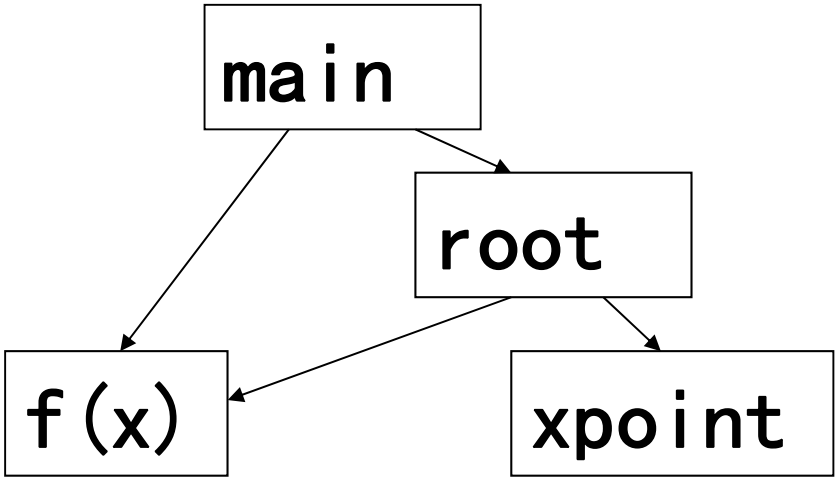
输入x1, x2, 求f(x1), f(x2)	
直到f(x1)*f(x2)<0	
求f(x1)与f(x2)的连线与x轴的交点x	
y=f(x), y2=f(x2)	
y与y2异号吗? y	n
x1=x	x2=x
直到|y|<ε	
root=x, 输出x	



函数嵌套调用程序实例

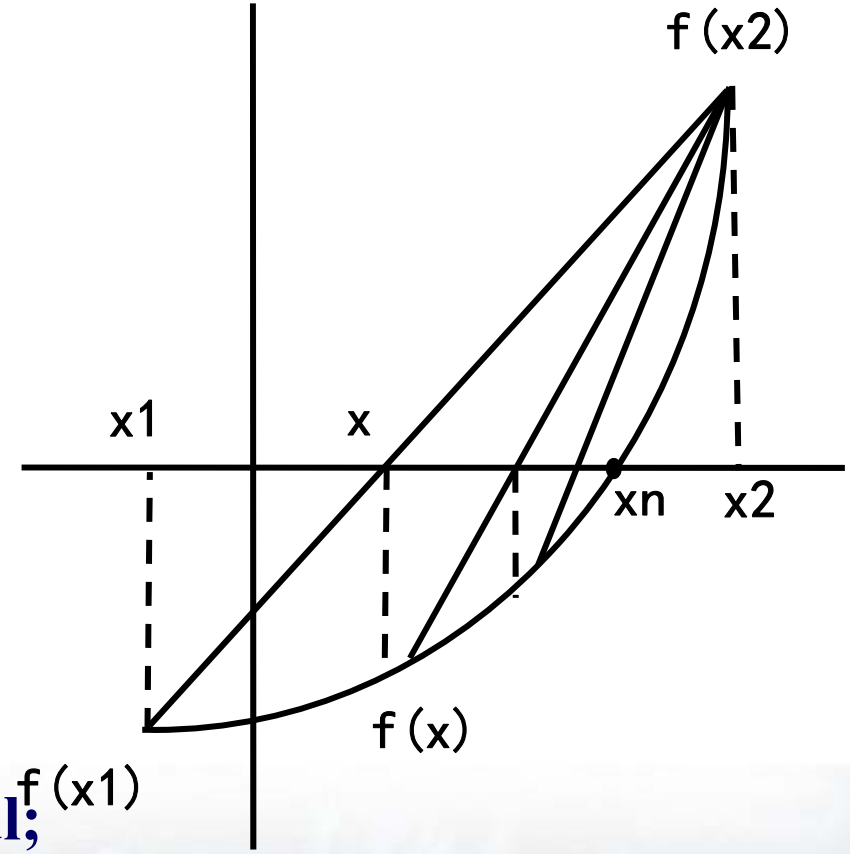
可以组织出的函数

- 求f(x):
  - 输入x，输出f (x);
- xpoint (x1, x2):
  - 输入x1, x2 输出弦与x轴的交点；
- 总方程：root(x1, x2):
  - 输入x1, x2 输出根；



函数嵌套调用程序实例

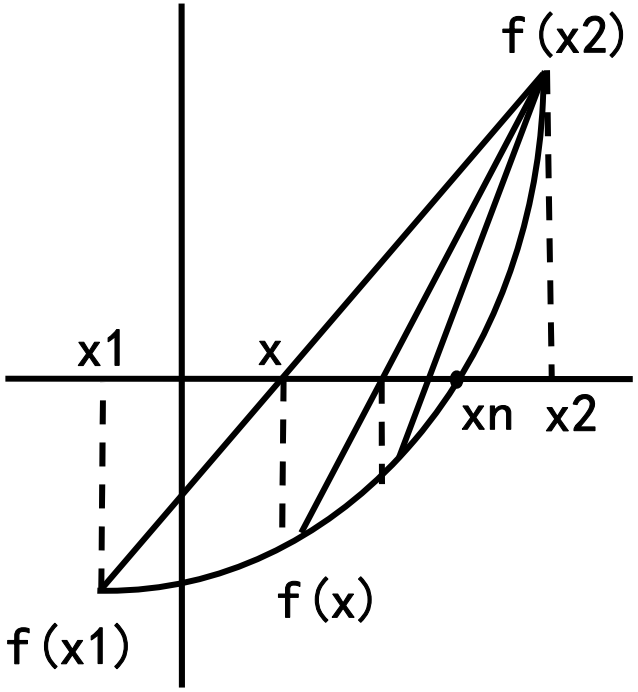
```
#include<iostream>
#include<cmath>
using namespace std;
float Root(float, float);
float XPoint(float, float);
float f(float x);
int main( )
{
    float x1, x2, f1, f2, x;
    do{
        cin>>x1>>x2;
        f1=f(x1); f2=f(x2);
    }while(f1*f2 >= 0);
    x=Root(x1,x2);
    cout<<"The root is"<<x<<endl;
    return 0;
}
```



北京大学

函数嵌套调用程序实例

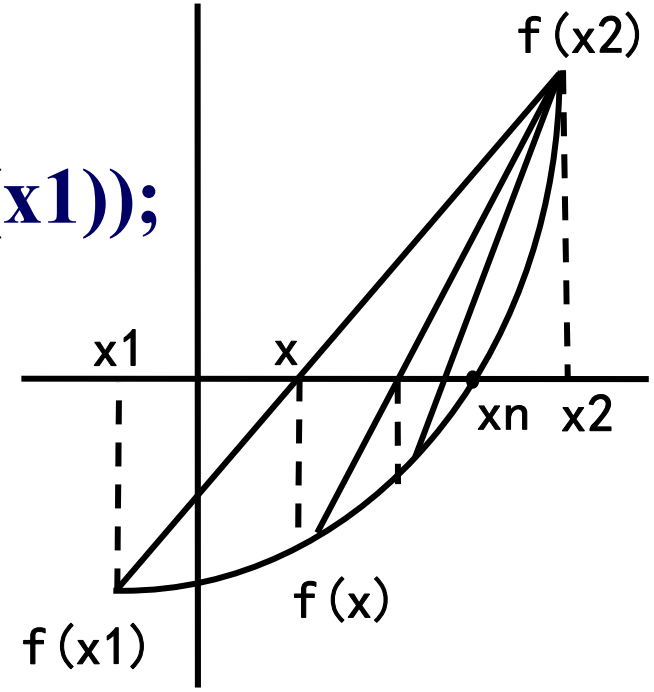
```
float Root(float x1, float x2)
{
    float x, y, y1;
    y1 = f(x1);
    do {
        x = XPoint(x1, x2);
        y = f(x) ;
        if (y*y1<0)
        {  x1= x; }
        else
            x2 = x;
    } while(fabs(y)>=0.0001);
    return(x);
}
```



函数嵌套调用程序实例

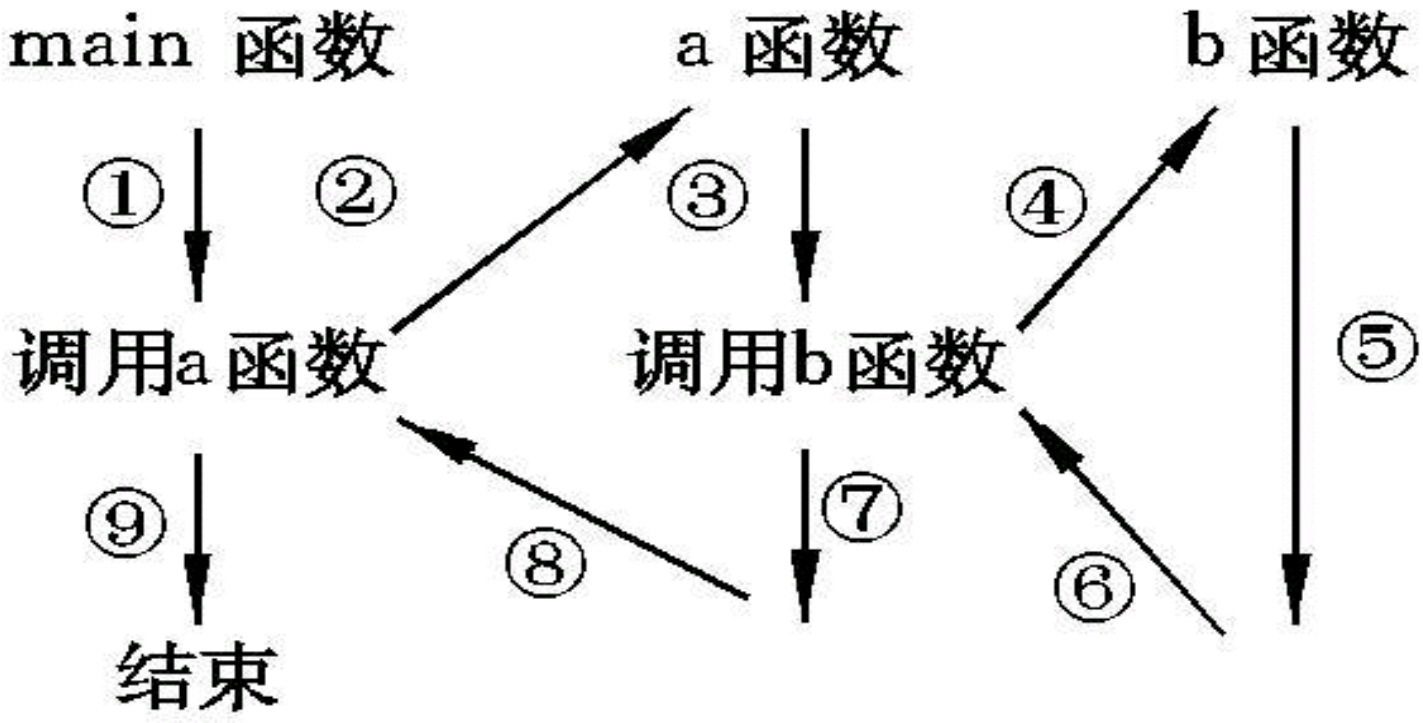
```
float XPoint(float x1, float x2)
{
    float x;
    x=(x1*f(x2)-x2*f(x1))/(f(x2)-f(x1));
    return(x);
}

float f(float x)
{
    float y;
    y=((x-0.5)*x+16.0)*x-80.0;
    return(y);
}
```



函数的嵌套调用

- 函数不能嵌套定义，但可以嵌套调用
  - 在调用一个函数的过程中，又调用另一函数



例题分析（1）

- 有5个人坐在一起，问第5个人多少岁？他说比第4个人大2岁。问第4个人岁数，他说比第3个人大2岁。问第3个人，又说比第2个人大2岁。问第2个人，说比第1个人大2岁。最后问第1个人，他说是10岁。请问第5个人多大。
  - Age[1] = 10;
  - Age[2] = age[1] + 2;
  - Age[3] = age[2] + 2;
  - Age[4] = age[3] + 2;
  - Age[5] = age[4] + 2;

例题分析（1）

```
#include <iostream.h>
int main()
{  int age[6];
   age[1] = 10;
   for (int i = 2; i <= 5; i++)
   {  age[i] = age[i-1] + 2;}
   cout << "第5个人的年龄是：  "<<age[5] <<endl;
   return 0;
}
```

另一种解决方案

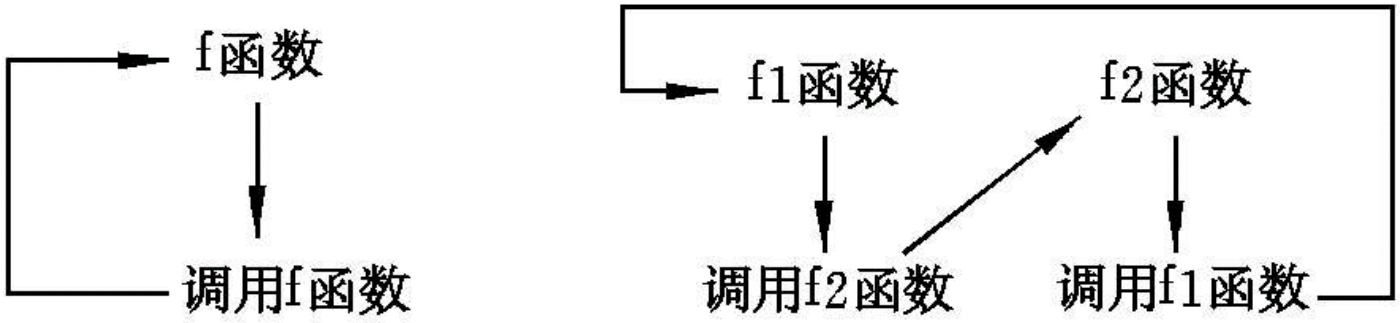
```
#include<iostream.h>
int age(int n)
{   int c;
    if(n == 1)
        c = 10;
    else
        c = age(n-1)+2;
    return(c);
}
void main()
{   cout<<“第五个人的年龄是：”<<age(5);
}
```

另一种解决方案

```
int age(int n)
{
    int c;
    if(n == 1)
        c = 10;
    else
        c = age(n-1)+2;
    return(c);
}
```

递 归

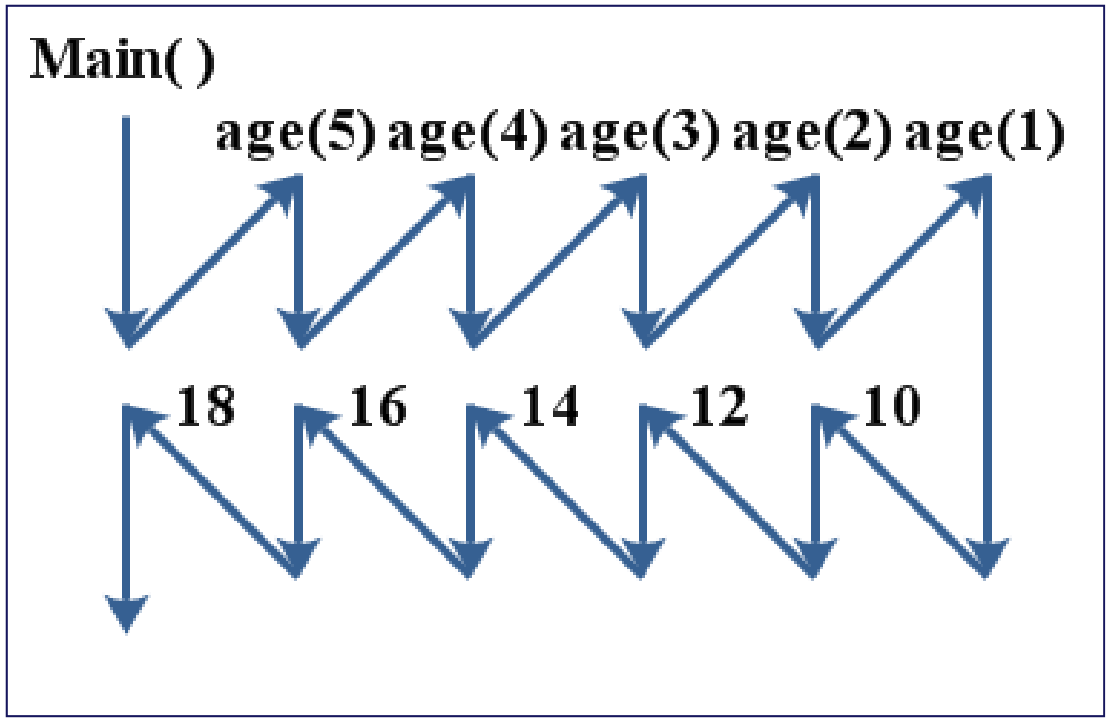
- 在调用一个函数的过程中又出现直接或间接地调用该函数本身。



- 程序中不应出现无终止的递归调用，必须控制只有在某一条件成立时才继续执行递归调用，否则就不再继续。

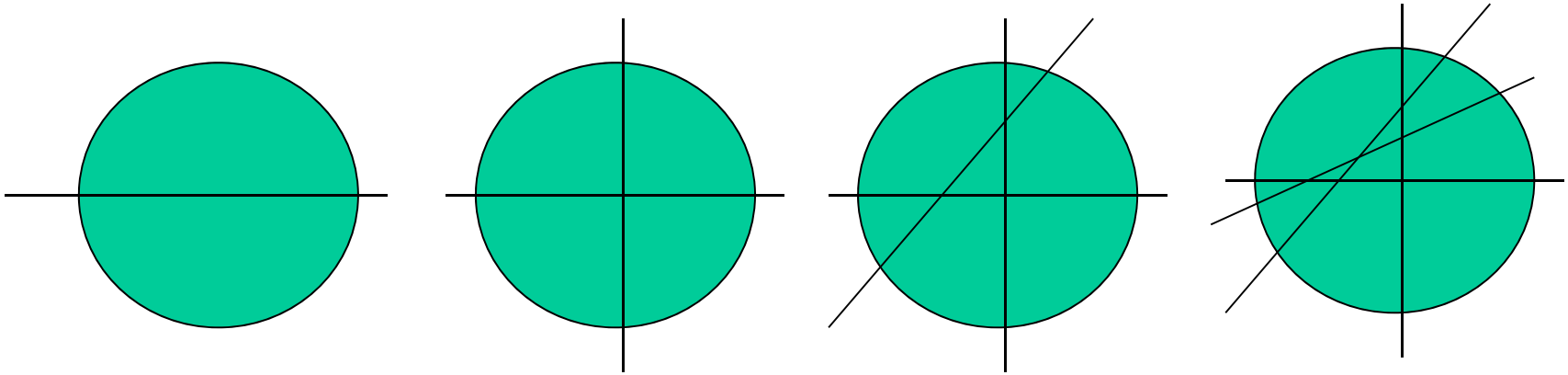
另一种解决方案

```
#include<iostream>
using namespace std;
int age(int n)
{   int c;
    if(n == 1)
        c = 10;
    else
        c = age(n-1)+2;
    return(c);
}
void main()
{
    cout<<“第五个人的年龄是：”<<age(5);
}
```



例题分析（2）

- 切饼，100刀最多能切多少块？



- q(1)=1+1=2
- q(2)=1+1+2=4;
- q(3)=1+1+2+3=7;
- q(4)=1+1+2+3+4=11;
- q(n)=q(n−1)+n;q(0)=1;

北京大学

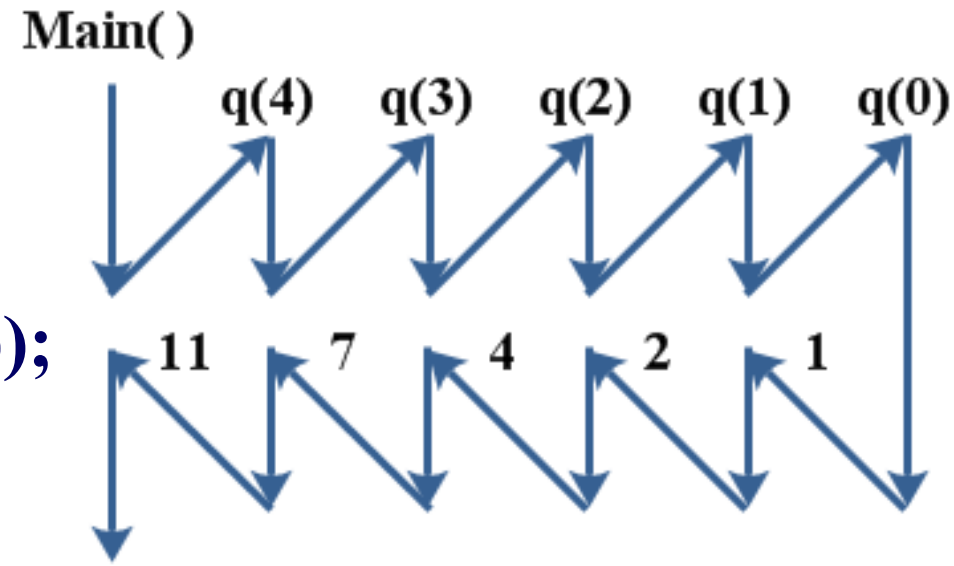
例题分析（2）

```
#include <iostream.h>
int main()
{  int q[101];
   q[0] = 1;
   for (int i = 1; i <= 100; i++)
   {  q[i] = q[i-1] + i;}
   cout<<"100刀最多可切"<<q[100]<<"块"<<endl;
   return 0;
}
```

另一种解决方案

```
#include<iostream>
using namespace std;
int q(int n){
    if (n == 0)
        return 1;
    else
        return( n + q(n-1));
}

int main(){
        cout<<q(4)<<endl;
        return 0;
}
```



好好想想,有没有问题?

谢　谢　！

北京大学