西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C++程序设计
Programming in C++

1011018

主讲：魏英，计算机学院

指针与函数

- 3、引用
- 4、指向函数的指针

- 通过对象名称直接访问对象，优点是直观，操作哪个对象一目了然，缺点一个函数内部不能使用另一个函数的局部变量；
- 通过指针（或地址）间接访问对象，优点是无所不能，缺点是程序中大量出现的间接访问，实在分不清具体是哪个对象，需要通过上下文去分析。
- C++扩充了C语言对象访问方式，提供了引用访问。通过引用访问对象，结合了按名访问和按地址访问各自的优点，非常适合作为函数参数。

- 1. 引用作为函数参数
- 简单地说，引用（reference）就是一个对象的别名（alias name），其声明形式为：

```
引用类型  &引用名称=对象名称 ， ……;
```

```
int x; //定义整型变量x
int &r=x; //声明r是x的引用
```

- 在C++中，引用全部是const类型，声明之后不可更改（即不能再是别的对象的引用）。

- 引用的规则
- （1）声明一个引用类型变量时，必须同时初始化它，声明它是哪个对象的别名，即绑定对象。例如：

```
int &r; //错误，引用是const类型，必须在声明时初始化
int x, &r=x; //正确 声明r是x的引用
```

- （2）不能有空引用，引用必须与有效对象的内存单元关联。

- （3）引用一旦被初始化，就不能改变引用关系，不能再作为其他对象的引用。例如：

```
int x, y; //定义整型变量x,y
int &r=x; //正确 声明r是x的引用
int &r=y; //错误 r不能再是别的对象的引用
```

- （4）指定类型的引用不能初始化到其他类型的对象上，例如：

```
double f; //定义浮点型变量f
int &r=f; //错误 r值整型的引用，不能绑定到浮点型的对象上
```

- （5）引用初始化与对引用赋值含义完全不同，例如：

```
int x; //定义整型变量x
int &r=x; //初始化  指明r是x的引用，即将r绑定到x
r=100; //引用赋值  100赋值到r绑定的内存单元中（即x）
```

- （6）取一个引用的地址和取一个对象的地址完全一样，都是用取地址运算。例如：

```
int x, &r=x; //定义整型变量x,y
int *p1=&x; //p1指向x
int *p2=&r; //p2指向r，本质上指向x
```

- 引用作为函数形参

- C++之所以扩充引用类型，主要是把它作为函数形参，使得C++中给一个函数传递参数有三种方法：
- ①传递对象本身；
- ②传递指向对象的指针；
- ③传递对象的引用。

【例19.5】变量、指针、引用作为函数参数比较。

```
1  //程序① 传递对象本身
2  #include <iostream>
3  using namespace std;
4  //对象作为函数形参
5  void swap(int a,int b)
6  { int t;
7    t=a, a=b, b=t;
8  }
9  int main()
10 { int x=10, y=20;
11   swap(x,y);
12   cout<<x<<","<<y;
13   return 0;
14 }
```

```
1  //程序② 传递对象的指针
2  #include <iostream>
3  using namespace std;
4  //指针作为函数形参
5  void swap(int *a,int *b)
6  { int t;
7    t=*a, *a=*b, *b=t;
8  }
9  int main()
10 { int x=10, y=20;
11   swap(&x,&y);
12   cout<<x<<","<<y;
13   return 0;
14 }
```

```
1  //程序③ 传递对象的引用
2  #include <iostream>
3  using namespace std;
4  //引用作为函数形参
5  void swap(int &a,int &b)
6  { int t;
7    t=a, a=b, b=t;
8  }
9  int main()
10 { int x=10, y=20;
11   swap(x,y);
12   cout<<x<<","<<y;
13   return 0;
14 }
```

- 显然，函数引用传递方式也可以实现多个数据结果返回到主调函数中，其功能与指针方式相同。但指针方式返回数据结果必须：
- ①实参为地址，即进行“&”取地址运算；
- ②形参分配指针变量接受实参地址；
- ③函数内部使用指针间接访问，即进行“*”间接访问运算。而引用传递方式把这个过程简化了。

- 使用引用作为函数形参，比使用指针变量简单、直观、方便，特别是避免了在被调函数中出现大量指针间接访问时，所指对象究竟是哪个具体对象伤脑筋的问题，从而降低了编程的难度。

- 引用作为函数返回值
- 函数的返回值可以是引用类型，即函数返回引用，其定义形式为：

```
引用类型&  函数名(形式参数列表)
{
        函数体
}
```

【例19.6】引用作为函数返回值举例。

```
1  //程序① 函数返回值
2  #include <iostream>
3  using namespace std;
4  int max(int a,int b)
5  { return (a>b?a:b); }
6  int main()
7  { int x=10,y=20,z;
8    z = max(x,y);
9    cout << z;
10   return 0;
11 }
```

```
1  //程序② 函数返回指针
2  #include <iostream>
3  using namespace std;
4  int* max(int a,int b)
5  { return (a>b? &a:&b); }
6  int main()
7  { int x=10,y=20,*z;
8    z = max(x,y);
9    cout << *z;
10   return 0;
11 }
```

```
1  //程序③ 函数返回引用
2  #include <iostream>
3  using namespace std;
4  int& max(int &a,int &b)
5  { return (a>b? a:b); }
6  int main()
7  { int x=10,y=20,z;
8    z = max(x,y);
9    cout << z;
10   return 0;
11 }
```

- 可以看出，函数返回引用与函数返回值有重大区别，它不是返回一个临时对象，而是相当于返回实体对象本身。正因为如此，函数返回引用可以作为左值。例如：

```
int& fun(int &a,int &b)
{ return (a>b? a:b); }
int x=10,y=20,z=5;
fun(x,y)=z; //调用fun函数后相当于y=z;
cout << y;
```

- 函数是实现特定功能的程序代码的集合，实际上，函数代码在内存中也要占据一段存储空间（代码区内），这段存储空间的起始地址称为函数入口地址。C++规定函数入口地址为函数的指针，即函数名既代表函数，又是函数的指针（或地址）。

- C++允许定义指向函数的指针变量，定义形式为：

```
返回类型 (*函数指针变量名)(形式参数列表),...;
```

- 它可以指向如下形式的函数：

```
返回类型 函数名(形式参数列表)
{
        函数体
}
```

```
int (*p)(int a, int b); //定义函数指针变量
```

- 使函数指针指向函数
- 可以将函数的地址赋值给函数指针变量，形式为

函数指针变量=函数名；

- 它要求函数指针变量与指向函数必须有相同的返回类型、参数个数、参数类型。

- 例如假设：

```
int max(int a, int b); //max函数原型
int min(int a, int b); //min函数原型
int (*p)(int a, int b); //定义函数指针变量
```

- 则

```
p=max;
```

- 称p指向函数max。它也可以指向函数min，即可以指向所有与它有相同的返回类型、参数个数、参数类型的函数。

- 通过函数指针调用函数
- 对函数指针间接引用即是通过函数指针调用函数，一般形式为：

```
函数指针(实参列表)
```

- 通过函数指针调用函数，在实参、参数传递、返回值等方面与函数名调用相同。例如：

```
c=p(a,b); //等价于c=max(a,b);
```

- 函数指针的用途

- 指向函数的指针多用于指向不同的函数，从而可以利用指针变量调用不同函数，相当于将函数调用由静态方式（固定地调用指定函数）变为动态方式（调用哪个函数是由指针值来确定）。熟练掌握函数指针的应用，有利于程序的模块化设计，提高程序的可扩展性。

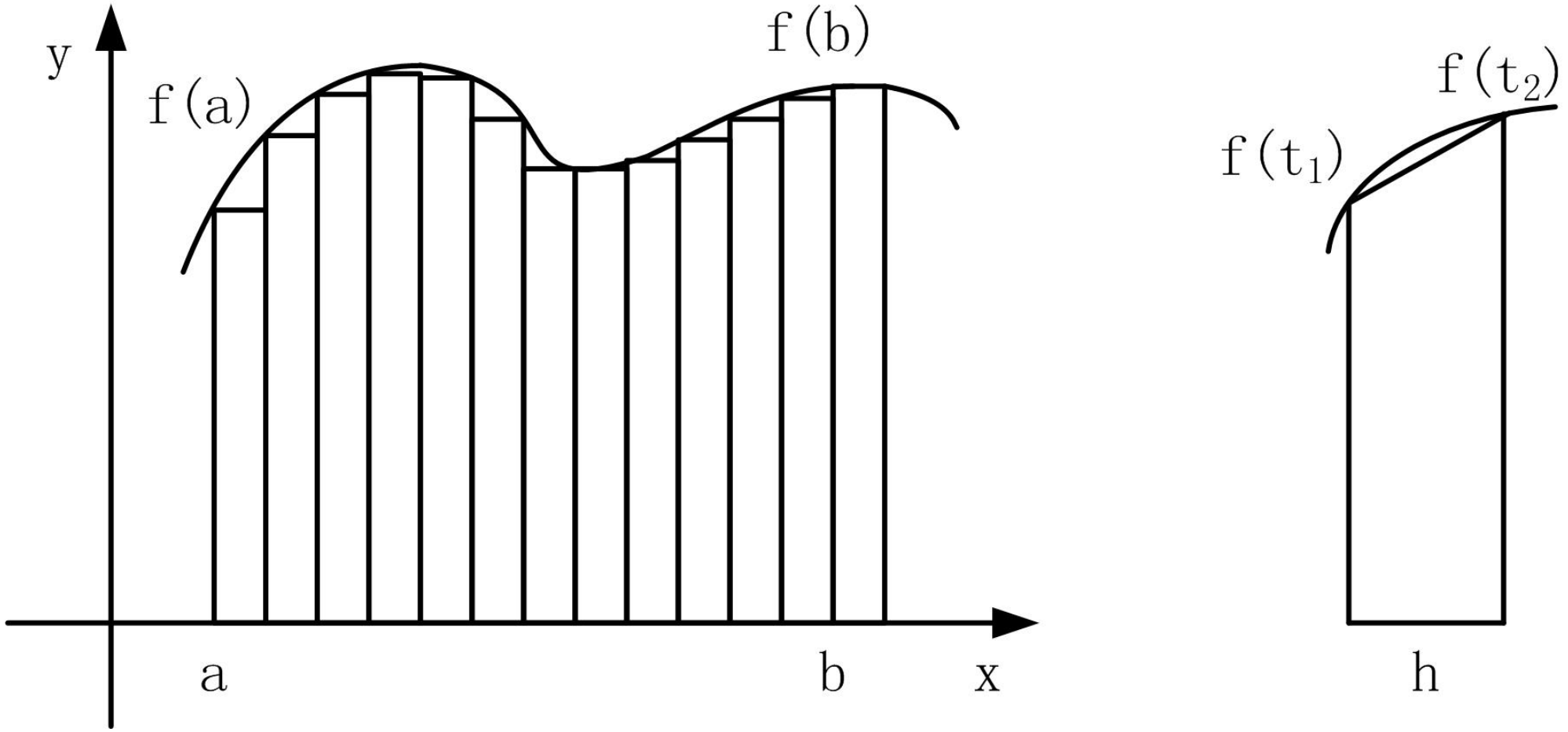
【例19.7】编写程序计算如下公式。

$$\int_a^b (1+x)dx + \int_a^b e^{-\frac{x^2}{2}}dx + \int_a^b x^3 dx$$

说明：

这里用梯形法求定积分 $\int_a^b f(x)dx$ 的近似值。如图所示，求 f(x)的定积分就是求f(x)曲线与x轴包围图形的面积，梯形法是把所要求的面积垂直分成n个小梯形，然后面积求和。

图19.2 梯形法求定积分示意



```
#include <iostream>
#include <cmath>
using namespace std;
double integral(double a,double b,double (*f)(double x))
{ //求定积分
  int n=1000, i;
  double h, x, s=0.0;
  h=(b-a)/n;
  for(i=1;i<=n;i++) {
    x=a+(i-1)*h;
    s=s+(f(x)+f(x+h))*h/2; //调用f函数求f(x)、f(x+h)
  }
  return s;
}
```

```
15 double f1(double x)
16 { return 1+x;
17 }
18 double f2(double x)
19 { return exp(-x*x/2);
20 }
21 double f3(double x)
22 { return x*x*x;
23 }
24 int main()
25 {
26   double a,b;
27   cin>>a>>b;
28   cout<<(integral(a,b,f1)+integral(a,b,f2)+integral(a,b,f3))<<end
l;
29   return 0;
30 }
```

C程序设计