

## 1.2 算法及其描述

### 1.2.1 什么是算法

数据元素之间的关系有逻辑关系和物理关系，对应的运算有基于逻辑结构的运算描述和基于存储结构的运算实现。

通常把基于存储结构的运算实现的步骤或过程称为算法。



## 算法的五个重要的特性

- (1) 有穷性：在有穷步之后结束，算法能够停机。
  - (2) 确定性：无二义性。
  - (3) 可行性：可通过基本运算有限次执行来实现，也就是算法中每一个动作能够被机械地执行。
  - (4) 有输入
  - (5) 有输出
- } 表示存在数据处理

【例1-2】考虑下列两段描述，这两段描述均不能满足算法的特性，试问它们违反了哪些特性？

(1) 描述一

```
void exam1()
{
    int n=2;
    while (n%2==0)
        n=n+2;
    printf("%d\n",n);
}
```

其中有一个死循环，违反了算法的有穷性特性。

## (2) 描述二

```
void exam20  
{   int x,y;  
    y=0;  
    x=5/y;  
    printf("%d,%d\n",x,y);  
}
```

其中包含除零错误，违反了  
算法的可行性特性



思考题：

算法和程序有什么不同？

## 1.2.2 算法描述

输入



算法



输出

算法描述的一般格式

返回值 算法对应的函数名(形参列表)

```
{  
    //临时变量的定义  
    //实现由输入参数到输出参数的操作  
    ...  
}
```

函数体

- 返回值：通常为bool类型，表示算法是否成功执行。
- 形参列表：由输入型参数和输出型参数构成。



算法输入



算法输出

## 如何描述输出型参数？

C++语言中提供了一种引用运算符“&”用于描述输出型参数。

### 引用示例

```
int a=10;
```

```
int &b=a;
```



引用

*a*  
*b*



10

两个变量共享内存空间

*a*、*b*同步发生改变

**示例：**设计一个交换两个整数的算法。

编写一个函数swap1(x, y):

```
void swap1(int x, int y)
{   int tmp;
    tmp=x; x=y; y=tmp;
}
```

} 交换形参x和y的值



当执行语句swap1(a,b)时，a和b实参值不会发生了交换。

**分析：**x、y既是输入型参数，也是输出型参数



**改正方法1：**采用指针的方式来回传形参的值，需将上述函数改为：

```
void swap2(int *x, int *y)
```

```
{  int tmp;
```

```
    tmp=*x;    //将x的值放在tmp中
```

```
    *x=*y;    //将x所指的值改为*y
```

```
    *y=tmp;    //将y所指的值改为tmp
```

```
}
```

} 交换形参x和y  
所指向的值

上述函数的调用改为swap2(&a,&b) ⇒ 比较复杂。

**改正方法2：**采用引用型形参 ⇨ 将输出型形参改为引用类型。

```
void swap(int &x, int &y)
```

```
//形参前的“&”符号不是指针运算符
```

```
{   int tmp=x;
```

```
    x=y; y=tmp;
```

```
}
```

} 交换形参x和y的值

当执行语句**swap(a,b)**时，形、实参的匹配相当于：

**int &x=a;** //a为x的引用

**int &y=b;** //b为y的引用

这样，*a*与*x*共享存储空间、*b*与*y*共享存储空间，因此执行函数后*a*和*b*的值发生了交换 ⇨ 简单。

## 普通的参数传递

```
void fun1(int n)
```

```
{   int m=2;
```

```
    fun2(m);
```

```
    printf("%d\n",m);
```

```
}
```

实参

```
void fun2(int x)
```

```
{   x++;
```

```
    printf("%d\n",x);
```

```
}
```

普通形参

实参到形参单向值传递

fun1(m)

fun2(x)

## 引用类型的参数传递

```
void fun1(int n)
```

```
{   int m=2;
```

```
    fun2(m);
```

```
    printf("%d\n",m);
```

```
}
```

实参

```
void fun2(int &x)
```

```
{   x++;
```

```
    printf("%d\n",x);
```

```
}
```

引用型形参

① 实参到形参单向值传递

fun1(m)

fun2(x)

② 形参回传给实参，实参和  
形参同步发生改变

## 描述算法示例

【例1-3】设计一个算法：求一元二次方程 $ax^2+bx+c=0$ 的根。

算法可以采用自然语言、流程图或者表格方式等来描述。

**但是，**一个学习计算机的学生应该使用某种计算机语言来描述算法。本课程采用C/C++语言描述算法。

C++的作用是在描述算法时使用其提供的引用类型！

用C/C++描述如下：

算法框架：

输入：  $a$   $b$   $c$



solution



输出：

● 根个数  
●  $x_1$   $x_2$



```
int solution(float a, float b, float c,  
             float &x1, float &x2)  
{ float d, x1, x2;  
  d=b*b-4*a*c;  
  if (d>0)  
  { x1=(-b+sqrt(d))/(2*a);  
    x2=(-b-sqrt(d))/(2*a);  
    return 2;           //2个实根  
  }  
  else if (d==0)  
  { x1=(-b)/(2*a);  
    return 1;           //1个实根  
  }  
  else                  //d<0的情况  
    return 0;           //不存在实根  
}
```

## 思考题

- ① 在用C/C++语言描述算法时，输入型参数和输出型参数如何设计？
- ② 一个算法只能用C/C++语言中的一个函数描述吗？

——本讲完——