数组array：具有一定顺序关系的若干对象的集合体，组成数组的对象称为该数组的元素，数组可由除void外任意类型构成

n维数组：每个元素有n个下标

数组类型的声明：

数组类型 标识符[常量表达式1] [常量表达式2]…;

使用数组时不能越界,下标必须为整数

特点：物理相邻，逻辑相邻

int a[4]={1,2,3,4};与int a[]={1,2,3,4};等价 int a[4]={1,2,3}

注：当指定数组元素个数小于数组大小时，后面指定的会被赋予默认值

当没有初始化时，静态数组会被赋默认值，动态数组则不赋值；

数组也可被声明为常量，被声明为常量的必须初始化，不可改变。

多维数组元素被全部赋值时，第一维的下标可省略。每维可以用花括号表示。

使用数组元素作为实参时，跟普通实参相同

使用数组作为实参时，实参和形参都应该是数组名，且类型要相同，他们之间传递地址。

实参元素的个数不应该小于形参元素的个数。否则发生越界

对形参的修改会影响到实参

把数组作为参数时，一般不指定第一维的大小，即使指定，也被被忽略

对象数组：元素是对象，不仅有数据成员，还有函数成员

语法：

类名 数组名[常量表达式];

访问共有成员格式：

数组名[下标表达式].成员名

当对象数组元素不同时，定义带形参的构造函数，相同时，定义带默认值的构造函数。…

-------------------------------------------------------------------------------------------------------

指针的好处：使用变量名不方便或者没有变量名可使用时使用指针

传递大量数据时可以减少系统开销，提高效率

指针也是一种数据类型，具有指针类型的变量称为指针变量

语法格式：

数据类型 \*标识符;

\* 指针运算符（解析）获取指针所指向变量的值

& 取地址运算符，用来得到一个对象的地址

初始化赋值：

存储类型 数据类型 \*指针名=初始地址；

赋值语句：指针名=地址;

数组名称是一个不能被赋值的指针，即指针常量

（1）可以声明指向常量的指针：不能通过指针来改变对象值，但可以改变指针的值

（2）可以声明指针类型的常量，此时指针的值不能改变

（3）一般情况下，指针的值指针赋给相同变量的指针。任何对象的指针都可以赋给void 形的指针，通过显示转换，void形指针可以访问任何变量。

指针的赋值运算，关系运算，算术运算

算术运算

\*(p±n)与p[±n]等价；表示指针前后第n个数据类型的地址

指针的算术运算要慎用，可能破坏其他地址的数据或超出程序有效的地址；

指针的关系运算：指向相同数据类型的指针之间进行的关系运算；

指针可以跟整数零进行比较；

数组名就是指针

int array[5];

void f(int p[]);

void f(int p[]);

void f(int \*p);等价

指针数组：该数组的元素都是指针变量，都必须是相同类型的指针；

指针数组的初始化必不可少；

语法格式： 数据类型：数据名[下标表达式]；

用指针作为函数参数

传递连续存储数据的地址可以减少开销，提高效率。

1.使实参和形参指向相同的内存空间，双向传递数据

2.减少函数调用时的数据传递的开销

3.指向函数代码的首地址

注；函数体中不需要通过指针改变所指向对象的内容，应该声明为指向常量的指针 const 数据类型 \*p;

在设计程序时，引用比指针的可读性要好。

指针形函数：返回指针类型，用于返回大量数据

语法格式：

数据类型 \*函数名（参数列表）{

函数体

}

指向函数的指针：指向函数代码的首地址

语法：数据类型 （\*函数指针名）（形参表）；

赋值语法：

函数指针名=函数名；

起别名类型

typedef 数据类型（\*函数指针类型）（参数列表）

函数指针类型 函数指针命名；

对象指针：指向对象的指针；

对象有两种成员，一般只储存数据成员；

语法：类名 \*对象指针名

访问语法：对象指针名->成员名 （\*对象指针名）->成员名

this指针：this指针隐藏在每一个类的非静态成员函数中的特殊指针，它指向被成员函数操作的对象

this指针是类成员函数的一个隐藏参数，类成员地址会被赋予this

对于常成员函数来说，这个隐藏的参数是常指针

\*this 是指该对象

在局部作用域中有类成员相同的成员，可以使用this指针访问类成员

指向类的非静态成员的指针，先声明，赋值引用

类型说明符 类名：：\*指针名；

类型说明符 （类名：：\*指针名）（参数表）

指针名=&类名：：数据成员名；

指针名=&类名：：函数成员名；

只存放相对位置，实例化不能使用

对象名.\*类成员指针名

对象指针名->类成员指针名

指向常成员函数必须明确给出const

void(A::\*p)() const;

(对象名.\*类成员指针名)（参数表）

(对象名->类成员指针名)（参数表）

使用指针访问类的静态数据成员

数据类型 \* 指针名=&类名：：静态对象名

返回类型 (\*指针名)(参数表)=类名：：静态函数名

-------------------------------------------------------------------------------------------------------

动态内存分配（在堆中分配）建立和删除

new数据类型（初始化列表）

delete 指针名 同一内存空间不能多次使用

内存泄露：程序占用的内存越来越大；

对于基本数据类型，括号可以省略，不初始化；不省略括号，默认值初始化

有用户定义的默认构造函数，new T和new T()效果相同

无用户定义的默认构造函数，newT 和newT()效果不同，newT()会为基本数据类型和指针赋默认值；

new 类型名[数组长度]() delete[] 指针名 加不加()在于怎么初始化

动态分配多维数组

new 类型名 T[维数] [维数][维数]…;

其中第一维为结果为正整数的表达式，其他维必须为常量正整数表达式

返回类型为指向数组指针；