**数组声明**：类型 数组名[数组大小]，其中数组大小只能是整型常数，不能是变量。数组索引从0开始编号。只有在定义的时候可以初始化数组，也不能将一个数组赋值给另一个数组，使用大括号初始化。

C++11的数组初始化：

1)省略等号，double earn[4]{1.2, 1.3, 1.4, 1.5};

2) 大括号可以为空，默认初始化为0，float balances[100] {};

字符串以空字符\0结尾。双引号初始化字符串常量，空字符被隐式包含，如char bird[] = “Mr Cheeps”。注意，字符串常量与字符常量不能互换，如”S”表示字符S和\0,且”S”表示的是字符串所在的内存地址和’S’表示字符。因此语句char size=”S”是错误的。

任何两个由**空白**(空格、tab和换行)分隔的字符串常量都将自动拼接成一个。拼接时不会被连接的字符串中添加空格。

在#include <cstring>中包含函数strlen()，可以计算字符串的长度，但不包括空字符。strcpy(dest,source)复制字符串,strncpy()可以接受第三个参数，控制复制字符串的长度。

输入字符串：cin使用**空白**(空格、tab和换行)来确定字符串的结束位置，如输入New York时，只能读入New。所以用cin.getline()或cin.get()来读取一行数据。getline()通过回车来确定结尾,有两个参数，name和长度，如cin.getline(name,20)最多读取19个字符，保存在name数组中。cin.get()会保留最后的换行符，cin.get()可以读取下一个字符，通常用来处理换行符。如cin.get(name,20).get()。getline使用简单一些，但get更利于检查错误。

另一个错误：

int year;

cin>>year;

char address[80];

cin.getline(address,80);

无法读取address，因为cin将**回车**留在**输入队列**中，后面的getline认为是空行，将空字符串赋给address。解决方法：(cin>>year).get()。

**String类**

需要包含#include <string>

string str1； //创建实例

string类包含在名称空间std中，自动处理字符串的大小，更方便安全。更多string类用法参考P83

getline(cin,str)读取输入，赋值给string类str。

#include <>

**结构**

定义：

struct inflat

{

char name[20];

float volumn;

double price;

};

创建变量：

inflat hat;

inflat woopie;

通过’.’来访问,如hat.volumn。

结构声明可以放在main函数内(内部声明)或main函数外(外部声明)，内部声明只能再main函数内使用，外部声明可以供所有的函数使用。提倡外部结构声明，内部变量声明。

**共同体union**

用法和struct类似，但只存储一个定义的类型，可以节省内存。

**枚举enum** re

**指针**

定义变量：int data;

变量地址： &data

定义指针： int\* p; //这里p是指针,这里\*左右的空格是可选的

给指针赋值： p = &data;

指针处的值： data = \*p;

对每个指针都需要一个\*： int\* p1,p2; //表示指针p1和int变量p2

**注意**：C++再创建指针时只分配了存储地址的内存，而没有分配存储指针所指向的数据的内存，需要给数据提供空间，否则容易出错，一定要在对指针应用解除引用符(\*)之前，将指针初始化为一个确定的、适当的地址，这是使用指针的金科玉律。不能直接将整型地址赋给指针，如：

int \* pt;

pt = 0xB8000000; //是错的

**用new分配内存**，

int \* pn = new int;

int higgens;

int \* pt = &higgens;

第一种情况只能通过指针pn来访问，第二种情况可以通过指针和变量两种方法来访问。

**用delete来释放内存**，

int \* pn = new int;

delete pn;

delete和new要配对使用。

**分配和访问动态数组**

int\* ps = new int [10]; //给一个数组分配内存，并指向他

delete [] ps; //注意方括号，表示删除数组

对于分配的数组，可以把指针当初数组名来访问，如ps[0], ps[2]。

获取数组的地址，两种方法：

double wages[3];

short stacks[3];

double\* pw = wages; //数组名即是地址

short\* ps = &stacks[0]; //数组首位的地址

stacks[1]等价于\*(stacks+1)。可以修改指针的值，但数组名是常量，不能修改。

在cout和多少C++表达式中，char数组名，char指针和引号括起来的字符串常量都被解释位字符串第一个字符的地址。

给cout提供一个指针，它将打印地址，但如果指针类型位char\*，则cout会打印指向的字符串。应用strcpy来复制字符串，而不是赋值运算符来将字符串赋值给数组。

**动态**表示内存是在运行时，而不是在编译时分配的。

用new创建动态结构，

inflatable \* ps =new inflatable;

但访问成员时不能用点运算符(.)，而应该用箭头运算符(->):如ps->price;或(\*ps).price。

自动存储，静态存储和动态存储

自动存储：在函数内部定义的常规变量，函数结束时自动被释放。

静态存储：在整个程序执行期间都存在。定义方式有两种，1：在函数外面定义；2：使用关键字static。

动态存储：使用new和delete来管理内存，被称为自由存储空间(free store)或堆(heap)。

模板类vector：

包含#include <vector>,包含在名称空间std中。

定义：vector<int> vi;vector<double> vd(n);等。