# 函数的使用技巧

1.重复使用的操作，比如输入输出一些结构类似的数据。可以不必重复写输入输出的代码，减少代码长度。

2.编译系统是根据相同的参数名，而不是参数顺序来决定传递关系的。所以原则上，各个参数以及函数的书写顺序任意，但是习惯上应该对应

# 数组的使用技巧

1.对于不确定元素数的数组，可以先定义一个元素类型的指针，然后通过new指令建立数组（数组的长度可以用变量表达式），并将首地址赋给指针，则指针就可以当成数组名使用。

如：p=new RGB[r\*c]

1.1对于空长度的数组a[]，在没有给数组全部元素赋值之前，没法给数组分配空间，使用情况不同

# 数组初始化

C++中全局数组，全局类中的数组，或静态数组会被自动初始化，一般设为0或’/0’；局部数组，比如函数中的数组不会被自动初始化；对数组进行部分初始化后，未被初始化的元素自动设为0。注意无法使用一般初始化的方式对数组批量赋非零的初值，除非一一赋值。实现这个目的可以使用向量vector代替数组。

# 数据定义

1.全局常量用全大写的字母表示，在程序内部的引用也是大小写敏感的

# 数据转换

1.字符串和数字之间的转换

使用stringstream最佳

Example：

// using stringstream constructors.

#include <iostream>

#include <sstream>

using namespace std;

int main () {

int a;float b;long c;

char d[10],e[10],f[10];

/\* string to number \*/

stringstream ss;

ss << "32";

ss >> a;

ss.clear();

ss << "3.1415";

ss >> b;

ss.clear();

ss << "567283";

ss >> c;

ss.clear();

cout<<a<<","<<b<<","<<c<<endl;

/\* number to string \*/

ss << a;

ss >> d;

ss.clear();

ss << b;

ss >> e;

ss.clear();

ss << c;

ss >> f;

ss.clear();

cout<<d<<","<<e<<","<<f<<endl;

return 0;}

Output:

32,3.141500,567283

32,3.141500,567283