1、如何识别string类型读取完毕，与char \* 字符数组结束符\0的区别；

1.cin >> s; // 遇空格中止

从标准输入读取 string 并将读入的串存储在 s 中。string 类型的输入操

作符：

• 读取并忽略开头所有的空白字符（如空格，换行符，制表符）。

• 读取字符直至再次遇到空白字符，读取终止。

Eg.

string s1, s2;

cin >> s1 >> s2; // read first input into s1, second into s2

cout << s1 << s2 << endl; // write both strings

input:Hello world

output:Helloworld//s1=Hello s2=World

2.getline//读取整行

getline 函数从输入流的下一行读取，并保存读取的内容到不包括换行符。和输入操作符不一样的是，getline 并不忽略行开头的换行符。只要 getline 遇到换行符，即便它是输入的第一个字符，getline 也将停止读入并返回。如果第一个字符就是换行符，则 string 参数将被置为空 string。

int main()

{

string line;

// read line at time until end-of-file

while (getline(cin, line))

cout << line << endl;

return 0;

}

3.直接双引号赋值，可以含空格

2、联合类型，对所有成员赋值之后，是否会存在覆盖情况；

共同体变量所占内存长度是各最长的成员占的内存长度。

共同体每次只能存放哪个的一种！！

共同体变量中起作用的成员是最后一次存放的成员，在存入新的成员后原有的成员失去了作用！

一次只能给联合类型的一个成员赋值，当向联合变量的另一个成员赋值时，其原有的值将被覆盖。（覆盖是按照起始地址开始按内存大小开始存储）

union myun

{

struct { int x; int y; int z; }u;

int k;

}a;

int main()

{

a.u.x =4;

a.u.y =5;

a.u.z =6;

a.k = 0;

printf("%d %d %d\n",a.u.x,a.u.y,a.u.z);

return 0;

}

union类型是共享内存的，以size最大的结构作为自己的大小，这样的话，myun这个结构就包含u这个结构体，而大小也等于u这个结构体的大小，在内存中的排列为声明的顺序x,y,z从低到高，然后赋值的时候，在内存中，就是x的位置放置4，y的位置放置5，z的位置放置6，现在对k赋值，对k的赋值因为是union，要共享内存，所以从union的首地址开始放置，首地址开始的位置其实是x的位置，这样原来内存中x的位置就被k所赋的值代替了，就变为0了，这个时候要进行打印，就直接看内存里就行了，x的位置也就是k的位置是0，而 y，z的位置的值没有改变，所以应该是0,5,6

main()

{

union{

int i;

struct{

char first;

char second;

}half;

}number;

number.i=0x4241;

printf("%c%c\n", number.half.first, number.half.second); -----AB, A的ascii码41H

number.half.first='A';

number.half.second='B';

printf("%x\n", number.i); -------4241

printf("%c%c\n", number.half.first, number.half.second); -----AB, A的ascii码41H

getch();

}

3、不同指针类型不能相互赋值，why？

C语言是强类型语言  
编译器要求不同类型的变量就是不能互相赋值，在编译期就给出错误可以防止编写代码失误引入的bug  
除非你加上强转，明确告之编译器你的意图

5、extern如何使用？

// 1.cpp   
int x = 10;   
// 2.cpp 注意没有包含1.cpp   
#include <iostream>   
using namespace std;   
extern int x;  
int main ()   
{ cout << x << endl; }   
//则输出10

Extern 在源程序外使用时，必须是全局变量，且不是static类型

若为const 类型，则在源程序定义中就要说明为extern类型 extern const int i