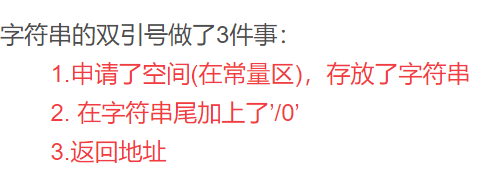
**数组和字符数组**

**数组定义方式：**

Int a[] = {1,2,3}或int a[3] = {1,2,3}，但是不能int a[3](声明); a = {1,2,3}(初始化)，因为声明时就分配好了地址，而a是一个指向该地址的指针常量，不能使a={1,2,3}。



因此可以char \*p = “aaaa”但是不能char \*p = {‘a’,’a’,’a’,’a’,’\0’}，因为数组{…}并不能返回地址。

另外char \*p = “aaaa” 中的p是指向字符串常量区的，不能够修改此地址的内容

而 char p[] = “aaaa” 中的p是指向栈区的，是可以修改的。

**文件读写**

**mode:**

“r”只能读

”w”只能写，无此文件则创建，有则清空

“a”只能写，无此文件则创建，有则在后面追加写

“r+”可读可写文件，无此文件不能创建

“w+”可读可写，无此文件则创建，有则清空

“a+”可读可写，无此文件则创建，有则再后面追加写

**写文件：**

fputc(c,fp)写入字符c，fputs(s,fp)写入字符串s，fprintf(s,fp)与fputs一样。

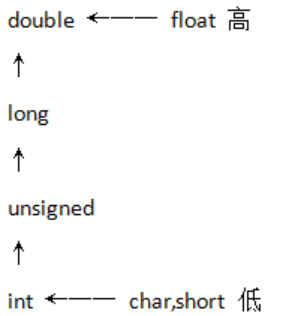
**读文件：**

fgetc(fp)读取fp指向的文件中的下一个字符并返回；

fgets(s,n,fp)读取fp指向的文件中的接下类n-1个字符并保存到字符串s中，但是当未读到n-1个字符便遇到换行符文件末尾，则停止读取；

fscanf(fp,n,s)与fgets功能一样，但是在遇到空格或换行符时则停止读取。

**类型转换**



**可变参数**

* 引入头文件stdarg.h；
* 定义函数时func(int num,…)，num为参数数量，…为具体参数；
* 在函数内先定义一个va\_list变量valist(va\_list类型在头文件中定义的)；
* 使用va\_start(valist,num)初始化参数列表va\_start也是在头文件中定义的；
* 使用va\_arg(valist,int)来访问valist指针指向的参数，int表示该参数的类型，之后将指针指向下一个参数；
* 最后要使用va\_end清理valist的内存。

**关于函数返回**

一般的来说，函数是可以返回局部变量的。 局部变量的作用域只在函数内部，在函数返回后，局部变量的内存已经释放了。因此，如果函数返回的是局部变量的值，不涉及地址，程序不会出错。但是如果返回的是局部变量的地址(指针)的话，程序运行后会出错。因为函数只是把指针复制后返回了，但是指针指向的内容已经被释放了，这样指针指向的内容就是不可预料的内容，调用就会出错。准确的来说，函数不能通过返回指向栈内存的指针(注意这里指的是栈，返回指向堆内存的指针是可以的)