C 语言知识点

const 与 * 操作符

答：　（const 在*的左边）说明：不可以通过指针 b 来修改 a 的值（但可以通过其他方式改变 a 的值，比如 a 自增一；同时 b 的指向可以修改）

（const 在*的右边）说明：b 仅仅可以指向 a,而不可以指向其他对象（但 a 的值，既可以通过 b 改变，也可以通过其他方式改变，比如自增一

break 的作用

答：　跳出最近的循环语句或 switch 语句（注：跟 if 语句没关系！）

限制对于函数的访问

答：　函数名前加 static 说明：函数仅在这个文件内可见

函数名前加 extern 说明：函数在任何地方都可见

关于 struct，union 和 enum

答：　struct：内部各类型用“分号”分隔

union：内部各类型用“分号”分隔，各部分共用存储空间

enum：内部各名字用“逗号”分隔，作用：将一串名字与一串”整型“值联系在一起。只可以使用声明时使用的名字对其赋值

声明的优先级规则

答：　A：　声明从它的名字开始读取，然后按照优先级顺序依次读取

B：　优先级从高到底依次是：

B.1:　声明中被括号括起来的那部分

B.2:　后缀操作符：

括号( )表示这是一个函数，而

方括号[ ]表示这是一个数组。

B.3　前缀操作符：

星号*表示“指向...的指针”。

C:　如果 const（和/或）volatile 关键字的后面紧跟类型说明符（如 int，long 等），那么它作用于类型说明符。在其他情况下，const（和/或）volatile 关键字作用于它左边紧邻的指针星号

例：

char * const * (*next)( );

读做：“next 是一个指向函数的指针，该函数返回另一个指针，该指针指向一个只读的指向 char 的指针”。

typedef 和#define 的区别

答：　A：可以用其他类型说明符对宏类型名进行扩展，但对 typedef 所定义的类型名却不能这样做。

例：　#define peach int

unsigned peach i; /*没问题*/

typedef int banana;

unsigned banana i; /*非法*/

B：在连续几个变量的声明中，用 typedef 定义的类型能够保证声明中所有的变量均为同一种类型，而#define 定义的类型则无法保证。

例：　#define int_ptr int*

in_ptr peach, banana; /*peach 类型为指针，banana 类型为 int*/

typedef 用在哪？

答：　应该用在：A：数组/结构/指针以及函数的组合类型。

B：可移植类型。

C：为强制类型转换提供简单的名子

不建议用：结构体定义