# C语言综合研究与高强度程序设计训练16

## 1

从面向对象程序设计的角度来看，a.c和b.c的ch型数据（对象）的封装性那个更  
好？

* a.c
* #define screen ((char far \*)0xb8000000)  
    
  typedef struct c {  
   char chr;  
   char color;  
   void (\*put)(struct c \*, int, int);  
  } ch;  
    
  void f(ch \*, int, int);  
    
  main() {  
   int n;  
   ch a;  
    
   a.chr = 'c';  
   a.color = 2;  
   a.put = f;  
   a.put(&a, 10, 40);  
  }  
    
  /\* 函数，在屏幕的row行和col列打印字符，row、co1从1开始编号。＊ \*/  
  void f(ch \*p, int row, int col) {  
   screen[(row - 1) \* 160 + (col - 1) \* 2] = p->chr;  
   screen[(row - 1) \* 160 + (col - 1) \* 2 + 1] = p->color;  
  }
* b.c
* #define screen ((char far \*)0xb8000000)  
    
  typedef struct c {  
   char chr;  
   char color;  
   void (\*setch)(struct c \*, char);  
   void (\*setcolor)(struct c \*, char);  
   void (\*put)(struct c \*, int, int);  
  } ch;  
    
  void f(ch \*, int, int);  
  void f1(ch \*, char);  
  void f2(ch \*, char);  
    
  main() {  
   int n;  
   ch a;  
    
   a.put = f;  
   a.setch = f1;  
   a.setcolor = f2;  
    
   a.setch(&a, 'c');  
   a.setcolor(&a, 2);  
   a.put(&a, 10, 40);  
  }  
    
  void f(ch \*p, int row, int col) {  
   screen[(row - 1) \* 160 + (col - 1) \* 2] = p->chr;  
   screen[(row - 1) \* 160 + (col - 1) \* 2 + 1] = p->color;  
  }  
    
  void f1(ch \*p, char a) { p->chr = a; }  
  void f2(ch \*p, char color) { p->color = color; }
* 通过对比以上两个代码可以看出b,c的封装性更强。

## 2

* 补全下面程序中宏 new 的实现
* c.c
* #define screen ((char far \*)0xb8000000)  
    
  #define new(x) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
    
  typedef struct c {  
   char chr;  
   char color;  
   void (\*put)(struct c \*, int, int);  
  } ch;  
  void f(ch \*, int, int);  
    
  main() {  
   int n;  
    
   ch \*a = new(ch);  
    
   a->chr = 'c';  
   a->color = 2;  
   a->put = f;  
   a->put(a, 10, 40);  
  }  
    
  /\* 函数，在屏幕的row行和col列打印字符，row、co1从1开始编号。＊ \*/  
  void f(ch \*p, int row, int col) {  
   screen[(row - 1) \* 160 + (col - 1) \* 2] = p->chr;  
   screen[(row - 1) \* 160 + (col - 1) \* 2 + 1] = p->color;  
  }
* 首先通过malloc分配内存然后使程序可以正常运行，然后将malloc对应的语句改写成宏定义
* #define screen ((char far \*)0xb8000000)  
    
  #define new(x) (x\*)malloc(sizeof(x))  
    
  typedef struct c {  
   char chr;  
   char color;  
   void (\*put)(struct c \*, int, int);  
  } ch;  
  void f(ch \*, int, int);  
    
  main() {  
   int n;  
    
   ch \*a = new(ch);  
   /\*ch \*a = (ch\*)malloc(sizeof(ch));\*/  
    
   a->chr = 'c';  
   a->color = 2;  
   a->put = f;  
   a->put(a, 10, 40);  
  }  
    
  /\* 函数，在屏幕的row行和col列打印字符，row、co1从1开始编号。＊ \*/  
  void f(ch \*p, int row, int col) {  
   screen[(row - 1) \* 160 + (col - 1) \* 2] = p->chr;  
   screen[(row - 1) \* 160 + (col - 1) \* 2 + 1] = p->color;  
  }
* 验证结果
* 