5.12 有一个首地址为 MEM 的 100D 字数组，试编制程序删除数组中所有为零的项，并将后续项向前压缩，最后将数组剩余部分补上零。

源代码如下：

数据段：定义一个 0 开头后跟 98 个 99 和一个 1

```
DATAS SEGMENT
    ;此处输入数据段代码
    MEM DW 0
        DW 98 DUP(99)
        DW 1
    string1 db 'before:',0ah,0dh,'$';$终止符
    string2 db 0ah,0dh,'after:',0ah,0dh,'$';$终止符
DATAS ENDS
```

代码段：

```
CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS,SS:STACKS
START:
    MOV AX,DATAS
    MOV DS,AX
    mov dx,offset string1;'sum:'
    mov ah,9
    int 21h
    mov si, offset MEM  ;传入数组初始地址
    mov cx,100
    call output

    mov bx,0
    mov si,(100-1)*2; 末尾地址
first_:
    cmp MEM[bx],0
    jz  delete
    add bx,2
    loop first_
    jmp print
```

```
delete:;执行删除
    mov di,bx
loop_in:
    cmp di,si
    jnb next_
    mov ax,MEM[di]
    xchg ax,MEM[di+2]
    xchg MEM[di],ax
    add di,2
    jmp loop_in

next_:
    loop first_
print:
    mov si, offset MEM  ;传入数组初始地址
    mov cx,100
    mov dx,offset string2;'sum:'
    mov ah,9
    int 21h
    call output

    MOV AH,4CH
    INT 21H
```

堆栈段：

```
STACKS SEGMENT
    db 40 dup(0) ;此处输入堆栈段代码
STACKS ENDS
```

子程序：

用于打印结果：

```
output proc near
    push cx
    push ax
    push bx
    push dx;
    push si;
    ;mov  dx,offset string;'sum:'
    ;mov ah,9
    ;int 21h
    mov bx,10;设置被除数
initial:
    push cx;
    mov cx,0  ;设置被除次数计数，即数字的十进制位数
    mov ax,[si];设置被除的数
    ;mov ah,0
first:  ;单个数字输出，每次输出一个余数
    mov dx,0
    div bx; 余在dx,商在ax
    push dx
    inc cx
    cmp ax,0  ;判断商是否为0是否除尽
    ;mov ah,0
    jnz first
```

```
second:
    pop dx
    or dx,30H
    dec cx   ;
    ;push dx  ;压入保存
    ;mov dl,ah  ;赋值准备打印
    mov ah,2
    int 21h
    ;pop cx  ;弹出
    cmp cx,0 ;判断当前的数字位数是否输出完毕
    jnz second


next:
    call space
    add si,2;
    pop cx;
    loop initial

   ; call newline
    pop si
    pop dx
    pop bx
    pop ax
    pop cx
    ret
output  endp
```

用于打印空格：

```
space proc near
      push ax
      push dx
      mov  dl,20h
      mov ah,2
      int 21h
      pop dx
      pop ax
    ret
space endp

CODES ENDS
    END START
```

结果截图：

```
before:
0 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99
99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99
 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 9
9 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 1
after:
99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99
 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 9
9 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99
99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 99 1 0
```

5.19 已知数组 A 包含 15 个互不相等的整数，数组 B 包含 20 个互不相等的整数。试编制一程序，把既在 A 中又在 B 中出现的整数存放于数组 C 中。

源代码如下：

数据段：A 中为 0~30 的偶数，B 为 1~20

```
DATAS SEGMENT
    ;此处输入数据段代码
    array_A DW 2,4,6,8,10,12,14,16,18,20,22,24,26,28,30
    array_B DW 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
    array_C DW 15 DUP(?)
DATAS ENDS
```

代码段：

主要功能完成。

```
CODES SEGMENT
    ASSUME CS:CODES,DS:DATAS,SS:STACKS
START:
    MOV AX,DATAS
    MOV DS,AX
    ;此处输入代码段代码
    mov bx,0
    mov cx,15;记录A数组元素数目
    mov si,0;用于遍历A
out_loop:
    mov di,0;用于遍历B
    mov dx,20
    mov ax,array_A[si]
compare:
    cmp ax,array_B[di]
    jnz not_e
    jmp equal
not_e:
    add di,2
    dec dx
    jnz compare;B没有遍历完继续比对
    jmp next_
equal:
    mov array_C[bx],ax
    add bx,2
next_:
    add si,2
    loop out_loop
```

调用打印

```
    mov si,offset array_C
    mov cx,15
    call output
    MOV AH,4CH
    INT 21H
```

堆栈段：

```
STACKS SEGMENT
    db 40 dup(0) ;此处输入堆栈段代码
STACKS ENDS
```

子程序：
用于打印结果：

```
output proc near
    push cx
    push ax
    push bx
    push dx;
    push si;
    ;mov  dx,offset string;'sum:'
    ;mov ah,9
    ;int 21h
    mov bx,10;设置被除数
initial:
    push cx;
    mov cx,0  ;设置被除次数计数，即数字的十进制位数
    mov ax,[si];设置被除的数
    ;mov ah,0
first:  ;单个数字输出，每次输出一个余数
    mov dx,0
    div bx; 余在dx,商在ax
    push dx
    inc cx
    cmp ax,0  ;判断商是否为0是否除尽
    ;mov ah,0
    jnz first
```

```
second:
    pop dx
    or dx,30H
    dec cx   ;
    ;push dx  ;压入保存
    ;mov dl,ah  ;赋值准备打印
    mov ah,2
    int 21h
    ;pop cx  ;弹出
    cmp cx,0 ;判断当前的数字位数是否输出完毕
    jnz second


next:
    call space
    add si,2;
    pop cx;
    loop initial

   ; call newline
    pop si
    pop dx
    pop bx
    pop ax
    pop cx
    ret
output  endp
```

用于打印空格：

```
space proc near
      push ax
      push dx
      mov  dl,20h
      mov ah,2
      int 21h
      pop dx
      pop ax
    ret
space endp

CODES ENDS
    END START
```

结果截图：

```
2 4 6 8 10 12 14 16 18 20 0 0 0 0 0
```

为 0 的是数组 C 初始化设值。