5.2

（1）直接寻址 mov ax, x+2

等价写法： mov ax, x[2]

mov ax, [x+2]

(2) 寄存器间接寻址

lea ebx, x+2

mov ax, [ebx]

；ebx 可以换成esi,edi, ecx, 等等寄存器，但不建议换成eax,因为结果在 ax 中

等价写法1 ： mov ebx, offset x+2

mov ax, [ebx]

等价写法2： Mov ebx, offset x or lea ebx, x

add ebx, 2

mov ax, [ebx]

(3) 变址寻址

mov ebx, 1

mov ax, x[ebx\*2]

等价写法： mov ebx, offset x

Mov ax, [ebx +2]

(4) 基址加变址寻址

mov ebx, 0

mov esi, 1

mov ax, x[ebx + esi\*2]

等价写法： mov ebx, offset x

Mov esi, 2

Mov ax, [ebx+esi]

还有很多等价写法。

注意：Win32 下编程，地址是 32位的，出现在 方括号中的寄存器 一定是 32 位的通用寄存器，不要使用 16位的寄存器。在16位寻址方式下，对寄存器使用有更多的约束，很容易出现语法错误的。

5.3

(1) cmp ax, 0

Js L 等价 Jl L

(2) test ax, 8000H

JNZ L

注意， ax 是16位的寄存器，不能用 test ax, 80H;

也不能用 test ax， 1000H， 它对应的是 0001 0000 0000 0000B

(3) or ax, ax 等价 ax, 0

Js L

(4) shl ax, 1

Jc L

(5) rol ax, 1

Jc L

5.4

(1) cmp ax, 0

Jge l1

neg AX ; 后面的写法，仅替换此处的语句

L1:

(2) IMUL Ax, -1

(3) MOV bx, ax

Mov ax, 0

sub ax, bx

(4) not ax

Inc ax

5.5

（1） add ax, ax

(2) imul ax, 2

(3) shl ax, 1