stack segment

dw 128 dup(?)

stack ends

code segment

assume cs:code,ds:code,ss:stack

ARY1  DB  14,-25,66,-17,78,-33,46,-96,71,80

LEN EQU   $ - ARY1

ODDNUM DB 0

EVENNUM DB 0

main:

mov ax,stack

mov ss,ax

mov sp,128

mov ax,code

mov ds,ax

LEA BX,ARY1

XOR ax,ax

MOV CX,LEN

L1:

  TEST[BX],01H

  JNZ L2

  JMP L3

L2: INC AL

L3: inc bx

LOOP  L1

MOV ODDNUM,AL

push  ax

mov al,LEN

pop bx

sub al,bl

mov EVENNUM,al

code ends

end main

stack segment ; 堆栈段开始

dw 128 dup(?) ; 定义大小为128字节的堆栈

stack ends ; 堆栈段结束

code segment ; 代码段开始

assume cs:code,ds:code,ss:stack ; 定义代码段、数据段和堆栈段的偏移地址

ARY1 DB 14,-25,66,-17,78,-33,46,-96,71,80 ; 定义一个包含10个元素的数组，这些元素的值为14、-25、66、-17、78、-33、46、-96、71和80

LEN EQU $ - ARY1 ; 计算数组的长度

ODDNUM DB 0 ; 定义ODDNUM变量并初始化为0

EVENNUM DB 0 ; 定义EVENNUM变量并初始化为0

main: ; 程序入口

mov ax,stack ; 将堆栈段的地址移动到AX寄存器中

mov ss,ax ; 将AX中的值移动到SS寄存器中，该寄存器指向堆栈段的开头

mov sp,128 ; 将堆栈指针设置为堆栈段的末尾

mov ax,code ; 将代码段的地址移动到AX寄存器中

mov ds,ax ; 将AX中的值移动到DS寄存器中，该寄存器指向代码段的开头

LEA BX,ARY1 ; 将BX设置为ARY1数组的偏移地址

XOR ax,ax ; 将AX寄存器清零

MOV CX,LEN ; 将CX寄存器设置为ARY1数组的长度

L1:

TEST[BX],01H ; 测试当前元素是否为奇数，如果是则跳转到L2，否则跳转到L3

JNZ L2

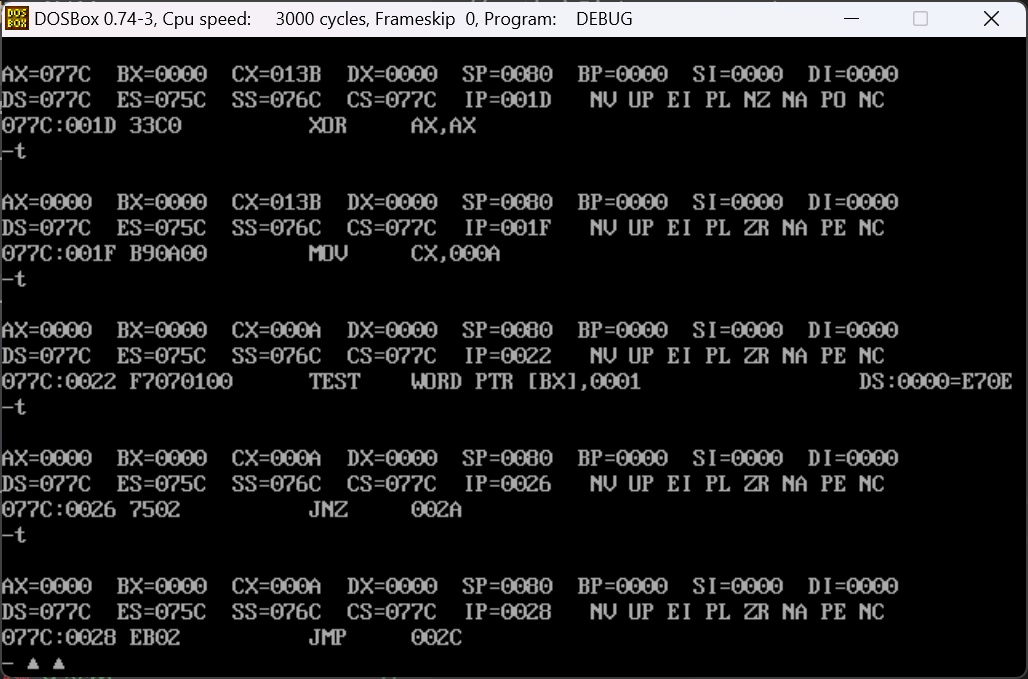
JMP L3

L2:

INC AL ; 将AL寄存器加1，表示找到了一个奇数

L3:

inc bx ; 将BX寄存器加1，指向下一个元素



LOOP L1 ; 循环处理所有元素

MOV ODDNUM,AL ; 将AL寄存器中奇数数量的值存储在ODDNUM变量中

push ax ; 将AX寄存器压入堆栈中

mov al,LEN ; 将AL寄存器设置为ARY1数组的长度

pop bx ; 弹出AX寄存器并将其保存在BX寄存器中

sub al,bl ; 计算偶数数量并将其保存在AL寄存器中

mov EVENNUM,al ; 将AL寄存器中偶数数量的值存储在EVENNUM变量中

code ends ; 代码段结束

end main ; 程序结束标记