# 汇编语言第3次上机

班级	学号	姓名
计算机2205	2204112913	李雨轩

# 1. 子程序设计

设有10个学生的成绩分别是76,69,84,90,73,88,99,63,100和80分。试编制一个子程序统计60~69分,70~79分,80~89分,90~99分和100分的人数并分别存放到56,57,58,59和510单元中。数据段中至少需要定义以下内容:

- (1) ID db '2186123456'(说明:以学号2186123456为例,此处应更换为自己的学号)
- (2) array db ... (存放10个学生的成绩) (3) 定义S6, S7, S8, S9, S10的内存单元

#### 1.1 反汇编的截图

```
3000 cycles, Frameskip 0, Program:
LINK : warning L4021: no stack segment
   Generate score.exe successfully.
   debug score.exe ...
0780:0000 1E
                      PUSH
                              DS
0780:0001 3300
                      XOR
                              AX,AX
0780:0003 50
                      PUSH
                              ΑX
0780:0004 B87E07
                      MOV
                              AX,077E
0780:0007 8ED8
                      MOV
                              DS,AX
0780:0009 BF0A00
                      MOV
                              DI,000A
0780:000C BE1400
                      MOU
                              SI,0014
0780:000F BA0A00
                      MOV
                              DX,000A
0780:0012 E80100
                              0016
                      CALL
0780:0015 CB
                      RETF
                              CX,DX
0780:0016 8BCA
                      MOV
0780:0018 8A05
                      MOV
                              AL,[DI]
0780:001A 3C64
                      CMP
                              AL,64
0780:001C 7413
                      JZ
                              0031
0780:001E 3C5A
                      CMP
                              AL,5A
```

DOSBox 0.74-3, Cpu speed:	2 1	1 , 3	DEBUG		_	×
0780:0012 E80100	CALL	0016				
0780:0015 CB	RETF	OU NU				
0780:0016 8BCA	MOV	CX,DX				
0780:0018 8A05	MOV	AL,[DI]				
0780:001A 3C64	CMP	AL,64				
0780:001C 7413	JZ	0031				
0780:001E 3C5A −u	CMP	AL,5A				
0780:0020 7D16	JGE	0038				
0780:0022 3050	CMP	AL,50				
0780:0024 7D19	JGE	003F				
0780:0026 3046	CMP	AL,46				
0780:0028 7D1C	JGE	0046				
0780:002A 3C3C	CMP	AL,3C				
0780:002C 7D1F	JGE	004D				
0780:00ZE EB21	JMP	0051				
0780:0030 90	NOP					
0780:0031 FE061800	INC	BYTE PTR	[0018]			
0780:0035 EB1A	JMP	0051				
0780:0037 90	NOP					
0780:0038 FE061700	INC	BYTE PTR	[0017]			
0780:003C EB13	JMP	0051				
0780:003E 90	NOP					
0780:003F FE061600	INC	BYTE PTR	[0016]			
_						
	2000	11. 0.5				 
DOSBox 0.74-3, Cpu speed:		skip 0, Program:	DEBUG		_	×
0780:0037 90	NOP				_	X
0780:0037 90 0780:0038 FE061700	NOP INC	BYTE PTR			_	×
0780:0037 90 0780:0038 FE061700 0780:003C EB13	NOP INC JMP				_	X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90	NOP INC JMP NOP	BYTE PTR 0051	[0017]		_	×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600	NOP INC JMP	BYTE PTR	[0017]		_	×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u	NOP INC JMP NOP INC	BYTE PTR 0051 BYTE PTR	[0017]		_	×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C	NOP INC JMP NOP INC JMP	BYTE PTR 0051	[0017]		_	X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90	NOP INC JMP NOP INC JMP NOP	BYTE PTR 0051 BYTE PTR 0051	[0017] [0016]			X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500	NOP INC JMP NOP INC JMP NOP INC	BYTE PTR 0051 BYTE PTR 0051 BYTE PTR	[0017] [0016]			×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:004A EB05	NOP INC JMP NOP INC JMP NOP INC JMP	BYTE PTR 0051 BYTE PTR 0051	[0017] [0016]			×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:004C 90	NOP INC JMP NOP INC JMP NOP INC JMP	BYTE PTR 0051 BYTE PTR 0051 BYTE PTR 0051	[0017] [0016] [0015]			×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:004C 90 0780:004D FE061400	NOP INC JMP NOP INC JMP NOP INC	BYTE PTR 0051 BYTE PTR 0051 BYTE PTR 0051 BYTE PTR	[0017] [0016] [0015]			×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:004C 90 0780:004D FE061400 0780:0051 47	NOP INC JMP NOP INC JMP NOP INC INC	BYTE PTR 0051 BYTE PTR 0051 BYTE PTR 0051 BYTE PTR DI	[0017] [0016] [0015]			×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:004C 90 0780:004D FE061400 0780:0052 E2C4	NOP INC JMP INC JMP NOP INC JMP INC INC LOOP	BYTE PTR 0051 BYTE PTR 0051 BYTE PTR 0051 BYTE PTR	[0017] [0016] [0015]			×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:0040 FE061400 0780:0051 47 0780:0052 EZC4 0780:0054 C3	NOP INC JMP NOP INC JMP NOP INC JMP NOP INC LOOP RET	BYTE PTR 0051 BYTE PTR 0051 BYTE PTR 0051 BYTE PTR DI 0018	[0017] [0016] [0015]			×
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:0040 FE061400 0780:0051 47 0780:0052 E2C4 0780:0055 4E	NOP INC JMP NOP INC JMP NOP INC INC LOOP RET DEC	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR DI 0018  SI	[0017] [0016] [0015]			X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:004C 90 0780:004D FE061400 0780:0051 47 0780:0052 E2C4 0780:0055 4E 0780:0056 42	NOP INC JMP NOP INC JMP NOP INC JMP NOP INC INC INC INC INC INC INC	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR DI 0018  SI DX	[0017] [0016] [0015] [0014]			X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:004C 90 0780:0051 47 0780:0052 E2C4 0780:0055 4E 0780:0056 42 0780:0057 3030	NOP INC JMP NOP INC JMP NOP INC JMP NOP INC INC INC LOOP RET DEC INC XOR	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR DI 0018  SI	[0017] [0016] [0015] [0014]			X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 FE061500 0780:0040 FE061400 0780:0051 47 0780:0052 E2C4 0780:0055 4E 0780:0056 42 0780:0057 3030 0780:0059 D7	NOP INC JMP NOP INC JMP NOP INC JMP NOP INC INC LOOP RET DEC INC XOR XLAT	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR DI 0018  SI DX [BX+SI],1	[0017] [0016] [0015] [0014]			X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:0040 FE061400 0780:0051 47 0780:0052 E2C4 0780:0054 C3 0780:0055 4E 0780:0056 42 0780:0057 3030 0780:0059 D7 0780:005A 0100	NOP INC JMP NOP INC JMP NOP INC JMP NOP INC INC LOOP RET DEC INC XOR XLAT ADD	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR DI 0018  SI DX [BX+SI],I	[0017] [0016] [0015] [0014]			X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:004A EB05 0780:004D FE061400 0780:0051 47 0780:0052 E2C4 0780:0054 C3 0780:0055 4E 0780:0056 42 0780:0057 3030 0780:005A 0100 0780:005C 0000	NOP INC JMP NOP INC JMP NOP INC JMP NOP INC LOOP RET DEC INC XOR XLAT ADD ADD	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0018  SI DX [BX+SI],6 [BX+SI],6	[0017] [0016] [0015] [0014] DH			X
0780:0037 90 0780:0038 FE061700 0780:0036 EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:0040 PE061400 0780:0051 47 0780:0051 47 0780:0052 E2C4 0780:0055 4E 0780:0055 4E 0780:0056 42 0780:0057 3030 0780:0058 0100 0780:005C 0000 0780:005E 0000	NOP INC JMP NOP INC JMP NOP INC INC LOOP RET DEC INC XOR XLAT ADD ADD	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0018  SI DX [BX+SI],1 [BX+SI],6 [BX+SI],6	[0017] [0016] [0015] [0014]  DH  AX AL AL			X
0780:0037 90 0780:0038 FE061700 0780:003C EB13 0780:003E 90 0780:003F FE061600 -u 0780:0043 EB0C 0780:0045 90 0780:0046 FE061500 0780:0046 EB05 0780:0040 FE061400 0780:0051 47 0780:0052 E2C4 0780:0054 C3 0780:0055 4E 0780:0056 42 0780:0057 3030 0780:0058 0100 0780:0056 0100	NOP INC JMP NOP INC JMP NOP INC JMP NOP INC LOOP RET DEC INC XOR XLAT ADD ADD	BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0051  BYTE PTR 0018  SI DX [BX+SI],6 [BX+SI],6	[0017] [0016] [0015] [0014]  DH  AX AL AL AL			X

1.2 在进行计算前,显示ID、array以及S6——S10的内存值的截图(多显示、少显示均扣分)

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
0780:0046 FE061500
                         INC
                                  BYTE PTR [0015]
0780:004A EB05
                         JMP
                                  0051
0780:004C 90
                         NOP
0780:004D FE061400
                         INC
                                  BYTE PTR [0014]
0780:0051 47
                         INC
                                  DΙ
0780:0052 E2C4
                         LOOP
                                  0018
0780:0054 C3
                         RET
0780:0055 4E
                         DEC
                                  SI
0780:0056 42
                         INC
                                  \mathbf{D}\mathbf{X}
0780:0057 3030
                         XOR
                                  [BX+SI],DH
0780:0059 D7
                         XLAT
0780:005A 0100
                         ADD
                                  [BX+SI].AX
0780:0050 0000
                         ADD
                                  [BX+SI],AL
0780:005E 0000
                                  [BX+SI].AL
                         ADD
0780:0060 0000
                         ADD
                                  [BX+SI].AL
                         ADD
                                  [BX+SI],AL
0780:0062 0000
-g 12
AX-077E BX-0000 CX-027E DX-000A SP-FFFC BP-0000 SI-0014 DI-000A
                   SS=077D CS=0780 IP=0012
DS=077E ES=076E
                                                NU UP EI PL ZR NA PE NC
0780:0012 E80100
                         CALL
                                  0016
-d 077e:0000 0018
077E:0000 32 32 30 34 31 31 32 39-31 33 4C 45 54 5A 49 58
                                                                 2204112913LETZIX
077E:0010 63 3F 64 50 00 00 00 00-00
                                                                 c?dP....
```

# 1.3 执行完计算后,显示ID、array以及S6——S10的内存值的截图(多显示、少显示均扣分

```
BOSBox 0.74-3, Cpu speed:
                   3000 cycles, Frameskip 0, Program: DEBUG
0780:0056 42
                        INC
0780:0057 3030
                        XOR
                                [BX+SI].DH
0780:0059 D7
                        XLAT
0780:005A 0100
                        ADD
                                [BX+SI],AX
0780:0050 0000
                        ADD
                                [BX+SI].AL
0780:005E 0000
                        ADD
                                [BX+SI],AL
0780:0060 0000
                                [BX+SI],AL
                        ADD
0780:0062 0000
                        ADD
                                [BX+SI].AL
-g 12
AX-077E BX-0000 CX-027E DX-000A SP-FFFC BP-0000 SI-0014 DI-000A
DS=077E ES=076E SS=077D CS=0780 IP=0012
                                              NU UP EI PL ZR NA PE NC
0780:0012 E80100
                        CALL
                                0016
-d 077e:0000 0018
077E:0000 32 32 30 34 31 31 32 39-31 33 4C 45 54 5A 49 58
                                                              2204112913LETZIX
077E:0010 63 3F 64 50 00 00 00 00-00
                                                              c?dP....
-g 15
AX=0750 BX=0000 CX=0000 DX=000A SP=FFFC
                                             BP=0000 SI=0014 DI=0014
DS=077E ES=076E
                 SS=077D CS=0780 IP=0015
                                              NU UP EI PL NZ NA PE NC
0780:0015 CB
                        RETF
-d 077e:0000 0018
077E:0000 32 32 30 34 31 31 32 39-31 33 4C 45 54 5A 49 58
                                                              2204112913LETZIX
                                                              c?dP....
077E:0010 63 3F 64 50 02 02 03 02-01
```

### 1.4 源代码

```
1 name Score
2 title Count students score
3
4 MSG MACRO COUNT
5
    S&COUNT DB 0
6 ENDM
7
8
   data segment
9
      ID db '2204112913'
      array db 76, 69, 84, 90, 73, 88, 99, 63, 100, 80
      counts label byte
11
      X = 6
12
      REPT 5
13
14
          MSG %X
15
          X = X + 1
16
      ENDM
17 data ends
18
19 code segment
20
      assume cs:code, ds:data
21
22
      main proc far
23
          push ds
24
           xor ax, ax
25
           push ax
26
           mov ax, seg data
27
           mov ds, ax
28
29
          mov di, offset array
30
           mov si, offset counts
31
          mov dx, counts - array
32
           call countScore
33
34
          ret
35
      main endp
36
37
      countScore proc near
38
          mov cx, dx
39
40 count scores:
           mov al, [di]
41
42
           cmp al, 100
43
44
           je increment S10
45
           cmp al, 90
           jge increment_S9
46
           cmp al, 80
47
48
           jge increment S8
```

```
49
           cmp al, 70
50
           jge increment S7
51
            cmp al, 60
52
            jge increment S6
53
            jmp next student
54
55 increment S10:
56
           inc byte ptr S10
57
            jmp next student
58 increment S9:
59
           inc byte ptr S9
60
            jmp next student
61 increment S8:
62
            inc byte ptr S8
63
            jmp next student
64 increment S7:
65
           inc byte ptr S7
66
            jmp next student
67 increment S6:
68
            inc byte ptr S6
69 next student:
70
           inc di
71
            loop count scores
72
           ret
73
        countScore endp
74
75 code ends
76
      end main
```

# 2. 高级汇编语言技术

试使用条件汇编和重复汇编编写一段程序,完成以下功能:根据给定名为X的字符串长度汇编以下指令,如果X的长度为n,当n<=5时,汇编n次;当n>5时,汇编6次。ADD AX, AX

数据段中至少需要定义以下内容:

- (1) ID db '2186123456'(说明:以学号2186123456为例,此处应更换为自己的学号)
- (2) X 是一个目标字符串,根据下面的测试场景需求进行定义;(3) 从这里定义自己认为必要的变量

#### 测试场景要求:

(1) X 是一个长度为5 的字符串,例如: '23456'; 学号的后5位 (2) X 是一个长度为10的字符串,例如: '2186123456'; 完整的学号 (3) 测试时,只能改变数据段中X的定义内容(通过注释符号),其它数据段、代码段中的内容必须保持不变。

# 2.1 场景1的.lst文件的截图

# ASM3\REPTASM.LST

	osoft (R) Macro Assembler ional Assembly and repiti		4/21/24 23:38:09 Page 1-1	
		name REPTASM title conditional Asse bly	mbly and repitition Assem	
0000 0000	32 32 30 34 31 31 32 39 31 33	data segment ID db '2204112913'		
000A 000A 000F	32 32 30 34 31	strBegin label byt X db '22041' strEnd label byte	е	
000F		data ends		
0000		code segment		
0000		assume cs:code, ds main proc far	:data	
	B8 R	mov ax, seg da	ta	
	8E D8	mov ds, ax		
0005	B8 0005	mov ax, strEnd	-strBegin	
		IF strEnd-strBegin ng is less than or equ REPT strEnd-st add ax, ax ENDM	rBegin	
	03 CO 1	add ax, ax		
	03 C0 1	add ax, ax		
	03 C0 1 03 C0 1	add ax, ax add ax, ax		
	03 CO 1	add ax, ax		
		ENDIF		
	B8 4C00 CD 21	mov ax, 4c00h int 21h main endp		
0017		code ends		
W:		end main	4/91/94 99.20.00	
	osoft (R) Macro Assembler ional Assembly and repiti		4/21/24 23:38:09 Symbols-1	
Segmen	ts and Groups:			
	N a m e	Length Align	Combine Class	
			NONE NONE	
Symbols:				
	N a m e	Type Value	Attr	
ID		. L BYTE 0000	DATA	
MAIN .		. F PROC 0000	CODE Length = 0017	
	IN		DATA DATA	

```
X . . . . . . . . L BYTE 000A DATA

@FILENAME . . . . . . . . TEXT reptASM

35 Source Lines
40 Total Lines
9 Symbols

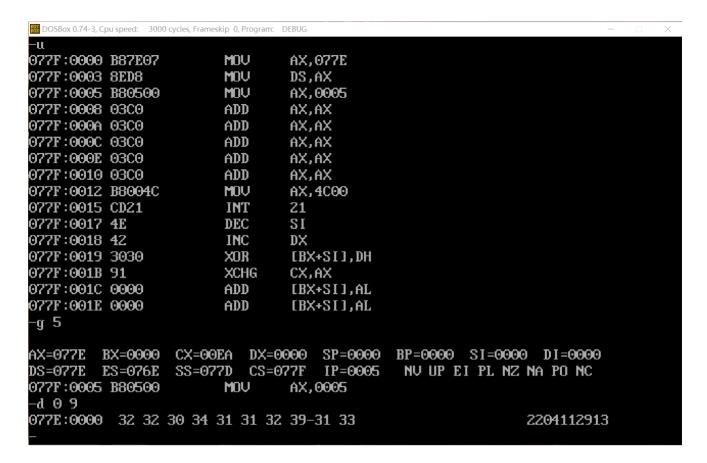
50960 + 465424 Bytes symbol space free

0 Warning Errors
0 Severe Errors
```

### 2.2 场景1的反汇编的截图

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
   Generate reptasm.exe successfully.
______
   debug reptasm.exe ...
077F:0000 B87E07
                        MOV
                                AX,077E
077F:0003 8ED8
                        MOV
                                DS,AX
077F:0005 B80500
                        MOU
                                AX,0005
077F:0008 03C0
                        ADD
                                AX,AX
077F:000A 03C0
                        ADD
                                AX, AX
077F:000C 03C0
                        ADD
                                AX, AX
077F:000E 03C0
                        ADD
                                AX, AX
077F:0010 03C0
                                AX, AX
                        ADD
077F:0012 B8004C
                        MNU
                                AX,4000
077F:0015 CD21
                        INT
                                21
077F:0017 4E
                        DEC
                                SI
077F:0018 42
                        INC
                                DX
077F:0019 3030
                                [BX+SI], DH
                        XOR
077F:001B 91
                        XCHG
                                CX,AX
077F:001C 0000
                        ADD
                                [BX+SI],AL
077F:001E 0000
                        ADD
                                [BX+SI1,AL
```

2.3 场景1的显示X的内存值的截图(多显示、少显示均扣分)



#### 2.4 场景2的.lst文件的截图

# ASM3\REPTASM.LST

ASMIS (ILLI IASMILEST				
Microsoft (R) Macro Assembler conditional Assembly and repiti		4/21/24 23:40:26 Page 1-1		
	name REPTASM title conditional Assemble bly	ly and repitition Assem		
0000 0000 32 32 30 34 31 31 32 39 31 33	data segment ID db '2204112913'			
000A 000A 32 32 30 34 31 31 32 39 31 33	strBegin label byte X db '2204112913'			
0014	strEnd label byte			
0014	data ends			
0000 0000 B8 R 0003 8E D8 0005 B8 000A	code segment assume cs:code, ds:da main proc far mov ax, seg data mov ds, ax mov ax, strEnd-st			
2000	ELSE			
	REPT 6 add ax, ax ENDM			
0008 03 C0 1 000A 03 C0 1	add ax, ax add ax, ax			
000C 03 C0 1	add ax, ax			
000E 03 C0 1 0010 03 C0 1	add ax, ax add ax, ax			
0012 03 C0 1	add ax, ax			
	ENDIF			
0014 B8 4C00	mov ax, 4c00h			
0017 CD 21 0019	int 21h main endp			
0019	code ends			
Microsoft (R) Macro Assembler conditional Assembly and repiti		4/21/24 23:40:26 Symbols-1		
Segments and Groups:				
N a m e	Length Align (	Combine Class		
CODE		NONE NONE		
Symbols:				
N a m e	Type Value	Attr		
ID	. L BYTE 0000 I	DATA		
MAIN	. F PROC 0000 C	CODE Length = 0019		
STRBEGIN		DATA DATA		

```
X . . . . . . . . . . . . L BYTE 000A DATA

@FILENAME . . . . . . . . . . . TEXT reptASM

35 Source Lines
41 Total Lines
9 Symbols

50960 + 465424 Bytes symbol space free

0 Warning Errors
0 Severe Errors
```

#### 2.5 场景2的反汇编的截图

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
    Generate reptasm.exe successfully.
______
   debug reptasm.exe ...
-u
0780:0000 B87E07
                        MOV
                                AX,077E
0780:0003 8ED8
                        MOV
                                DS,AX
0780:0005 B80A00
                        MOU
                                AX,000A
0780:0008 0300
                        ADD
                                AX, AX
0780:000A 03C0
                        ADD
                                AX, AX
0780:000C 03C0
                        ADD
                                AX,AX
                                AX,AX
0780:000E 03C0
                        ADD
0780:0010 0300
                        ADD
                                AX, AX
0780:0012 0300
                        ADD
                                AX, AX
0780:0014 B8004C
                        MOU
                                AX,4000
0780:0017 CD21
                        INT
                                21
0780:0019 4E
                        DEC
                                SI
0780:001A 42
                        INC
                                DX
                                [BX+SI],DH
0780:001B 3030
                        XOR
0780:001D 91
                        XCHG
                                CX,AX
0780:001E 0000
                        ADD
                                [BX+SI],AL
```

2.6 场景2的显示X的内存值的截图(多显示、少显示均扣分)

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program:
-u
0780:0000 B87E07
                         MNU
                                  AX,077E
0780:0003 8ED8
                         MOU
                                  DS,AX
0780:0005 B80A00
                         MOU
                                  AX,000A
0780:0008 0300
                         ADD
                                  AX, AX
0780:000A 03C0
                         ADD
                                  AX, AX
0780:000C 03C0
                         ADD
                                  AX, AX
0780:000E 03C0
                         ADD
                                  AX, AX
0780:0010 0300
                         ADD
                                  AX.AX
0780:0012 0300
                         ADD
                                  AX, AX
0780:0014 B8004C
                         MOV
                                  AX.4000
0780:0017 CD21
                         INT
                                  21
                                  SI
0780:0019 4E
                         DEC
0780:001A 42
                                  DX
                         INC
0780:001B 3030
                         XNR
                                  [BX+SI].DH
0780:001D 91
                         XCHG
                                  CX,AX
0780:001E 0000
                         ADD
                                  [BX+SI],AL
-g 5
AX=077E BX=0000
                  CX=00FC DX=0000 SP=0000 BP=0000 SI=0000 DI=0000
DS=077E ES=076E
                   SS=077D CS=0780 IP=0005
                                                 NV UP EI PL NZ NA PO NC
0780:0005 B80A00
                         MOU
                                  AX.000A
-409
077E:0000 32 32 30 34 31 31 32 39-31 33
                                                                  2204112913
```

#### 2.7 源代码

```
name REPTASM
2
    title conditional Assembly and repitition Assembly
3
4
   data segment
5
       ID db '2204112913'
7
        strBegin label byte
8
        X db '2204112913'
9
        strEnd label byte
10
   data ends
12
13
    code segment
14
        assume cs:code, ds:data
15
        main proc far
16
            mov ax, seg data
17
            mov ds, ax
18
            mov ax, strEnd-strBegin
19
20
        IF strEnd-strBegin LE 5; if length of string is less than or equal to
21
            REPT strEnd-strBegin
                add ax, ax
23
            ENDM
24
        ELSE
```

```
25 REPT 6
26 add ax, ax
27 ENDM
28 ENDIF
29
30 mov ax, 4c00h
31 int 21h
32 main endp
33
34 code ends
35 end main
```