## 汇编语言第2次上机

班级	学号	姓名
计算机2205	2204112913	李雨轩

## 1. 循环程序设计

#### (1). 反汇编的截图

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
LINK : warning L4021: no stack segment
    Generate maxnum.exe successfully.
    debug maxnum.exe ...
-u
07BE:0000 1E
                         PUSH
                                  DS
                                  AX,AX
07BE:0001 33C0
                         XOR
07BE:0003 50
                         PUSH
                                  ΑX
07BE:0004 B87E07
                         MOV
                                  AX,077E
07BE:0007 8ED8
                         MOV
                                  DS,AX
07BE:0009 BAF901
                                  DX.01F9
                         MOV
07BE:000C BE0000
                         MOV
                                  SI,0000
07BE:000F 8BFE
                         MOV
                                  DI,SI
07BE:0011 E80800
                         CALL
                                  0010
07BE:0014 A3F203
                         MOV
                                  [03F2],AX
07BE:0017 893EF403
                                  [03F4],DI
                         MOV
07BE:001B CB
                         RETF
07BE:001C 8BCA
                         MOV
                                  CX,DX
07BE:001E 8B04
                         MOV
                                  AX,[SI]
```

DOSBox 0.74-3, Cpu speed:	3000 cycles, Frame	skip 0, Program: DEBUG – 🗆 🗙
07BE:0011 E80800	CALL	001C
07BE:0014 A3F203	MOV	[03F2],AX
07BE:0017 893EF403	MOV	[03F4],DI
07BE:001B CB	RETF	
07BE:001C 8BCA	MOV	CX,DX
07BE:001E 8B04	MOV	AX,[SI]
–u		
07BE:0020 833C00	CMP	WORD PTR [SI],+00
07BE:0023 7D02	JGE	0027
07BE:0025 F71C	NEG	WORD PTR [SI]
07BE:0027 3904	CMP	[SI],AX
07BE:0029 7E04	JLE	002F
07BE:002B 8B04	MOV	AX,[SI]
07BE:002D 8BFE	MOV	DI,SI
07BE:002F 8D7402	LEA	SI,[SI+02]
07BE:0032 EZEC	LOOP	0020
07BE:0034 C3	RET	
07BE:0035 4E	DEC	SI
07BE:0036 42	INC	DX
07BE:0037 3030	XOR	[BX+SI],DH
07BE:0039 07	POP	ES
07BE:003A 0100	ADD	[BX+SI],AX
07BE:003C 0000	ADD	[BX+SI],AL
07BE:003E 0000	ADD	[BX+SI],AL
- <u>-</u>		

# (2). 在进行计算前,显示数组M开始的n+2个字的内存值的截图(只能显示这n+2个字的内存值,多显示、少显示均扣分)

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program:
                                          DEBUG
AX=077E
         BX=0000 CX=01AE DX=0019 SP=FFFC
                                              BP=0000 SI=0000 DI=0000
DS=077E ES=076E
                  SS=077D CS=0782 IP=0011
                                               NU UP EI PL ZR NA PE NC
0782:0011 E80800
                        CALL
                                 0010
-u
0782:0011 E80800
                        CALL
                                 0010
0782:0014 A33200
                        MOV
                                 [0032],AX
0782:0017 893E3400
                        MOV
                                 [0034],DI
0782:001B CB
                        RETF
0782:001C 8BCA
                        MOV
                                 CX,DX
0782:001E 8B04
                        MOV
                                 AX,[SI]
0782:0020 833000
                                 WORD PTR [SI],+00
                        CMP
0782:0023 7D02
                         JGE
                                 0027
0782:0025 F71C
                        NEG
                                 WORD PTR [SI]
0782:0027 3904
                        CMP
                                 [SI],AX
0782:0029 7E04
                         JLE
                                 002F
0782:002B 8B04
                        MOV
                                 AX,[SI]
0782:002D 8BFE
                        MOV
                                 DI,SI
0782:002F 8D7402
                        LEA
                                 SI,[SI+02]
-d ds:0 35
077E:0000 22 00 04 00 11 00 29 00-13 00 FF FF 02 00 03 00
           04 00 FB FF FF FF 02 00-03 00 04 00 FB FF FF FF
077E:0010
          02 00 03 00 04 00 FB FF-FF FF 02 00 03 00 04 00
077E:0020
077E:0030 FB FF 00 00 00 00
```

# (3). 执行完计算后,显示数组M开始的n+2个字的内存值的截图(只能显示这n+2个字的内存值,多显示、少显示均扣分)

```
BOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
0782:0029 7E04
                     JLE
                            002F
0782:002B 8B04
                     MOV
                            AX.[SI]
0782:002D 8BFE
                     MOV
                            DI,SI
0782:002F 8D7402
                     LEA
                            SI,[SI+02]
-d ds:0 35
077E:0000 22 00 04 00 11 00 29 00-13 00 FF FF 02 00 03 00
                                                       ".....).........
077E:0020 02 00 03 00 04 00 FB FF-FF FF 02 00 03 00 04 00
077E:0030 FB FF 00 00 00 00
-p
AX=0029 BX=0000 CX=0000 DX=0019 SP=FFFC BP=0000 SI=0032 DI=0006
DS=077E ES=076E SS=077D CS=0782 IP=0014
                                         NU UP EI NG NZ AC PO CY
0782:0014 A33200
                     MOV
                            [0032].AX
                                                            DS:0032=0000
-q 1b
AX-0029 BX-0000 CX-0000 DX-0019 SP-FFFC BP-0000 SI-0032 DI-0006
DS=077E ES=076E SS=077D CS=0782 IP=001B NV UP EI NG NZ AC PO CY
0782:001B CB
                     RETF
-d ds:0 35
077E:0000 22 00 04 00 11 00 29 00-13 00 01 00 02 00 03 00
. . . . . . . . . . . . . . . .
077E:0020 02 00 03 00 04 00 05 00-01 00 02 00 03 00 04 00
077E:0030 05 00 29 00 06 00
                                                       . . ) . . .
```

#### (4). 源代码

```
name MaxNumber
title Find Max Number
data segment
; length dw 16
   array label word
    dw 22h, 04h, 11h, 29h, 13h
   dw \ 4 \ dup(-1,2,3,4,-5)
   arrend label word
    max dw ?
    ofs dw ? ;store the first max number
data ends
code segment
   assume cs:code, ds:data
   main proc far
       push ds
        xor ax, ax
        push ax
        mov ax, data
```

```
mov ds, ax
        mov dx, (arrend - array)/2; get array's length
        mov si, offset array
        mov di, si
        call findMax
        mov max, ax
        mov ofs, di
       ret
   main endp
   findMax proc near
    ; dx - length, si - array, di - max index
    ; return the max num in ax
       mov cx, dx
        mov ax, [si]; ax stores the max number
loopH: cmp word ptr [si], 0;
       jnl short whennl
        neg word ptr [si]
whennl: cmp [si], ax
       jng short whenng
        mov ax, [si]
       mov di, si
whenng: lea si, 2[si]
       loop loopH
       ret
    findMax endp
code ends
   end main
```

## 2. 分支程序设计

### (1). 反汇编的截图

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
    Generate count.exe successfully.
   debug count.exe ...
-u
0782:0000 1E
                         PUSH
                                  DS
                                  AX,AX
0782:0001 3300
                         XOR
0782:0003 50
                         PUSH
                                  ΑX
0782:0004 B87E07
                         MOV
                                  AX,077E
0782:0007 8ED8
                         MOV
                                  DS,AX
0782:0009 3300
                         XOR
                                  AX,AX
0782:000B BF0000
                         MOV
                                  DI,0000
0782:000E BE2400
                         MOV
                                  SI,0024
                                  AL,[DI]
0782:0011 8A05
                         MOV
                         CMP
0782:0013 3039
                                  AL,39
0782:0015 7F11
                         JG
                                  0028
0782:0017 3030
                                  AL,30
                         CMP
0782:0019 7COD
                         JL
                                  0028
0782:001B 2C30
                         SUB
                                  AL,30
0782:001D 98
                         CBW
0782:001E 8BE8
                         MOV
                                  BP,AX
```

DOSBox 0.74-3, Cpu speed:	3000 cycles, Frames	skip 0, Program:	DEBUG	_	×
0782:000E BE2400	MOV	SI,0024			
0782:0011 8A05	MOV	AL,[DI]			
0782:0013 3039	CMP	AL,39			
0782:0015 7F11	JG	0028			
0782:0017 3030	CMP	AL,30			
0782:0019 7COD	JL	0028			
0782:001B 2C30	SUB	AL,30			
0782:001D 98	CBW				
0782:001E 8BE8	MOV	BP,AX			
–u					
0782:0020 03E8	ADD	BP,AX			
0782:0022 3E	DS:				
0782:0023 830201	ADD	WORD PTR	[BP+SI],+01		
0782:0026 EBEB	JMP	0013			
0782:0028 3024	CMP	AL,24			
0782:002A 7407	JZ	0033			
0782:002C 83C701	ADD	DI,+01			
0782:002F 8A05	MOV	AL,[DI]			
0782:0031 EBE0	JMP	0013			
0782:0033 BA0A00	MOV	DX,000A			
0782:0036 E84500	CALL	007E			
0782:0039 81EF2400	SUB	DI,0024			
0782:003D D1EF	SHR	DI,1			
0782:003F 50	PUSH	AX			
_					

DOSBox 0.74-3, Cpu speed:	3000 cycles, Frames	kip 0, Program:	DEBUG	_	X
0782:0036 E84500	CALL	007E			
0782:0039 81EF2400	SUB	DI,0024			
0782:003D D1EF	SHR	DI,1			
0782:003F 50	PUSH	AX			
–u					
0782:0040 52	PUSH	DX			
0782:0041 8BC7	MOV	AX,DI			
0782:0043 053000	ADD	AX,0030			
0782:0046 8ADO	MOV	DL,AL			
0782:0048 B402	MOV	AH,02			
0782:004A CD21	INT	21			
0782:004C 5A	POP	DX			
0782:004D 58	POP	AX			
0782:004E 50	PUSH	AX			
0782:004F 52	PUSH	DX			
0782:0050 B8FCFF	MOV	AX,FFFC			
0782:0053 053000	ADD	AX,0030			
0782:0056 8ADO	MOV	DL,AL			
0782:0058 B402	MOV	AH,02			
0782:005A CD21	INT	21			
0782:005C 5A	POP	DX			
0782:005D 58	POP	AX			
0782:005E 50	PUSH	AX			
0782:005F 52	PUSH	DX			
_					
			22110		 
DOSBox 0.74-3, Cpu speed:		-	DEBUG	_	×
0782:005A CD21	INT	21	DEBUG	_	×
0782:005A CD21 0782:005C 5A	INT POP	21 DX	DEBUG	-	×
0782:005A CD21 0782:005C 5A 0782:005D 58	INT POP POP	21 DX AX	DEBUG	_	×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50	INT POP POP PUSH	21 DX AX AX	DEBUG	_	×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52	INT POP POP	21 DX AX	DEBUG	_	×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u	INT POP POP PUSH PUSH	21 DX AX AX DX	DEBUG	_	×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF	INT POP POP PUSH PUSH	21 DX AX AX DX	DEBUG	_	×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000	INT POP POP PUSH PUSH MOV ADD	21 DX AX AX DX AX,FFF0 AX,0030	DEBUG	_	X
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0	INT POP POP PUSH PUSH MOV ADD MOV	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL	DEBUG	_	×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0068 B402	INT POP POP PUSH PUSH MOV ADD MOV MOV	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL AH,02	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0068 B402 0782:006A CD21	INT POP POP PUSH PUSH MOV ADD MOV MOV INT	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL AH,02 21	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0068 B402 0782:006A CD21 0782:006C 5A	INT POP POP PUSH PUSH MOV ADD MOV INT POP	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL AH,02 21 DX	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 BAD0 0782:0068 B402 0782:006A CD21 0782:006C 5A 0782:006D 58	INT POP POP PUSH PUSH  MOV ADD MOV INT POP POP	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL AH,02 21 DX AX	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0068 B402 0782:006A CD21 0782:006C 5A 0782:006D 58	INT POP POP PUSH PUSH MOV ADD MOV MOV INT POP POP	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0068 B402 0782:006A CD21 0782:006C 5A 0782:006E 50 0782:006F 52	INT POP POP PUSH PUSH MOV ADD MOV INT POP PUSH PUSH	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX AX DX	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 BAD0 0782:0068 B402 0782:006A CD21 0782:006C 5A 0782:006E 50 0782:006F 52 0782:006F 52	INT POP POP PUSH PUSH MOV ADD MOV INT POP PUSH PUSH MOV	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX AX DX AX,AX	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0068 B402 0782:006A CD21 0782:006C 5A 0782:006E 50 0782:006F 52	INT POP POP PUSH PUSH MOV ADD MOV INT POP POP PUSH PUSH MOV ADD	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL AH,02 21 DX AX AX DX AX,AX AX,0030	DEBUG		X
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 B402 0782:006A CD21 0782:006C 5A 0782:006C 5A 0782:006E 50 0782:006F 52 0782:0070 8BC0 0782:0072 053000	INT POP POP PUSH PUSH MOV ADD MOV INT POP PUSH PUSH MOV	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX AX DX AX,AX	DEBUG		X
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 BAD0 0782:006A CD21 0782:006C 5A 0782:006D 58 0782:006F 52 0782:006F 52 0782:0070 8BC0 0782:0072 053000 0782:0075 8AD0	INT POP POP PUSH PUSH MOV ADD MOV INT POP POP PUSH PUSH MOV ADD MOV	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL AH,02 21 DX AX AX AX,AX AX,0030 DL,AL	DEBUG		X
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:006A CD21 0782:006C 5A 0782:006D 58 0782:006F 52 0782:006F 52 0782:0072 053000 0782:0075 8AD0 0782:0077 B402	INT POP POP PUSH PUSH MOV ADD MOV INT POP POP PUSH PUSH MOV ADD MOV MOV ADD	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX AX AX,0030 DL,AL AH,02	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0066 B402 0782:006A CD21 0782:006C 5A 0782:006D 58 0782:006E 50 0782:006F 52 0782:0070 8BC0 0782:0072 053000 0782:0075 8AD0 0782:0077 B402	INT POP POP PUSH PUSH MOV ADD MOV INT POP POP PUSH PUSH MOV ADD MOV INT	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX AX DX AX,AX AX,0030 DL,AL AH,02 21	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 8AD0 0782:0068 B402 0782:006A CD21 0782:006C 5A 0782:006E 50 0782:006F 52 0782:006F 52 0782:0070 8BC0 0782:0072 053000 0782:0075 8AD0 0782:0079 CD21 0782:0079 CD21	INT POP POP PUSH PUSH MOV ADD MOV INT POP PUSH PUSH MOV ADD MOV INT POP PUSH PUSH MOV INT POP	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX AX DX AX,AX AX,0030 DL,AL AH,02 21 DX	DEBUG		X
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 BAD0 0782:0068 B402 0782:006A CD21 0782:006C 5A 0782:006E 50 0782:006F 52 0782:006F 52 0782:0070 BBC0 0782:0072 053000 0782:0075 BAD0 0782:0079 CD21 0782:0078 5A 0782:0078 5A	INT POP POP PUSH PUSH MOV ADD MOV INT POP PUSH PUSH MOV ADD MOV INT POP POP PUSH POP POP	21 DX AX AX DX AX,FFFO AX,0030 DL,AL AH,02 21 DX AX AX DX AX,AX AX,0030 DL,AL AH,02 21 DX	DEBUG		×
0782:005A CD21 0782:005C 5A 0782:005D 58 0782:005E 50 0782:005F 52 -u 0782:0060 B8F0FF 0782:0063 053000 0782:0066 BAD0 0782:006A CD21 0782:006C 5A 0782:006E 50 0782:006F 52 0782:006F 52 0782:0070 BBC0 0782:0072 053000 0782:0075 BAD0 0782:0079 CD21 0782:0078 5A 0782:007C 58 0782:007C 58	INT POP POP PUSH PUSH MOV ADD MOV INT POP POP PUSH MOV ADD MOV INT POP POP PUSH PUSH MOV ADD MOV INT POP POP RETF	21 DX AX AX DX AX,FFF0 AX,0030 DL,AL AH,02 21 DX AX AX,AX AX,0030 DL,AL AH,02 21 DX AX,AX	DEBUG		×

(2). 在进行计算前,显示在数据段中定义的含学号的字符串的内存值的截图(只能显示该 完整的字符串,多显示、少显示均扣分)

DOSBox 0.74-3, Cpu speed:	3000 cycles, Framesk	skip 0, Program: DEBUG — 🗆 🗙
0782:0013 3039	CMP	AL,39
0782:0015 7F11	JG	0028
0782:0017 3030	CMP	AL,30
0782:0019 7COD	JL	0028
0782:001B 2C30	SUB	AL,30
0782:001D 98	CB₩	
0782:001E 8BE8	MOV	BP,AX
0782:0020 03E8	ADD	BP,AX
0782:0022 3E	DS:	
0782:0023 830201	ADD	WORD PTR [BP+SI],+01
0782:0026 EBEB	JMP	0013
0782:0028 3024	CMP	AL,24
0782:002A 7407	JZ	0033
0782:002C 83C701	ADD	DI,+01
0782:002F 8A05	MOV	AL,[DI]
-g11		
3		
AX=0000 BX=0000 0	CX=027F DX=0	0000 SP=FFFC BP=0000 SI=0024 DI=0000
DS=077E ES=076E S	SS=077D CS=0	0782 IP=0011 NU UP EI PL ZR NA PE NC
0782:0011 8A05	MOV	AL,[DI] DS:0000=32
-d ds:0 23		
	9 34 31 31 32	2 39-31 33 2D 6C 69 2D 79 75 2204112913-li-yu
077E:0010 78 75 61	1 6E 2D 61 73	3 73-65 6D 62 6C 79 39 39 39 xuan-assembly999
077E:0020 39 39 39	9 24	999\$
- <u>-</u>		

# (3). 在进行计算前,显示在数据段中定义的COUNT数组的内存值的截图(只能显示完整的COUNT数组内容,多显示、少显示均扣分)

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DEBUG
0782:0019 7COD
                         JL
                                 0028
0782:001B 2C30
                         SUB
                                 AL,30
0782:001D 98
                         CBW
0782:001E 8BE8
                         MOV
                                 BP,AX
0782:0020 03E8
                         ADD
                                 BP,AX
0782:0022 3E
                         DS:
0782:0023 830201
                         ADD
                                 WORD PTR [BP+SI],+01
0782:0026 EBEB
                         JMP
                                 0013
                         CMP
0782:0028 3024
                                 AL,24
0782:002A 7407
                         JZ
                                 0033
0782:002C 83C701
                                 DI,+01
                         ADD
0782:002F 8A05
                                 AL,[DI]
                         MOV
-g11
AX=0000 BX=0000 CX=027F DX=0000 SP=FFFC BP=0000 SI=0024 DI=0000
DS=077E ES=076E SS=077D CS=0782 IP=0011
                                                NU UP EI PL ZR NA PE NC
0782:0011 8A05
                         MOV
                                 AL,[DI]
                                                                       DS:0000=32
-d ds:0 23
077E:0000   32  32  30  34  31  31  32  39-31  33  2D  6C  69  2D  79  75
                                                                2204112913-1 i-yu
077E:0010 78 75 61 6E 2D 61 73 73-65 6D 62 6C 79 39 39 39
                                                                xuan-assembly999
077E:0020 39 39 39 24
                                                                999$
-d ds:24 37
077E:0020
                        00 00 00 00-00 00 00 00 00 00 00 00
077E:0030 00 00 00 00 00 00 00 00
```

(4). 执行完计算后,显示在数据段中定义的含学号的字符串的内存值的截图(只能显示该完整的字符串,多显示、少显示均扣分)

```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program:
                                          DEBUG
                         CMP
0782:0013 3039
                                 AL,39
0782:0015 7F11
                         JG
                                 0028
0782:0017 3030
                                 AL,30
                         CMP
0782:0019 7COD
                         JL
                                 0028
                                 AL,30
0782:001B 2C30
                         SUB
0782:001D 98
                         CBW
0782:001E 8BE8
                         MOV
                                 BP,AX
0782:0020 03E8
                         ADD
                                 BP.AX
0782:0022 3E
                         DS:
0782:0023 830201
                         ADD
                                 WORD PTR [BP+SI],+01
0782:0026 EBEB
                         JMP
                                 0013
0782:0028 3024
                         CMP
                                 AL,24
0782:002A 7407
                         JZ
                                 0033
0782:0020 830701
                                 DI,+01
                         ADD
0782:002F 8A05
                         MOV
                                 AL,[DI]
-q39
AX-0007 BX-0000 CX-0000 DX-000A SP-FFFC BP-0012 SI-0038 DI-0036
DS=077E ES=076E SS=077D CS=0782
                                     IP=0039
                                               NV UP EI PL NZ NA PO NC
0782:0039 81EF2400
                         SUB
                                 DI.0024
-d ds:0 23
077E:0000 32 32 30 34 31 31 32 39-31 33 2D 6C 69 2D 79 75
                                                                2204112913-1 i-uu
077E:0010 78 75 61 6E 2D 61 73 73-65 6D 62 6C 79 39 39 39
                                                               xuan-assembly999
           39 39 39 24
                                                                999$
077E:0020
```

(5). 执行完计算后,显示在数据段中定义的COUNT数组的内存值的截图(只能显示完整的COUNT数组内容,多显示、少显示均扣分)

DOSBox 0.74-3, Cpu speed:	3000 cycles, Framesk	kip 0, Program:	DEBUG	- 🗆 ×
0782:0019 7COD	JL	0028		
0782:001B 2C30	SUB	AL,30		
0782:001D 98	CBW			
0782:001E 8BE8	MOV	BP,AX		
0782:0020 03E8	ADD	BP,AX		
0782:0022 3E	DS:			
0782:0023 830201	ADD		[BP+SI],+01	
0782:0026 EBEB	JMP	0013		
0782:0028 3C24	CMP	AL,24		
0782:002A 7407	JZ	0033		
0782:002C 83C701	ADD	DI,+01		
0782:002F 8A05	MOV	AL,[DI]		
-g39				
	X=0000 DX=0			I=0036
	SS=077D_CS=0		39 NU UPEIPLNZNAI	PO NC
0782:0039 81EF2400	SUB	DI,0024		
-d ds:0 23				
				4112913-1 i-yu
		73-65 6D		n-assembly999
077E:0020 39 39 39	24		9999	ý
-d ds:24 37				
077E:0020			01 00 01 00 00 00	
077E:0030 00 00 00	0 00 00 00 07	00		
- <u>-</u>				

### (6). 程序在DOSBox下直接运行的截图

DOSBox 0.74-3, Cpu speed:	3000 cycles, Framesl	ip 0, Program: DOSBOX	- 🗆 X
0782:001E 8BE8	MOV	BP,AX	
0782:0020 03E8	ADD	BP,AX	
0782:0022 3E	DS:		
0782:0023 830201	ADD	WORD PTR [BP+SI],+01	
0782:0026 EBEB	JMP	0013	
0782:0028 3024	CMP	AL,24	
0782:002A 7407	JZ	0033	
0782:002C 83C701	ADD	DI,+01	
0782:002F 8A05	MOV	AL,[DI]	
-g39			
AX=0007 BX=0000 C	X=0000 DX=0	90A SP=FFFC BP=0012 SI=0038	B DI=0036
	SS=077D CS=0		
0782:0039 81EF2400	SUB	DI,0024	10
-d ds:0 23	332	21,0021	
	34 31 31 32	39-31 33 2D 6C 69 2D 79 75	2204112913-1 i-yu
		73-65 6D 62 6C 79 39 39 39	xuan-assembly999
077E:0020 39 39 39			999\$
-d ds:24 37			
077E:0020	01 00 03	00-03 00 01 00 01 00 00 00	
	0 00 00 00 07		
-q			
C:\LEARN\ASM2>.\cou	ınt.exe		
9, 7			
C:\LEARN\ASMZ>			

### (7). 源代码

```
printnum macro num
   push ax
   push dx
   mov ax, num
   add ax, 30h
   mov dl, al
   mov ah, 02h
   int 21h
   pop dx
   pop ax
endm
name CountString
title Count String
data segment
    mystring db '2204112913-li-yuxuan-assembly9999999$'
    countarray dw 10 dup(0)
   tests db 9
data ends
code segment
    assume cs:code, ds:data
    main proc far
        ; init
        push ds
        xor ax, ax
        push ax
        mov ax, data
        mov ds, ax
        xor ax, ax
        ; count number
        mov di, offset mystring
        mov si, offset countarray
        mov al, ds:[di]
begin: cmp al, 39h
        jg short incr
        cmp al, 30h
        jl short incr
        sub al, 30h
        cbw
        mov bp, ax
        add bp, ax
        add word ptr ds:[bp+si], 1
```

```
jmp short begin
incr: cmp al, '$'
       je short endstr
        add di, type mystring
        mov al, ds:[di]
        jmp short begin
endstr: mov dx, length countarray
       call findMax
       sub di, offset countarray
       shr di, 1
        printnum di
       printnum ','-30h
        printnum ' '-30h
       printnum ax
        ret
   main endp
   findMax proc near
   ; dx - length, si - array, di - max_index
    ; return the max num in ax
       mov cx, dx ; length
       mov ax, [si]; ax stores the max number
loopH: cmp word ptr [si], 0;
       jnl short whennl
       neg word ptr [si]
whennl: cmp [si], ax
       jl short whenl
       mov ax, [si]
       mov di, si
whenl: lea si, 2[si]
       loop loopH
       ret
   findMax endp
code ends
   end main
```