1 chapter10

1.1 1. 显示字符串

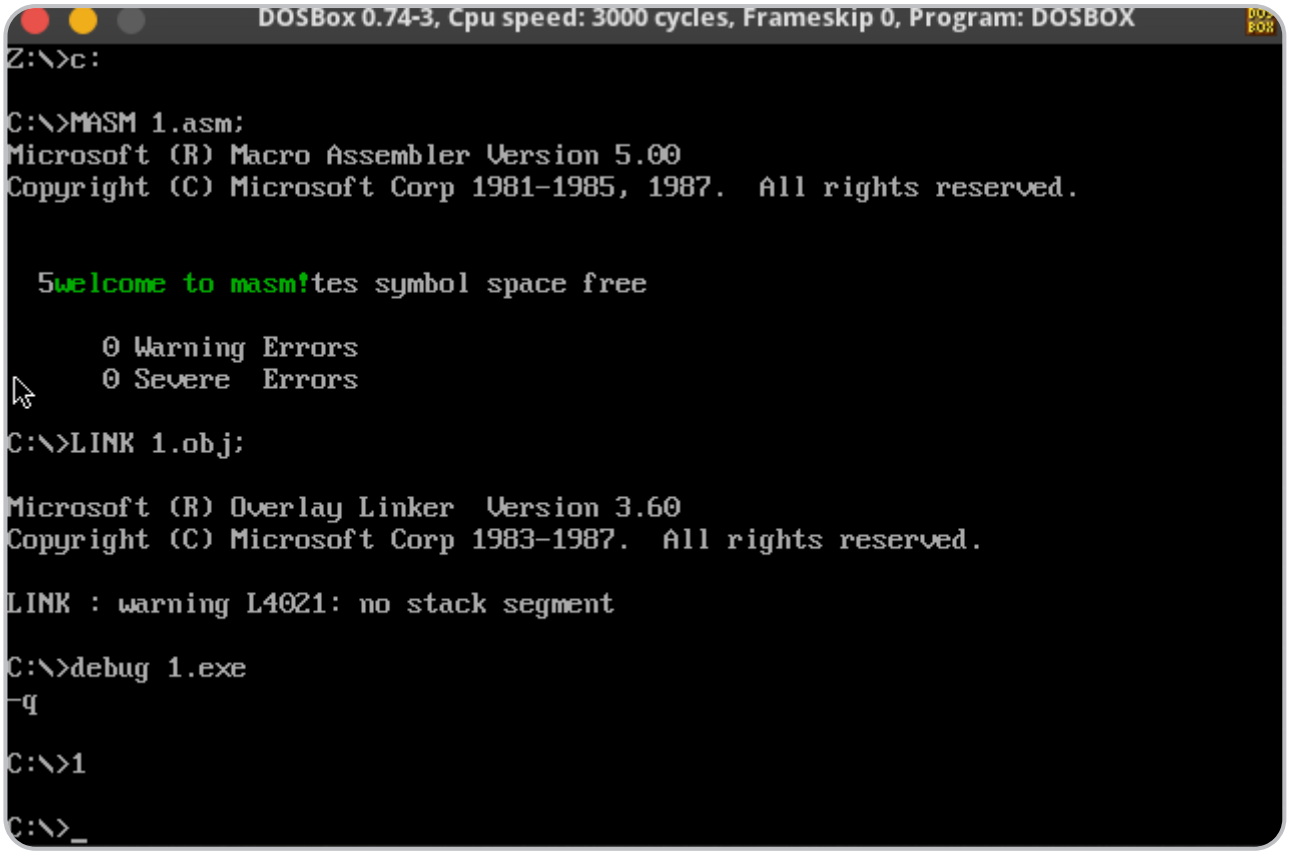
1.1.1 代码

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

data segment
    db 'welcome to masm!',0
data ends

stack segment
    db 128 dup(0)
stack ends

code segment


    start:

    mov ax,stack
    mov ss,ax
    mov sp,128

    mov ax,data
    mov ds,ax
    mov si,0

    mov dh,8;row
    mov dl,3;cul
    mov cl,2;color
    call show_str

    mov ax,4c00h
    int 21h

    show_str:
        push ax
        push dx
        push cx
```

```
        push es
        push si
        push di

        mov ax,0b800h
        mov es,ax
        mov di,0

        call get_row
        add di,ax
        call get_cul
        add di,ax

        call show

        pop di
        pop si
        pop es
        pop cx
        pop dx
        pop ax
        ret

    get_row:
        mov al,160
        mul dh
        ret
    get_cul:
        mov al,2
        mul dl
        ret

    show:
        push ax
        push bx
        push cx
        push di
        push si

        sub ax,ax
        sub bx,bx


        mov bl,cl
        show_1:
            mov cl,ds:[si]
            mov al,ds:[si]
            mov ch,0
            jcxz ok
            mov ah,bl
            mov es:[di],ax
            add di,2
```

```
88              inc si
89              jmp show_1
90
91
92          ok:
93          pop si
94          pop di
95          pop cx
96          pop bx
97          pop ax
98          ret
99
100         ret
101 code ends
102 end start
```

1.1.2 截屏



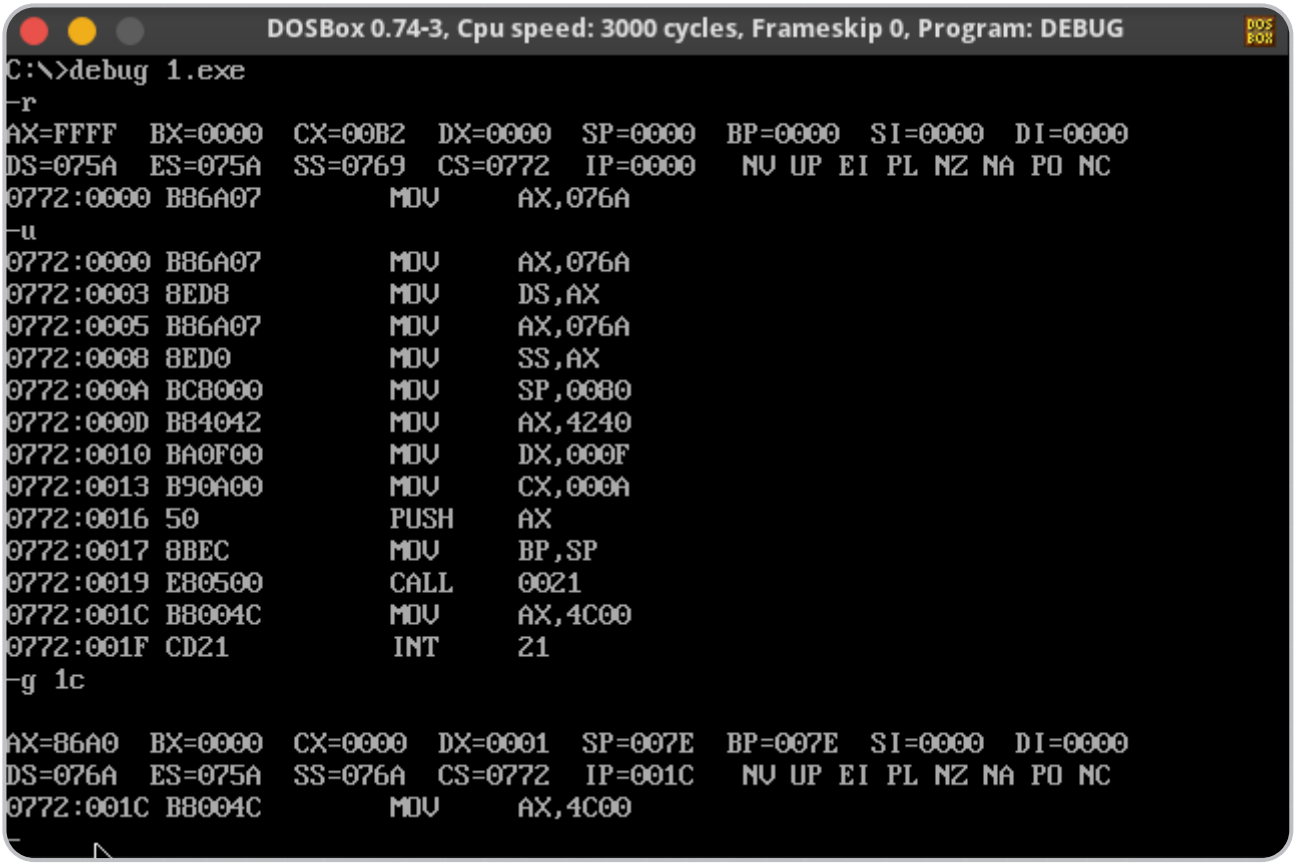
1.2 2. 解决除法溢出问题

1.2.1 代码

```
1 assume cs:code,ds:data,ss:stack
```

```
2
3   data segment
4   data ends
5
6   stack segment
7       db 128 dup(0)
8   stack ends
9
10  code segment
11      start:
12          mov ax,data
13          mov ds,ax
14          mov ax,stack
15          mov ss,ax
16          mov sp,128
17
18          mov ax,4240h;L
19          mov dx,000fh;H
20          mov cx,0ah;N
21
22          push ax;暂时存储ax
23          mov bp,sp;记录ax中数据的位置
24          call divdw
25
26          mov ax,4c00h
27          int 21h
28
29          divdw:
30
31              mov ax,dx
32              mov dx,0
33              div cx
34              push ax
35
36              mov ax,ss:[bp+0];
37              div cx
38              mov cx,dx
39
40              pop dx
41
42              ret
43  code ends
44  end start
```

1.2.2 截图



1.3 3. 数值显示

1.3.1 代码

```
1  assume cs:code,ds:data,ss:stack
2
3  data segment
4      dw 123,12666,1,8,3,38,0
5  data ends
6
7  string segment
8      db 10  dup (0),0
9  string ends
10
11 stack segment
12     db 128 dup(0)
13 stack ends
14
15 code segment
16
17
18     start:
19 ;初始化
20     mov ax,stack
21     mov ss,ax
```

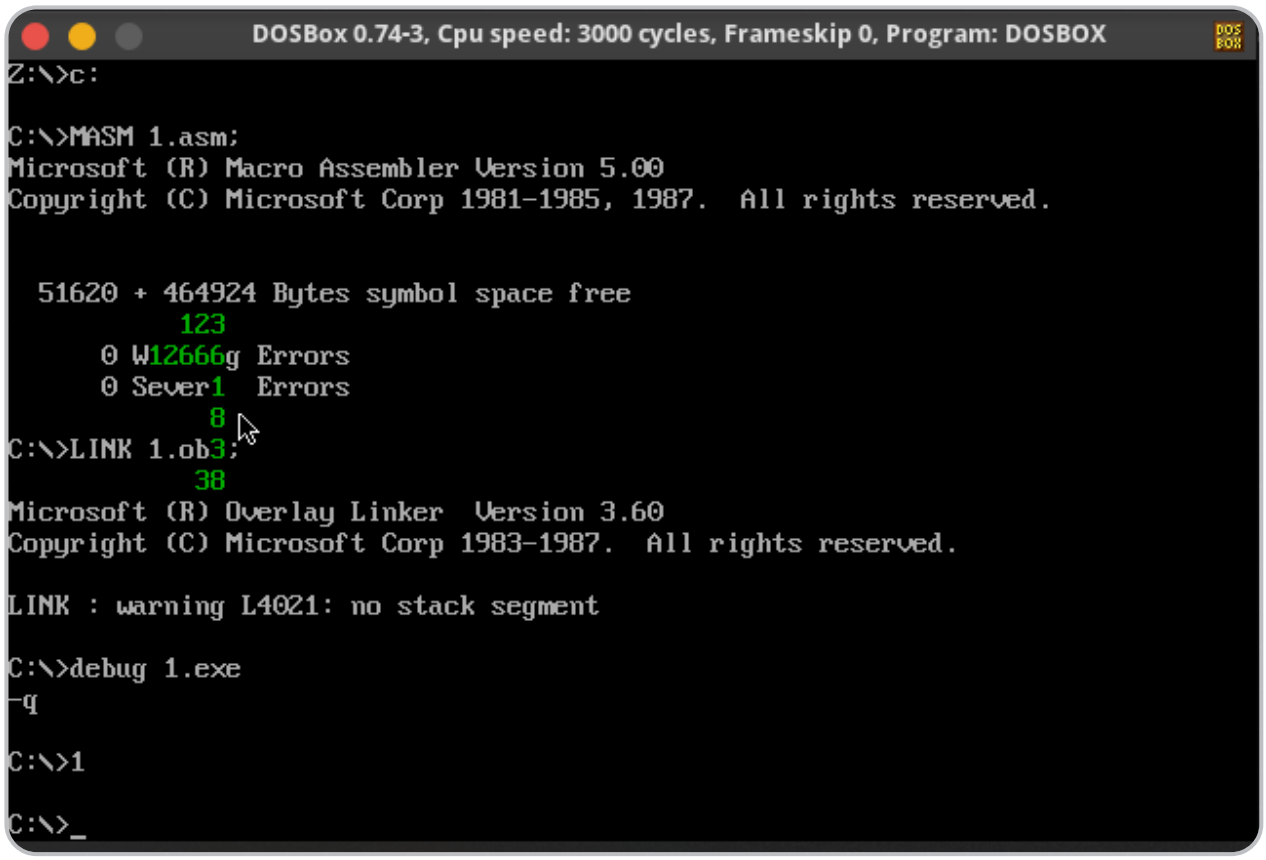
```
    mov sp,128

    call init_reg

    mov dh,8;row
    mov dl,3;cul
    mov cl,2;color
;显示
    a:
    mov ax,ds:[si]
    mov cx,ax
    jcxz Pro_end
    call dtoc

    inc dh
    mov dl,3;cul
    mov cl,2;color
    call show_str

    call zero

    add si,2
    jmp a

    Pro_end:
    mov ax,4c00h
    int 21h

;把string全部清零
    zero:
                push ax
                push bx
                push cx
                push dx
                push ds
                push es
                push si
                push di

        mov cx,10
        mov ax,string
        mov es,ax
        mov di,0

        lp:
        mov al,0
        mov es:[di],al
        inc di
        loop lp

    zero_ret:
                pop di
```

```
                pop si
                pop es
                pop ds
                pop dx
                pop cx
                pop bx
                pop ax
        ret
;==========
    dtoc:
                push ax
                push bx
                push cx
                push dx
                push ds
                push es
                push si
                push di

        mov si,0
        mov dx,0
    dtoc_bg:
        mov cx,10
        div cx
        add dl,30h
        mov es:[si],dl
        mov cx,ax
        jcxz dtoc_end
        inc si
        mov dx,0
        jmp dtoc_bg


    dtoc_end:
                pop di
                pop si
                pop es
                pop ds
                pop dx
                pop cx
                pop bx
                pop ax
        ret
;========
    init_reg:
        mov ax,data
        mov ds,ax
        mov ax,string
        mov es,ax

        mov si,0

```

```
126         ret
127
128 ;=======
129     show_str:
130                 push ax
131                 push bx
132                 push cx
133                 push dx
134                 push ds
135                 push es
136                 push si
137                 push di
138
139         mov ax,string
140         mov ds,ax
141         mov si,0
142
143         mov ax,0b800h
144         mov es,ax
145         mov di,0
146
147         call get_row
148         add di,ax
149         add dl,10
150         call get_cul
151         add di,ax
152
153         call show
154
155                 pop di
156                 pop si
157                 pop es
158                 pop ds
159                 pop dx
160                 pop cx
161                 pop bx
162                 pop ax
163         ret
164
165     get_row:
166         mov al,160
167         mul dh
168         ret
169     get_cul:
170         mov al,2
171         mul dl
172         ret
173
174     show:
175                 push ax
176                 push bx
177                 push cx
```

```
                push dx
                push ds
                push es
                push si
                push di

        sub ax,ax
        sub bx,bx


        mov bl,cl
        show_1:
            mov cl,ds:[si]
            mov al,ds:[si]
            mov ch,0
            jcxz ok
            mov ah,bl
            mov es:[di],ax
            sub di,2
            inc si
            jmp show_1


        ok:
                pop di
                pop si
                pop es
                pop ds
                pop dx
                pop cx
                pop bx
                pop ax
        ret

code ends
end start
```

1.3.2 截图



2 课程设计1

2.0.3 代码

```
1  assume cs:code,ds:data,ss:stack
2
3  data segment
4          ;0,0
5          db  '1975','1976','1977','1978','1979','1980','1981','1982','1983'
6          db  '1984','1985','1986','1987','1988','1989','1990','1991','1992'
7          db  '1993','1994','1995'
8          ;以上是表示21年的21个字符串 year
9
10         ;84,54h
11         dd  16,22,382,1356,2390,8000,16000,24486,50065,97479,140417,197514
12         dd  345980,590827,803530,1183000,1843000,2759000,3753000,4649000,5937000
13         ;以上是表示21年公司总收入的21个dword数据    sum
```

```
14
15          ;168,a8
16          dw  3,7,9,13,28,38,130,220,476,778,1001,1442,2258,2793,4037,5635,8226
17          dw  11542,14430,15257,17800
18  data ends
19
20
21
22  stack segment
23      db 128 dup(0)
24  stack ends
25
26
27  table segment   ;0123456789ABCDEF
28      db  21 dup ('year summ ne ?? ')
29  table ends
30
31  string segment ;0123456789ABCDEF
32      db  21 dup (0000000000000000)
33  string ends
34
35
36  code segment
37  start:
38              mov ax,stack;stack
39              mov ss,ax
40              mov sp,128
41
42              call clear_screen;清空屏幕
43
44              call init_reg;初始化寄存器组，把数据导入
45
46              call input_table;把数据格式化存储
47
48              call output_table;输出格式化数据
49
50              mov ax,4C00H
51              int 21H
52
53  ;================================;output_table
54  output_table:
55              push ax
56              push bx
57              push cx
58              push dx
59              push es
60              push ds
61              push di
62              push si
63
64      output_table_bg:
65              ;分别调用显示子程序
```

```
66              call print_year;年
67              call print_income;总收入
68              call print_employee;员工数量
69              call print_avg;人均收入
70
71      output_table_end:
72              pop si
73              pop di
74              pop ds
75              pop es
76              pop dx
77              pop cx
78              pop bx
79              pop ax
80              ret
81      ;=========================================
82      print_avg:
83              push ax
84              push bx
85              push cx
86              push dx
87              push es
88              push ds
89              push di
90              push si
91
92              mov ax,table
93              mov ds,ax
94              mov ax,string
95              mov es,ax
96
97              mov si,0
98
99          print_avg_bg:
100
101             mov cx,21
102             mov di,160*3+2*40;在屏幕中的显示位置
103         pa_lp1:
104             push cx
105
106             mov ax,ds:[si+13]
107             mov dx,0
108             mov cx,10
109
110             call transfer;改进后的子程序3
111
112             add si,16
113             add di,160
114             pop cx
115             loop pa_lp1
116
117             pop si
```

```
            pop di
            pop ds
            pop es
            pop dx
            pop cx
            pop bx
            pop ax
            ret

    ;=========================================
    print_employee:
            push ax
            push bx
            push cx
            push dx
            push es
            push ds
            push di
            push si

            mov ax,table
            mov ds,ax
            mov ax,string
            mov es,ax

            mov si,0

        print_employee_bg:
            mov cx,21
            mov di,160*3+2*30

        peb_lp1:
            push cx

            mov ax,ds:[si+10]
            mov dx,0
            mov cx,10

            call transfer

            add si,16
            add di,160
            pop cx
            loop peb_lp1

        print_employee_end:
            pop si
            pop di
            pop ds
            pop es
            pop dx
            pop cx
```

```
170             pop bx
171             pop ax
172             ret
173
174     ;=========================================
175     print_income:
176             push ax
177             push bx
178             push cx
179             push dx
180             push es
181             push ds
182             push di
183             push si
184
185             mov ax,table
186             mov ds,ax
187             mov ax,string
188             mov es,ax
189
190             mov si,0
191
192
193         print_income_bg:
194
195             mov cx,21
196             mov di,160*3+2*20
197         pib_lp1:
198             push cx
199
200             mov ax,ds:[si+5]
201             mov dx,ds:[si+7]
202             mov cx,10
203
204
205             call transfer
206
207             add si,16
208             ;add di,16
209             add di,160
210             pop cx
211             loop pib_lp1
212
213
214         print_income_end:
215             pop si
216             pop di
217             pop ds
218             pop es
219             pop dx
220             pop cx
221             pop bx
```

```
222             pop ax
223             ret
224 
225         ;=============================
226         transfer:
227             push ax
228             push bx
229             push cx
230             push dx
231             push es
232             push ds
233             push di
234             push si
235 
236             mov bx,15
237 
238         transfer_bg:    ;判断是否大于一个字节
239             mov cx,dx
240             jcxz short_div;不大于一个字节
241 
242             mov cx,10
243             push ax
244             mov bp,sp
245             call divdw
246             add sp,2
247 
248             add cl,30h
249             mov es:[bx],cl
250 
251             dec bx
252             jmp transfer_bg
253         sn:
254             call show_number;输出结果
255 
256         transfer_end:
257             pop si
258             pop di
259             pop ds
260             pop es
261             pop dx
262             pop cx
263             pop bx
264             pop ax
265             ret
266         ;===================
267         show_number:
268             push ax
269             push bx
270             push cx
271             push dx
272             push es
273             push ds
```

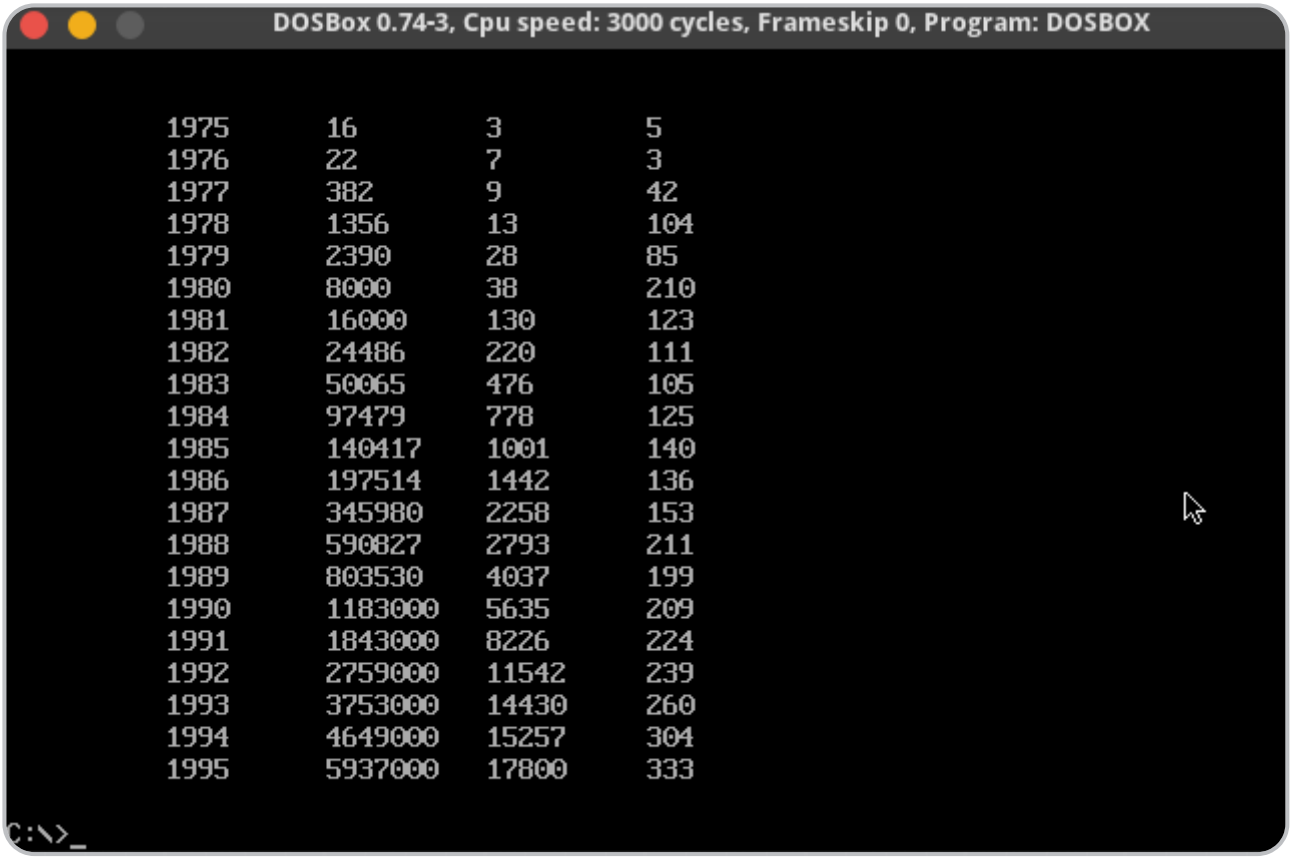
```
            push di
            push si

            mov ax,string
            mov ds,ax
            mov ax,0b800h
            mov es,ax


        sn_lp1:
            mov cx,0
            mov cl,ds:[bx]
            jcxz show_number_end
            mov ch,00000111b
            mov es:[di],cx

            inc bx
            add di,2

            jmp sn_lp1

        show_number_end:
            pop si
            pop di
            pop ds
            pop es
            pop dx
            pop cx
            pop bx
            pop ax
            ret

        ;====================
        divdw:

        mov ax,dx
        mov dx,0

        div cx
        push ax
        mov ax,ss:[bp+0]
        div cx

        mov cx,dx
        pop dx

        ret

        ;====================
        short_div:
        mov cx,10
        div cx
```

```
        add dl,30h
        mov es:[bx],dl
        mov cx,ax
        jcxz sn
        dec bx
        mov dx,0
        jmp short_div

    ;=============================================
    print_year:
            push ax
            push bx
            push cx
            push dx
            push es
            push ds
            push di
            push si

            ;0123456789ABCDEF
            ;year summ ne ??

        print_year_bg:
            mov ax,table
            mov ds,ax
            mov ax,0b800h
            mov es,ax
            mov si,0
            mov di,160*3+2*10

            mov cx,21
        pyb_lp1:
            push cx
            mov cx,4
            mov bx,0
        pyb_lp2:
            mov al,ds:[si+bx]
            ;mov ah,00000111b
            mov ah,00000111b
            push bx
            add bx,bx
            mov es:[di+bx],ax
            pop bx

            inc bx
            loop pyb_lp2

            add si,16
            add di,160
            pop cx
            loop pyb_lp1

```

```
        print_year_end:
            pop si
            pop di
            pop ds
            pop es
            pop dx
            pop cx
            pop bx
            pop ax
            ret

;================================;input_table
input_table:
            push ax
            push bx
            push cx
            push dx
            push es
            push ds
            push di
            push si

            mov si,0
            mov di,0
            mov bx,21*4*2

            mov cx,21


                ;0123456789ABCDEF
                ;year summ ne ??
    input_table_bg:
            ;year
            push ds:[si+0]
            pop es:[di+0]
            push ds:[si+2]
            pop es:[di+2]
            ;income
            mov ax,ds:[si+21*4+0]
            mov dx,ds:[si+21*4+2]
            mov es:[di+5],ax
            mov es:[di+7],dx
            ;employee
            push ds:[bx]
            pop es:[di+10]
            ;avg
            div word ptr ds:[bx]
            mov es:[di+13],ax

            add si,4
            add di,16
            add bx,2
```

```
430
431             loop input_table_bg
432     input_table_end:
433             pop si
434             pop di
435             pop ds
436             pop es
437             pop dx
438             pop cx
439             pop bx
440             pop ax
441             ret
442
443
444
445 ;==============================初始化寄存器组，把数据导入
446 init_reg:
447             mov bx,data;data in
448             mov ds,bx
449             mov bx,table;data out
450             mov es,bx
451             ret
452
453 ;==================================清空屏幕
454 clear_screen:
455             mov bx,0b800h
456             mov es,bx
457
458             mov bx,0
459             mov dx,0000h
460             mov cx,2000
461
462     clearScreen:
463                     mov es:[bx],dx
464                     add bx,2
465
466                     loop clearScreen
467
468                     ret
469 code ends
470
471 end start
```

2.0.4 截图


DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
1975 16 3 5
1976 22 7 3
1977 382 9 42
1978 1356 13 104
1979 2390 28 85
1980 8000 38 210
1981 16000 130 123
1982 24486 220 111
1983 50065 476 105
1984 97479 778 125
1985 140417 1001 140
1986 197514 1442 136
1987 345980 2258 153
1988 590827 2793 211
1989 803530 4037 199
1990 1183000 5635 209
1991 1843000 8226 224
1992 2759000 11542 239
1993 3753000 14430 260
1994 4649000 15257 304
1995 5937000 17800 333
C:\>_