



廣東工業大學

汇编语言实验报告

- 1、用表格形式显示字符
- 2、字母大小写转换
- 3、课程设计 1 (争优)

学 院	计算机学院
专 业	计算机科学与技术
学 号	3112005816
学生姓名	何柏超
指导教师	张 梅

2014 年 11 月 06 日



计算机学院 计算机科学与技术 专业 2 班 学号 3112005816
姓名 何柏超 协作者 _____ 教师评定 _____
实验题目 用表格形式显示 ASCII 字符

实验 1 用表格形式显示 ASCII 字符

1.1 实验目的与要求

学习用汇编语言设计与编写循环程序。

1.2 实验内容

按 15 行乘 16 列的表格形式显示 ASCII 码为 $10h - 100h$ 的所有字符，即以行为主的顺序及 ASCII 码递增的次序依次显示对应的字符。每 16 个字符为一行，每行中的相邻两格字符之间用空格隔开。

1.3 实验提示

显示每个字符可使用功能号为 $02h$ 的中断调用，使用方法如下：

```
1 mov ah, 02h
2 mov dl, bl ;输出字符的 ASCII 码
3 int 21h
```

本题可把 dl 初始化为 $10h$ ，然后不断使其加 1（用 inc 指令）以取得下一个字符的 ASCII 码。显示空白符时，用其 ASCII 码 0 置入 dl 寄存器。每行结束时，用显示回车（ASCII 为 $0dh$ ）和换行符（ASCII 为 $0ah$ ）来结束本行并开始下一行。

由于逐个显示相继的 ASCII 字符时，需要保存并不断修改 dl 寄存器的内容，而显示空白、回车、换行符时也需要使用 dl 寄存器，为此可使用堆栈来保存相继的 ASCII 字符。具体用法是：在显示空白或回车、换行符前用指令 `push dx` 把 dl 的内容保存到堆栈中去。在显示空白或回车、换行符后用指令 `pop dx` 恢复 dl 寄存器的原始内容。

1.4 程序运行截图

Figure 1: 实验 1 运行截图



1.5 程序代码

```
1 ; -----
2 ; 程序描述：
3 ;
4 ; 输出从 0x010 到 0x100 范围内的 ascii 码表（16 个一行）
5 ; -----
6 assume ds:datasg, cs:codseg
7
8 datasg segment
9
10 ; 以行为输出单位
11 line_buffer db 00, " ", 00, " ", 00, " ", 00, " ", 00, " ", 00, " ", 00, " ", \
12             00, " ", 00, " ", 00, " ", 00, " ", 00, " ", 00, " ", 00, " ", \
13             00, " ", 00, " ", 00, " ", 00, " ", 13, 10, 13, 10, "$"
14
15 datasg ends
16
```

```

17
18 codseg segment
19
20 start:
21     nop
22
23     ; 使用 si 寄存器作为行循环变量
24     mov si, 10h
25
26     ; -----
27     ; 循环填充各行
28     ;
29     ; 修改寄存器: si
30     ; -----
31 fill_lines:
32
33     call fill_line           ; 填充当前行
34     call print_line         ; 输出当前行
35
36     cmp si, 100h
37     jb fill_lines           ; 继续填充下一行
38
39     jmp terminate           ; 结束程序
40
41     ; -----
42     ; 填充缓冲行
43     ;
44     ; 修改寄存器: si
45     ; -----
46 fill_line:
47
48 fill_line_prologue:
49     push bx
50     push cx
51     push ds
52     push dx
53
54 fill_line_main:
55     ; 准备缓冲行地址到 ds:dx 段中
56     mov ax, datasg
57     mov ds, ax
58     lea dx, line_buffer
59     ; 准备循环变量
60     mov bx, 0h               ; 缓冲行修改单元下标
61     mov cx, 10h              ; 循环变量
62
63 fill_line_loop:
64     mov [bx], si             ; 将当前 ascii 字符填充到输出行中
65     inc si                   ; ascii 字符增加 1

```

```

66     add bx, 2                                ; 缓冲行单元下标增加 2( 跳过空格 )
67     loop fill_line_loop
68
69 fill_line_epilogue:
70     pop dx
71     pop ds
72     pop cx
73     pop bx
74     ret
75
76 ; -----
77 ; 打印缓冲行
78 ;
79 ; 修改寄存器：无
80 ; -----
81 print_line:
82
83 print_line_prologue:
84     push ax
85     push ds
86     push dx
87
88 print_line_body:
89     ; 准备缓冲行地址到 ds:dx 段中
90     mov ax, datasg
91     mov ds, ax
92     lea dx, line_buffer
93     mov ah, 09h
94     int 21h
95
96 print_line_epilogue:
97     pop dx
98     pop ds
99     pop ax
100    ret
101
102 ; -----
103 ; 结束程序执行
104 ;
105 ; 修改寄存器： ax
106 ; -----
107 terminate:
108     mov ax, 4c00h
109     int 21h
110
111 codseg ends
112
113 end start

```

广东工业大学

计算机学院 计算机科学与技术 专业 2 班 学号 3112005816
姓名 何柏超 协作者 _____ 教师评定 _____
实验题目 字母大小写改写程序

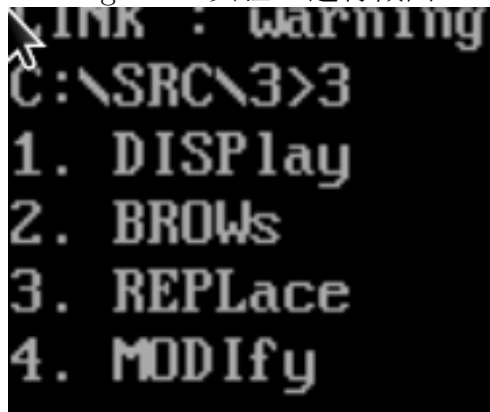
实验 2 字母大小写改写程序

2.1 实验要求

按书中要求编写程序，并将结果显示到屏幕上。

2.2 程序截图

Figure 2: 实验 2 运行截图



2.3 程序代码

```
1 ;-----  
2 ; 程序描述 :  
3 ;  
4 ; 将 datasg 段中每个单词的前 4 个字母改为大写字母  
5 ;-----  
6 assume cs:codesg, ss:stacksg, ds:datasg  
7  
8 stacksg segment  
9     dw 0, 0, 0, 0, 0, 0, 0, 0, 0  
10 stacksg ends
```

```

11
12 datasg segment
13     db '1. display'
14     db '2. brows'
15     db '3. replace'
16     db '4. modify'
17 datasg ends
18
19 codesg segment
20
21 start:
22     nop
23
24     mov cx, 4h      ; 使用 cx 寄存器作为循环变量
25     mov si, 0h
26
27 ; -----
28 ; 循环修改一行
29 ; -----
30 make_uppercase:
31     ; 转换单词
32     mov ax, datasg
33     push ax         ; (a)
34     mov ax, si
35     push ax         ; (o)
36     mov ax, 3       ; 跳过 3 个字符
37     push ax         ; (n)
38     mov ax, 4       ; 转换 4 个字符
39     push ax         ; (m)
40     call update_str
41
42     ; 输出单词
43     mov ax, datasg
44     push ax         ; (a)
45     mov ax, si
46     push ax         ; (o)
47     call print_str
48
49     add si, 10h     ; 处理下一个单词
50     loop make_uppercase
51
52     jmp terminate
53
54 ; -----
55 ; 修改一个字符串的前 m 个字母为大写
56 ;
57 ; 调用参数： 字符串基地址 (a)、字符串偏移地址 (o)、跳过字符数 (n)、转换数目 (m)
58 ; 修改寄存器： 无
59 ; -----

```

```

60 update_str:
61
62 update_str_prologue:
63     push bp
64     mov bp, sp
65
66     push ds
67     mov ds, [bp+10]
68     push bx
69     mov bx, [bp+8]
70
71     push ax
72     push cx
73
74 update_str_body:
75     add bx, [bp+6]      ; 跳过 n 个字符
76     mov cx, [bp+4]     ; 转换 m 个字符
77
78 update_str_loop:
79     mov ax, [bx]
80
81     ; 判断该字符是否为小写字母
82     cmp al, 61h
83     jb update_str_loop_0
84     cmp al, 7ah
85     ja update_str_loop_0
86
87     sub ax, 20h        ; 将小写字母转换为大写
88
89 update_str_loop_0:
90     mov [bx], ax
91     inc bx
92     loop update_str_loop
93
94 update_str_epilogue:
95     pop cx
96     pop ax
97     pop bx
98     pop ds
99     pop bp
100    ret 8
101
102    ; -----
103    ; 打印一个字符串
104    ;
105    ; 调用参数： 字符串基地址 (a)、字符串偏移地址 (o)
106    ; 修改寄存器： 无
107    ; -----
108 print_str:

```



```

109
110 print_str_prologue:
111     push bp
112     mov bp, sp
113     push dx
114     push ds
115     mov dx, [bp+4]
116     mov ds, [bp+6]
117     push ax
118
119 print_str_body:
120     mov ah, 09h
121     int 21h
122
123     ; 同时输出换行
124     mov al, 0dh
125     mov dl, al
126     mov ah, 02h
127     int 21h
128     mov al, 0ah
129     mov dl, al
130     mov ah, 02h
131     int 21h
132
133 print_str_epilogue:
134     pop ax
135     pop ds
136     pop dx
137     pop bp
138     ret 4
139
140 ; -----
141 ; 结束程序执行
142 ;
143 ; 修改寄存器:  ax
144 ; -----
145 terminate:
146     mov ax, 4c00h
147     int 21h
148
149 codesg ends
150
151 end start

```

广东工业大学

计算机学院 计算机科学与技术 专业 2 班 学号 3112005816
姓名 何柏超 协作者 _____ 教师评定 _____
实验题目 课程设计一

实验 3 课程设计一

3.1 实验要求

见书上 P211 页。

3.2 程序代码

```
1 ;-----  
2 ; 程序描述：  
3 ;  
4 ; 格式化输出 Power Idea 公司从 1975 年到 1995 年的收入情况  
5 ;-----  
6 assume cs:codesg, ds:datasg  
7  
8 datasg segment  
9 ; 年份信息 :0000h  
10 dd 1975, 1976, 1977, 1978, 1979, 1980, 1981, 1982,  
1983  
11 dd 1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991,  
1992  
12 dd 1993, 1994, 1995  
13  
14 ; 收入信息 :0054h  
15 dd 16, 22, 382, 1356, 2390, 8000, 16000, 24486,  
50065  
16 dd 97479, 140417, 197514, 345980, 590827, 803530, 1183000, 1843000, 2759000  
17 dd 3753000,4649000,5937000  
18  
19 ; 雇员人数信息 :00a8h  
20 dd 3, 7, 9, 13, 28, 38, 130, 220, 476  
21 dd 778, 1001, 1442, 2258, 2793, 4037, 5635, 8226,  
11542  
22 dd 14430, 15257, 17800  
23  
24 number_buffer db 'aaaaaaaaa$'  
25 datasg ends  
26
```

```

27
28 codesg segment
29
30 start:
31     nop
32
33     ; 建立数据段
34     mov ax, datasg
35     mov ds, ax
36
37     ; 重复 21 次
38     mov cx, 15h
39     jcxz print_year_finish
40 print_year_loop:
41     ; 计算偏移
42     mov di, 15h
43     sub di, cx
44     mov ax, 4
45     mul di
46     mov di, ax
47
48     ; 写入年份
49     mov ax, datasg:[di+0]
50     mov dx, datasg:[di+2]
51     lea si, number_buffer
52     call dtoc
53     call write_number_buffer
54
55     ; 写入制表符
56     mov dl, 09h
57     call write_char
58
59     ; 写入总收入
60     mov ax, datasg:[0054h+di]
61     mov dx, datasg:[0054h+di+2]
62     lea si, number_buffer
63     call dtoc
64     call write_number_buffer
65
66     ; 写入制表符
67     mov dl, 09h
68     call write_char
69
70     ; 写入雇员数
71     mov ax, datasg:[00a8h+di]
72     mov dx, 0h           ; datasg:[00a8h+di+2]
73     lea si, number_buffer
74     call dtoc
75     call write_number_buffer

```

```

76
77 ; 写入制表符
78 mov dl, 09h
79 call write_char
80
81 ; 求平均收入
82 mov ax, datasg:[0054h+di]
83 mov dx, datasg:[0054h+di+2]
84 push cx
85 mov cx, datasg:[00a8h+di]
86 call divdw
87 pop cx
88 ; 写入平均收入
89 lea si, number_buffer
90 call dtoc
91 call write_number_buffer
92
93 ; 写入换行符
94 mov dl, 0dh
95 call write_char
96 mov dl, 0ah
97 call write_char
98
99 loop print_year_loop ; 写入下一年的信息
100
101 print_year_finish:
102 jmp terminate
103
104 ; -----
105 ; 将 number_buffer 写入到终端 (最多 8 位, 右对齐)
106 ;
107 ; 参数: 字符串长度 (bx)
108 ; 返回: 无
109 ; 修改寄存器: 无
110 ; -----
111 write_number_buffer:
112
113 write_number_buffer_prologue:
114 push ax
115 push ds
116 push dx
117 push cx
118 push si
119
120 write_number_buffer_body:
121 mov ax, datasg
122 mov ds, ax
123 lea si, number_buffer ; 字符下标
124 mov cx, bx

```

```

125     jcxz write_number_buffer_epilogue
126
127 write_number_buffer_str:    ; 填充字符串
128     mov dx, [si]
129     call write_char
130     inc si
131     loop write_number_buffer_str
132
133 write_number_buffer_epilogue:
134     pop si
135     pop cx
136     pop dx
137     pop ds
138     pop ax
139     ret
140
141 ; -----
142 ; 显示一个字符到终端
143 ;
144 ; 参数: 显示字符 (dl)
145 ; 返回: 无
146 ; 改变寄存器: 无
147 ; -----
148 write_char:
149
150 write_char_prologue:
151     push ax
152
153 write_char_body:
154     mov ah, 02h
155     int 21h
156
157 write_char_epilogue:
158     pop ax
159     ret
160
161 ; -----
162 ; dword 除法
163 ;
164 ; 参数: 被除数 (dx:ax) 除数 (cx)
165 ; 返回: 商 (dx:ax) 余数 (cx)
166 ; 改变寄存器: 无
167 ; -----
168 divdw:
169
170 divdw_prologue:
171     push si
172
173 divdw_body:

```

```

174 ; 计算  $\text{int}(H/N)$ ,  $\text{rem}(H/N)$ 
175 mov si, ax
176 mov ax, dx
177 mov dx, 0
178 div cx
179 push ax ; 保存  $\text{int}(H/N)$ 
180
181 ; 计算  $(\text{rem}(H/N) * 65536 + L) / N$ 
182 mov ax, si ; L
183 div cx
184 mov cx, dx ; 余数
185 pop dx ; 结果高 16 位
186
187 divdw_epilogue:
188 pop si
189 ret
190
191
192 ; -----
193 ; 将一个字转换成十进制字符串, 以 '\0' 结尾
194 ;
195 ; 参数: 待转换数字 ( $dx:ax$ ) 指向字符串首地址 ( $ds:si$ )
196 ; 返回:  $bx$  转换后的字符串长度
197 ; 改变寄存器: 无
198 ; -----
199 dtoc:
200
201 dtoc_prologue:
202 push dx
203 push ax
204 push cx
205 push si
206
207 dtoc_body:
208 mov bx, 0h ; 记录位数
209
210 dtoc_div:
211 mov cx, 0ah ; 每次除以 10
212 call divdw
213 push cx ; 保存余数
214 inc bx
215 cmp ax, 0
216
217 ; 判断被除数是否为 0
218 jne dtoc_div
219 cmp dx, 0
220 jne dtoc_div
221
222 mov cx, bx ; 将数字转换为字符

```

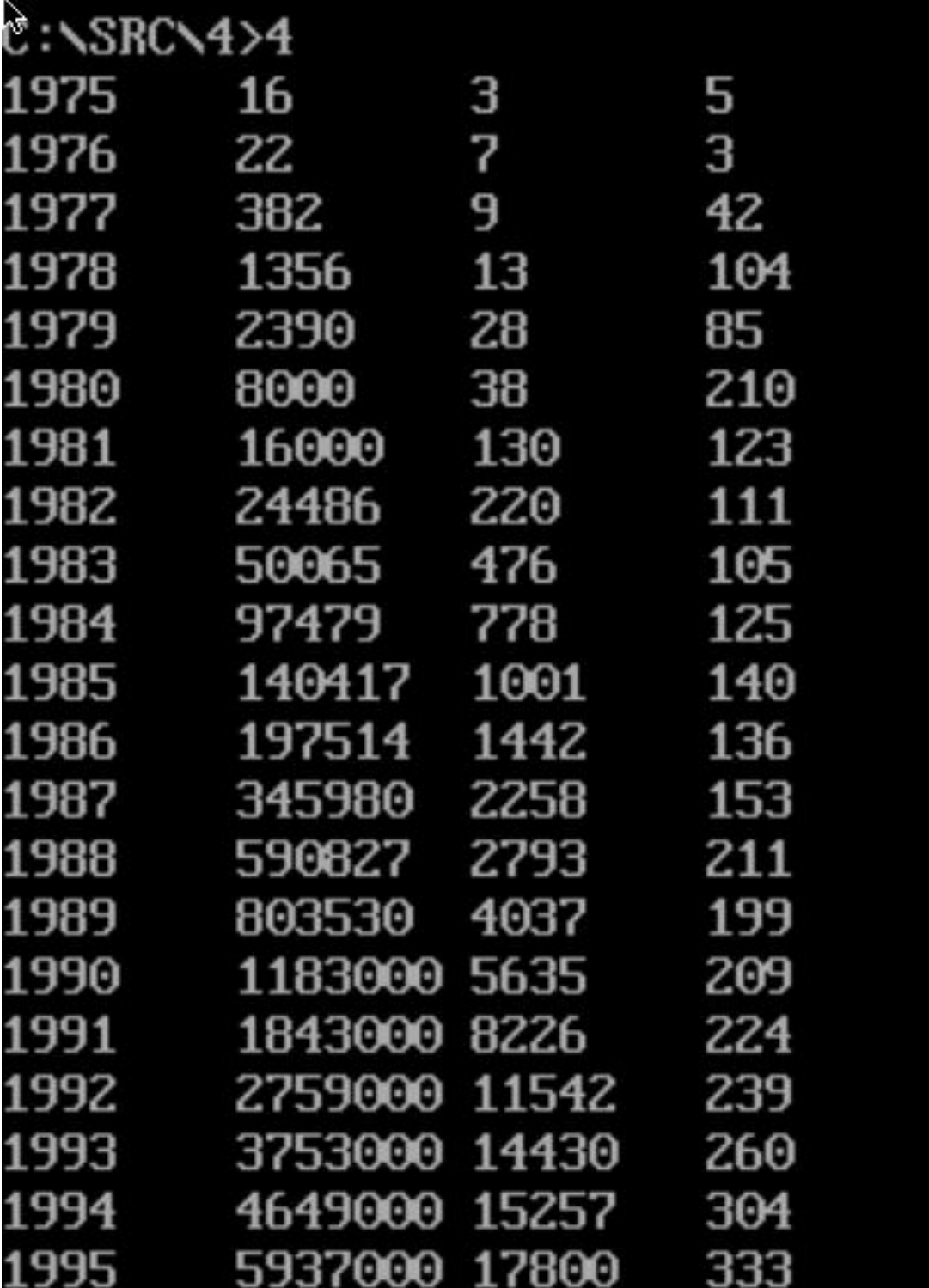
```

223     ; jmp dtoc_to_char
224
225 dtoc_to_char:
226     pop ax
227     add ax, 30h     ; 转换为 ascii 字符
228     mov [si], ax
229     inc si
230     loop dtoc_to_char
231     mov ax, 0h     ; 填充 0 结尾
232     mov [si], ax
233
234 dtoc_epilogue:
235     pop si
236     pop cx
237     pop ax
238     pop dx
239     ret
240
241 ; -----
242 ; 结束程序执行
243 ;
244 ; 修改寄存器:  ax
245 ; -----
246 terminate:
247     mov ax, 4c00h
248     int 21h
249
250 codesg ends
251
252 end start

```

3.3 程序截图

Figure 3: 实验 3 运行截图



A screenshot of a Windows command prompt window. The title bar is partially visible at the top. The prompt is 'C:\SRC\4>'. Below the prompt, a table of data is displayed, consisting of four columns of numbers. The first column contains years from 1975 to 1995. The second, third, and fourth columns contain numerical values corresponding to each year. The text is white on a black background.

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