Московский государственный технический университет
им. Н.Э. Баумана

УТВЕРЖДАЮ:

Большаков С.А.
"__"_____2024 г.

Курсовая работа по курсу «Системное программирование»

Исходный текст программного продукта
(вид документа)

писчая бумага
(вид носителя)

49
(количество листов)

ИСПОЛНИТЕЛИ:

студент группы ИУ5-45Б
Шакиров Т.М.
"__"_____2024 г.

Москва – 2024

Содержание

1. Файл tsr.lst

```
Turbo Assembler	 Version 3.1	    05/08/24 04:21:30	    Page 1
tsr.asm



			     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
			     ; tsr.asm
			     ;
			     ; Сборка:
			     ;	     tasm.exe /l tsr.asm
			     ;	     tlink /t /x tsr.obj
			     ;
			     ;	     МГТУ им. Н.Э. Баумана, ИУ5-45Б,	     2024 г.
			     ;	     Шакиров Т.М
			     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

 0000			     code segment    'code'
				     assume  CS:code, DS:code
				     org     100h
 0100				     _start:

 0100  E9 071B			     jmp _initTSR ; на начало программы

				     ; данные
 0103  A9 E6 E3 AA A5 AD A3+	     replaceWith			     DB
       +
       E8 E9 A7 E5 EA E4 EB+  'йцукенгшщзхъфывапролджёячсмитьбюqwertyuiopasdfghjklzxcvbnm' ; список
игнорируемых	     символов
       A2 A0 AF E0 AE AB A4+
       A6 F1 EF E7 E1 AC A8+
       E2 EC A1 EE 71 77 65+
       72 74 79 75 69 6F 70+
       61 73 64 66 67 68 6A+
       6B 6C 7A 78 63 76 62+
       6E 6D
 013D  89 96 93 8A 85 8D 83+	     ignoredChars		     DB
       +
       98 99 87 95 9A 94 9B+  'ЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЁЯЧСМИТЬБЮQWERTYUIOPASDFGHJKLZXCVBNM'
       82 80 8F 90 8E 8B 84+
       86 F0 9F 97 91 8C 88+
       92 9C 81 9E 51 57 45+
       52 54 59 55 49 4F 50+
       41 53 44 46 47 48 4A+
```

```
36          4B 4C 5A 58 43 56           42+
37          4E 4D
38          =003A                       ignoredLength                   equ     $-ignoredChars ; длина строки     +
39                                      ignoredChars
40      0177  00                       ignoreEnabled                   DB      0           ; флаг функции игнорирования   +
41                                      ввода
42      0178  4B 56 59 4A 47           translateFrom                   DB      'KVYJG' ; символы для замены (ЛМНОП
на +
43                                      англ. раскладке)
44      017D  8B 8C 8D 8E 8F                translateTo                             DB      'ЛМНОП' ; символы на
которые   +
45                                      будет идти      замена'     ; символы на которые будет идти      замена
46          =0005                       translateLength                 equ     $-translateTo   ; длина строки      +
47                                      trasnlateFrom
48      0182  00                       translateEnabled                DB      0           ; флаг функции перевода
49
50      0183  00                       signaturePrintingEnabled        DB      0           ; флаг функции вывода         +
51                                      информации      об авторе
52      0184  00                       cursiveEnabled                  DB      0           ; флаг перевода символа в курсив
53
54      0185  00                 cursiveSymbol       DB 00000000b        ; символ, составленный из единичек (его          курсивный
вариант)
55      0186  00                                       DB 00000000b
56      0187  00                                       DB 00000000b
57      0188  3F                                       DB 00111111b
```

```
   58      0189  33                                DB 00110011b
   59      018A  63                                          DB 01100011b
   60      018B  63                                          DB 01100011b
   61      018C  63                                          DB 01100011b
   62      018D  C6                                DB 11000110b
   63      018E  C6                                DB 11000110b
   64      018F  C6                                          DB 11000110b
   65      0190  C6                                          DB 11000110b
   66      0191  86                                DB 10000110b
   67      0192  00                                DB 00000000b
   68      0193  00                                DB 00000000b
   69      0194  00                                DB 00000000b
   70
   71
   72
   73      0195  8F                  charToCursiveIndex      DB      'П' ; символ для замены
   74      0196  10*(FF)                     savedSymbol                     DB      16 dup(0FFh) ; переменная
для          +
   75                                хранения старого символа
   76
   77        =00FF                   true                                            equ     0FFh ; +
   78                                константа истинности
   79      01A6  ????                        old_int9hOffset                 DW      ?       ; адрес старого
             +
   80                                обработчика int 9h
   81      01A8  ????                        old_int9hSegment                DW      ?       ; сегмент
старого      +
   82                                обработчика int 9h
   83      01AA  ????                        old_int1ChOffset                DW      ?       ; адрес старого
             +
   84                                обработчика int 1Ch
   85      01AC  ????                        old_int1ChSegment               DW      ?       ; сегмент
старого обработчика  +
   86                                int 1Ch
   87      01AE  ????                        old_int2FhOffset                DW      ?       ; адрес старого
             +
   88                                обработчика int 2Fh
   89      01B0  ????                        old_int2FhSegment               DW      ?       ; сегмент
старого обработчика  +
   90                                int 2Fh
   91
```

```
   92    01B2  00                    unloadTSR                                   DB      0 ; 1 -
выгрузить              +

   93                                резидент

   94    01B3  00                    notLoadTSR                          DB      0       ; 1 - не
загружать

   95    01B4  0000                          counter                             DW      0

   96          =0007                 printDelay                          equ     7 ; задержка перед
         выводом     +

   97                                "подписи" в секундах

   98    01B6  0001                          printPos                                    DW      1
          ; положение    +

   99                                подписи на      экране.     0 - верх, 1 - центр, 2 - низ

  100

  101                                       ;@      заменить на собственные     данные.   формирование таблицы идет по
строке большей длины   +

  102                               (1я строка).

  103    01B8  B3 98 A0 AA A8 E0     AE+           signatureLine1              DB      179, 'Шакиров Тимур
          +

  104          A2 20 92 A8 AC E3     E0+  ',      179

  105          20 20 20 20 20 20 20+

  106          20 20 20 20 20 20 20+

  107          20 20 20 20 20 20 20+

  108          20 20 20 20 20 20 20+

  109          20 20 20 20 20 20 20+

  110          20 20 20 B3

  111          =0035                 Line1_length                        equ     $-signatureLine1

  112    01ED  B3 88 93 35 2D 34     35+           signatureLine2              DB      179, 'ИУ5-45Б
          +

  113          81 20 20 20 20 20 20+  ',      179

  114          20 20 20 20 20 20 20+
```

```
115          20 20 20 20 20 20 20+
116          20 20 20 20 20 20 20+
117          20 20 20 20 20 20 20+
118          20 20 20 20 20 20 20+
119          20 20 20 B3
120          =0035                       Line2_length                                    equ     $-signatureLine2
121     0222  B3 82 A0 E0 A8 A0           AD+         signatureLine3                      DB          179, 'Вариант #18
                    +
122          E2 20 23 31 38 20 20+  ',     179
123          20 20 20 20 20 20 20+
124          20 20 20 20 20 20 20+
125          20 20 20 20 20 20 20+
126          20 20 20 20 20 20 20+
127          20 20 20 20 20 20 20+
128          20 20 20 B3
129          =0035                       Line3_length                                    equ     $-signatureLine3
130     0257  3E 74 73 72 2E 63           6F+         helpMsg DB          '>tsr.com [/?] ', 10, 13
131          6D 20 5B 2F 3F 5D           20+
132          0A 0D
133     0267  20 5B 2F 3F 5D 20           2D+                                             DB      ' [/?] - вывод данной справки',
          10, 13
134          20 A2 EB A2 AE A4           20+
135          A4 A0 AD AD AE A9           20+
136          E1 AF E0 A0 A2 AA           A8+
137          0A 0D
138     0285  20 75 6E 6C 6F 61           64+                                             DB      ' unloader.com запрос (Y или N ) -
выгрузка     +
139          65 72 2E 63 6F 6D           20+  резидента из памяти', 10, 13
140          A7 A0 AF E0 AE E1           20+
141          28 59 20 A8 AB A8           20+
142          4E 20 29 20 2D 20           A2+
143          EB A3 E0 E3 A7 AA           A0+
144          20 E0 A5 A7 A8 A4           A5+
145          AD E2 A0 20 A8 A7           20+
146          AF A0 AC EF E2 A8           0A+
147          0D
148     02C5  20 20 46 36 20 20           2D+                                             DB      '  F6  - вывод ФИО и группы по
таймеру в низ                +
149          20 A2 EB A2 AE A4           20+  экрана', 10, 13
150          94 88 8E 20 A8 20           A3+
```

```
151     E0 E3 AF AF EB 20       AF+
152     AE 20 E2 A0 A9 AC       A5+
153     E0 E3 20 A2 20 AD       A8+
154     A7 20 ED AA E0 A0       AD+
155     A0 0A 0D
156     02F9  20 20 46 37 20 20 2D+                     DB      ' F7  - включение и отключения
        курсивного вывода   +
157     20 A2 AA AB EE E7       A5+  русского символа П', 10, 13
158     AD A8 A5 20 A8 20       AE+
159     E2 AA AB EE E7 A5       AD+
160     A8 EF 20 AA E3 E0       E1+
161     A8 A2 AD AE A3 AE       20+
162     A2 EB A2 AE A4 A0       20+
163     E0 E3 E1 E1 AA AE       A3+
164     AE 20 E1 A8 AC A2       AE+
165     AB A0 20 8F 0A 0D
166     033E  20 20 46 38 20 20 2D+                     DB      ' F8  - включение и отключение
        частичной           +
167     20 A2 AA AB EE E7       A5+  русификации клавиатуры(KVYJG -> ЛМНОП)', 10, 13
168     AD A8 A5 20 A8 20       AE+
169     E2 AA AB EE E7 A5       AD+
170     A8 A5 20 E7 A0 E1       E2+
171     A8 E7 AD AE A9 20       E0+
```

```
172          E3 E1 A8 E4 A8 AA          A0+
173          E6 A8 A8 20 AA AB          A0+
174          A2 A8 A0 E2 E3 E0          EB+
175          28 4B 56 59 4A 47          20+
176          2D 3E 20 8B 8C 8D          8E+
177          8F 29 0A 0D
178    038F  20 20 46 39 20 20          2D+                                    DB      '  F9  - включение и отключение
       режима замены          +
179          20 A2 AA AB EE E7          A5+  Прописных на строчные', 10, 13
180          AD A8 A5 20 A8 20          AE+
181          E2 AA AB EE E7 A5          AD+
182          A8 A5 20 E0 A5 A6          A8+
183          AC A0 20 A7 A0 AC          A5+
184          AD EB 20 8F E0 AE          AF+
185          A8 E1 AD EB E5 20          AD+
186          A0 20 E1 E2 E0 AE          E7+
187          AD EB A5 0A 0D
188
189          =017C                      helpMsg_length                  equ  $-helpMsg
190    03D3  8E E8 A8 A1 AA A0          20+           errorParamMsg                     DB          'Ошибка параметров
       коммандной  +
191          AF A0 E0 A0 AC A5          E2+  строки', 10, 13
192          E0 AE A2 20 AA AE          AC+
193          AC A0 AD A4 AD AE          A9+
194          20 E1 E2 E0 AE AA          A8+
195          0A 0D
196          =0025                      errorParamMsg_length    equ     $-errorParamMsg
197
198    03F8  DA 33*(C4) BF                    tableTop                                                    DB          218,
Line1_length-2 dup+
199                                     (196), 191
200          =0035                      tableTop_length                 equ     $-tableTop
201    042D  C0 33*(C4) D9                    tableBottom                         DB          192, Line1_length-2 dup
(196), +
202                                     217
203          =0035                      tableBottom_length              equ  $-tableBottom
204
205                                             ; сообщения
206    0462  90 A5 A7 A8 A4 A5          AD+           installedMsg                      DB        'Резидент загружен!$'
207          E2 20 A7 A0 A3 E0          E3+
```

```
208          A6 A5 AD 21 24
209    0475  90 A5 A7 A8 A4 A5     AD+     alreadyInstalledMsg     DB     'Резидент уже загружен$'
210          E2 20 E3 A6 A5 20     A7+
211          A0 A3 E0 E3 A6 A5     AD+
212          24
213    048B  8D A5 A4 AE E1 E2     A0+     noMemMsg                                       DB
'Недостаточно памяти$'
214          E2 AE E7 AD AE 20     AF+
215          A0 AC EF E2 A8 24
216    049F  8D A5 20 E3 A4 A0     AB+     notInstalledMsg              DB     'Не удалось загрузить резидент$'
217          AE E1 EC 20 A7 A0     A3+
218          E0 E3 A7 A8 E2 EC     20+
219          E0 A5 A7 A8 A4 A5     AD+
220          E2 24
221
222    04BD  90 A5 A7 A8 A4 A5     AD+     removedMsg                                     DB     'Резидент
выгружен'
223          E2 20 A2 EB A3 E0     E3+
224          A6 A5 AD
225          =0011                 removedMsg_length          equ    $-removedMsg
226
227    04CE  8D A5 20 E3 A4 A0     AB+     noRemoveMsg                          DB     'Не удалось выгрузить
резидент'
228          AE E1 EC 20 A2 EB     A3+
```

```
                E0 E3 A7 A8 E2 EC       20+
                E0 A5 A7 A8 A4 A5       AD+
                E2
                =001D                   noRemoveMsg_length      equ     $-noRemoveMsg

04EB  46 36                             f1_txt                          DB      'F6'
04ED  46 37                             f2_txt                          DB      'F7'
04EF  46 38                             f3_txt                          DB      'F8'
04F1  46 39                             f4_txt                          DB      'F9'
                =0002                   fx_length                       equ     $-f4_txt

04F3                                    changeFx proc
04F3  50                                push AX
04F4  53                                push BX
04F5  51                                push CX
04F6  52                                push DX
04F7  55                                push BP
04F8  06                                push ES
04F9  33 DB                                     xor BX, BX

04FB  B4 03                                     mov AH, 03h
04FD  CD 10                                     int 10h
04FF  52                                push DX

0500  0E                                push CS
0501  07                                pop ES

0502                                    _checkF6:
0502  BD 04EBr                                 lea BP, f1_txt
0505  B9 0002                                   mov CX, fx_length
0508  B7 00                                     mov BH, 0
050A  B6 00                                     mov DH, 0
050C  B2 4E                                     mov DL, 78
050E  B8 1301                                   mov AX, 1301h

0511  80 3E 0183r FF                           cmp signaturePrintingEnabled, true
0516  74 07                                    je      _greenF6
```

```
267  0518                                  _redF6:
268  0518  B3 4F                                   mov BL, 01001111b ; red
269  051A  CD 10                                   int 10h
270  051C  EB 08 90                                jmp _checkF7
271
272  051F                                  _greenF6:
273  051F  BD 04EBr                                lea BP, f1_txt
274  0522  B3 2F                                   mov BL, 00101111b ; green
275  0524  CD 10                                   int 10h
276
277  0526                              _checkF7:
278  0526  BD 04EDr                            lea BP, f2_txt
279  0529  B9 0002                             mov CX, fx_length
280  052C  B7 00                               mov BH, 0
281  052E  B6 01                               mov DH, 1
282  0530  B2 4E                               mov DL, 78
283  0532  B8 1301                             mov AX, 1301h
284
285  0535  80 3E 0184r FF                      cmp cursiveEnabled, true
```

```
286  053A  74 07                        je      _greenF7
287
288  053C                               _redF7:
289  053C  B3 4F                            mov BL, 01001111b ; red
290  053E  CD 10                            int 10h
291  0540  EB 05 90                         jmp _checkF8
292
293  0543                               _greenF7:
294  0543  B3 2F                            mov BL, 00101111b ; green
295  0545  CD 10                            int 10h
296
297  0547                           _checkF8:
298  0547  BD 04EFr                     lea BP, f3_txt
299  054A  B9 0002                      mov CX, fx_length
300  054D  B7 00                        mov BH, 0
301  054F  B6 02                        mov DH, 2
302  0551  B2 4E                        mov DL, 78
303  0553  B8 1301                      mov AX, 1301h
304
305  0556  80 3E 0182r FF               cmp translateEnabled, true
306  055B  74 07                        je      _greenF8
307
308  055D                               _redF8:
309  055D  B3 4F                            mov BL, 01001111b ; red
310  055F  CD 10                            int 10h
311  0561  EB 05 90                         jmp _checkF9
312
313  0564                               _greenF8:
314  0564  B3 2F                            mov BL, 00101111b ; green
315  0566  CD 10                            int 10h
316
317  0568                           _checkF9:
318  0568  BD 04F1r                     lea BP, f4_txt
319  056B  B9 0002                      mov CX, fx_length
320  056E  B7 00                        mov BH, 0
321  0570  B6 03                        mov DH, 3
322  0572  B2 4E                        mov DL, 78
323  0574  B8 1301                      mov AX, 1301h
```

```
324
325     0577  80 3E 0177r FF                    cmp ignoreEnabled,          true
326     057C  74 07                             je        _greenF9
327
328     057E                                    _redF9:
329     057E  B3 4F                                       mov BL, 01001111b ; red
330     0580  CD 10                                       int 10h
331     0582  EB 05 90                                    jmp _outFx
332
333     0585                                    _greenF9:
334     0585  B3 2F                                       mov BL, 00101111b ; green
335     0587  CD 10                                       int 10h
336
337     0589                              _outFx:
338     0589  5A                          pop DX
339     058A  B4 02                                 mov AH, 02h
340     058C  CD 10                                 int 10h
341
342     058E  07                          pop ES
```

```
  343   058F  5D                        pop BP
  344   0590  5A                        pop DX
  345   0591  59                        pop CX
  346   0592  5B                        pop BX
  347   0593  58                        pop AX
  348   0594  C3                        ret
  349   0595                           changeFx endp
  350
  351                                  ;новый    обработчик
  352   0595                           new_int9h proc       far
  353                                          ; сохраняем значения всех,изменяемых регистров в стэке
  354   0595  56                        push SI
  355   0596  50                        push AX
  356   0597  53                        push BX
  357   0598  51                        push CX
  358   0599  52                        push DX
  359   059A  06                        push ES
  360   059B  1E                        push DS
  361                                          ; синхронизируем CS и DS
  362   059C  0E                        push CS
  363   059D  1F                        pop     DS
  364
  365   059E  B8 0040                          mov     AX, 40h ; 40h-сегмент,где хранятся       флаги сост-я клавиатуры,
кольц.      +
  366                                  буфер ввода
  367   05A1  8E C0                            mov     ES, AX
  368   05A3  E4 60                            in          AL, 60h ; записываем в AL скан-код нажатой  клавиши
  369
  370
  371                                          ;@      далее -     код для    всех вариантов
  372
  373                                          ;проверка F6-F9
  374   05A5                                   _test_Fx:
  375   05A5  2C 3A                            sub AL, 58          ; в AL теперь номер функциональной клавиши
  376   05A7                                   _F6:
  377   05A7  3C 06                                    cmp AL, 6 ; F6
  378   05A9  75 0A                                    jne _F7
  379   05AB  F6 16 0183r                      not signaturePrintingEnabled
```

```
380  05AF  E8 FF41                          call changeFx
381  05B2  EB 2E 90                         jmp _translate_or_ignore
382  05B5                                   _F7:
383  05B5  3C 07                            cmp AL, 7 ; F7
384  05B7  75 0D                            jne _F8
385  05B9  F6 16 0184r                      not cursiveEnabled
386  05BD  E8 FF33                          call changeFx
387  05C0  E8 01F4                          call setCursive ; перевод символа в курсив и обратно в
зависимости от +
388                                    флага cursiveEnabled
389  05C3  EB 1D 90                         jmp _translate_or_ignore
390  05C6                                   _F8:
391  05C6  3C 08                            cmp AL, 8 ; F8
392  05C8  75 0A                            jne _F9
393  05CA  F6 16 0182r                      not translateEnabled
394  05CE  E8 FF22                          call changeFx
395  05D1  EB 0F 90                         jmp _translate_or_ignore
396  05D4                                   _F9:
397  05D4  3C 09                            cmp AL, 9 ; F9
398  05D6  75 0A                            jne _translate_or_ignore
399  05D8  F6 16 0177r                      not ignoreEnabled
```

```
   400      05DC  E8 FF14                                    call changeFx
   401      05DF  EB 01 90                                   jmp _translate_or_ignore
   402
   403                                                       ;игнорирование и перевод
   404      05E2                                             _translate_or_ignore:
   405
   406      05E2  9C                                 pushf
   407      05E3  2E: FF 1E      01A6r                       call dword          ptr CS:[old_int9hOffset] ; вызываем стандартный
            обработчик прерывания
   408      05E8  B8 0040                                    mov    AX, 40h        ; 40h-сегмент,где хранятся флаги сост-я клавы,кольц.
+
   409                                буфер ввода
   410      05EB  8E C0                                      mov    ES, AX
   411      05ED  26: 8B 1E      001C                        mov    BX, ES:[1Ch]   ; адрес хвоста
   412      05F2  4B                                 dec    BX             ; сместимся назад к последнему
   413      05F3  4B                                 dec    BX             ; введённому символу
   414      05F4  83 FB 1E                                   cmp    BX, 1Eh ; не вышли ли мы за пределы буфера?
   415      05F7  73 03                                      jae    _go
   416      05F9  BB 003C                                    mov    BX, 3Ch ; хвост вышел за пределы буфера, значит последний
введённый    +
   417                                символ
   418                                                                        ; находится в конце буфера
   419
   420      05FC                                     _go:
   421      05FC  26: 8B 17                                  mov DX, ES:[BX] ; в DX 0 введённыйсимвол
   422                                                       ;включен ли режим блокировки ввода?
   423      05FF  80 3E 0177r FF                             cmp ignoreEnabled,          true
   424      0604  75 1E                                      jne _check_translate
   425
   426                                                       ; да, включен
   427      0606  BE 0000                                    mov SI, 0
   428      0609  B9 003A                                    mov CX, ignoredLength ;кол-во игнорируемых символов
   429
   430                                                       ; проверяем, присутствует ли текущий символ в списке игнорируемых
   431      060C                                     _check_ignored:
   432      060C  3A 94 013Dr                        cmp DL,ignoredChars[SI]
   433      0610  74 06                                      je     _block
   434      0612  46                                 inc SI
   435      0613  E2 F7                              loop _check_ignored
```

```
436      0615  EB 0D 90                                jmp _check_translate
437
438                                          ; блокируем
439      0618                                _block:
440                                                  ;mov ES:[1Ch], BX ;блокировка ввода символа
441                                                  ;@      если по     варианту нужно не блокировать ввод символа,
442                                                  ;@      а заменять одни      символы  другими,
443                                                  ;@      замените строку      выше строкой
444                                                  ;mov ES:[BX], AX
445                                                  ;@      на месте AX может быть '*' для замены всех символов множества
ignoredChars  +
446                                  на      звёздочки
447                                                  ;@      или, для перевода одних          символов в другие - завести массив
448                                                  ;@      replaceWith DB '...', где перечислить символы, на которые пойдёт
замена
449                                                  ;@      и раскомментировать строки ниже:
450      0618  33 C0                                   xor AX, AX
451      061A  8A 84 0103r                        mov AL, replaceWith[SI]
452      061E  26: 89 07                              mov ES:[BX], AX                 ; замена символа
453      0621  EB 23 90                               jmp _quit
454
455      0624                                _check_translate:
456                                                  ; включен ли режим           перевода?
```

```
0624  80 3E 0182r FF                    cmp translateEnabled, true
0629  75 1B                            jne _quit

                                       ; да, включен
062B  BE 0000                          mov SI, 0
062E  B9 0005                          mov CX, translateLength ; кол-во символов для перевода
                                       ; проверяем, присутствует ли текущий символ в списке для перевода
0631                                   _check_translate_loop:
0631  3A 94 0178r                      cmp DL, translateFrom[SI]
0635  74 06                                    je      _translate
0637  46                              inc SI
0638  E2 F7                            loop _check_translate_loop
063A  EB 0A 90                         jmp _quit

                                       ; переводим
063D                                   _translate:
063D  33 C0                                    xor AX, AX
063F  8A 84 017Dr                      mov AL, translateTo[SI]
0643  26: 89 07                                mov ES:[BX], AX ; замена символа

0646                           _quit:
                                       ; восстанавливаем все регистры
0646  1F                       pop     DS
0647  07                       pop     ES
0648  5A                       pop DX
0649  59                       pop CX
064A  5B                       pop     BX
064B  58                       pop     AX
064C  5E                       pop SI
064D  CF                       iret
064E                   new_int9h endp

                       ;=== Обработчик прерывания          int 1Ch     ===;
                       ;=== Вызывается каждые 55 мс ===;
064E                   new_int1Ch          proc far
064E  50               push AX
064F  0E               push CS
0650  1F               pop DS
```

```
495
496     0651  9C                        pushf
497     0652  2E: FF 1E       01AAr              call dword          ptr CS:[old_int1ChOffset]
498
499     0657  80 3E 0183r FF                     cmp signaturePrintingEnabled, true     ; если нажата управляющая клавиша (в
данном случае  +
500                                      F6)
501     065C  75 1D                              jne _notToPrint
502
503     065E  81 3E 01B4r 0080                           cmp counter, printDelay*1000/55 + 1 ; если         кол-во "тактов"
        эквивалентно            +
504                                      %printDelay% секундам
505     0664  74 03                              je         _letsPrint
506
507     0666  EB 0E 90                           jmp _dontPrint
508
509     0669                                     _letsPrint:
510     0669  F6 16 0183r                        not signaturePrintingEnabled
511     066D  C7 06 01B4r 0000                             mov counter, 0
512     0673  E8 0094                                      call printSignature
513
```

```
0676                                    _dontPrint:
0676  83 06 01B4r 01                    add counter, 1

067B                                _notToPrint:

067B  58                        pop AX

067C  CF                        iret
067D                            new_int1Ch      endp

                                ;=== Обработчик прерывания      int 2Fh     ===;
                                ;=== Служит для:
                                ;===  1) проверки факта присутствия TSR в памяти (при AH=0FFh, AL=0)
                                ;===     будет возвращён AH='i' в случае, если TSR      уже загружен
                                ;===  2) выгрузки TSR из памяти (при AH=0FFh, AL=1)
                                ;===
067D                            new_int2Fh      proc
067D  80 FC FF                          cmp     AH, 0FFh      ;наша функция?
0680  75 0B                             jne     _2Fh_std      ;нет - на старый обработчик
0682  3C 00                             cmp     AL, 0   ;подфункция проверки, загружен ли резидент      в память?
0684  74 0C                             je          _already_installed
0686  3C 01                             cmp     AL, 1   ;подфункция выгрузки из памяти?
0688  74 0B                             je          _uninstall
068A  EB 01 90                          jmp     _2Fh_std      ;нет - на старый обработчик

068D                            _2Fh_std:
068D  2E: FF 2E      01AEr              jmp     dword ptr CS:[old_int2FhOffset] ;вызов старого обработчика

0692                            _already_installed:
0692  B4 69                                     mov     AH, 'i' ;вернём 'i', если резидент      загружен     в память
0694  CF                                iret

0695                            _uninstall:
0695  1E                        push    DS
0696  06                        push    ES
0697  52                        push    DX
0698  53                        push    BX
```

```
0699  33 DB                          xor BX, BX

                                     ; CS = ES,          для доступа к переменным
069B  0E                       push CS
069C  07                       pop ES

069D  B8 2509                        mov     AX, 2509h
06A0  26: 8B 16      01A6r           mov DX, ES:old_int9hOffset                  ; возвращаем вектор прерывания
06A5  26: 8E 1E      01A8r         mov DS, ES:old_int9hSegment    ; на место
06AA  CD 21                          int     21h

06AC  B8 251C                        mov     AX, 251Ch
06AF  26: 8B 16      01AAr           mov DX, ES:old_int1ChOffset          ; возвращаем вектор прерывания
06B4  26: 8E 1E      01ACr         mov DS, ES:old_int1ChSegment            ; на место
06B9  CD 21                          int     21h

06BB  B8 252F                        mov     AX, 252Fh
06BE  26: 8B 16      01AEr           mov DX, ES:old_int2FhOffset          ; возвращаем вектор прерывания
06C3  26: 8E 1E      01B0r         mov DS, ES:old_int2FhSegment            ; на место
```

```
571  06C8  CD 21                    int   21h
572
573  06CA  2E: 8E 06   002C         mov   ES, CS:2Ch          ; загрузим в ES адрес окружения
574  06CF  B4 49                    mov   AH, 49h      ; выгрузим из памяти окружение
575  06D1  CD 21                    int   21h
576  06D3  72 0B                    jc    _notRemove
577
578  06D5  0E                  push  CS
579  06D6  07                  pop   ES           ;в ES - адрес резидентной программы
580  06D7  B4 49                    mov   AH, 49h  ;выгрузим из памяти резидент
581  06D9  CD 21                    int   21h
582  06DB  72 03                    jc    _notRemove
583  06DD  EB 15 90                 jmp _unloaded
584
585  06E0                      _notRemove: ; не удалось выполнить выгрузку
586                                 ; вывод сообщения о неудачной выгрузке
587  06E0  B4 03                    mov AH, 03h                ; получаем позицию
     курсора
588  06E2  CD 10                    int 10h
589  06E4  BD 04CEr                 lea BP, noRemoveMsg
590  06E7  B9 001D                  mov CX, noRemoveMsg_length
591  06EA  B3 07                    mov BL, 0111b
592  06EC  B8 1301                  mov AX, 1301h
593  06EF  CD 10                    int 10h
594  06F1  EB 12 90                 jmp _2Fh_exit
595
596  06F4                      _unloaded:      ; выгрузка прошла успешно
597                                 ; вывод сообщения об удачной выгрузке
598  06F4  B4 03                    mov AH, 03h                ; получаем позицию
     курсора
599  06F6  CD 10                    int 10h
600  06F8  BD 04BDr                 lea BP, removedMsg
601  06FB  B9 0011                  mov CX, removedMsg_length
602  06FE  B3 07                    mov BL, 0111b
603  0700  B8 1301                  mov AX, 1301h
604  0703  CD 10                    int 10h
605
606  0705                      _2Fh_exit:
607  0705  5B                  pop BX
```

```
608   0706  5A                pop    DX
609   0707  07                pop    ES
610   0708  1F                pop    DS
611   0709  CF                iret
612   070A                    new_int2Fh      endp
613
614                           ;=== Процедура вывода подписи (ФИО, группа)
615                           ;=== Настраивается      значениями переменных в     начале исходника
616                           ;===
617   070A                    printSignature proc
618   070A  50                push AX
619   070B  52                push DX
620   070C  51                push CX
621   070D  53                push BX
622   070E  06                push ES
623   070F  54                push SP
624   0710  55                push BP
625   0711  56                push SI
626   0712  57                push DI
627
```

```
   628      0713  33 C0                    xor AX, AX
   629      0715  33 DB                    xor BX, BX
   630      0717  33 D2                    xor DX, DX
   631
   632      0719  B4 03                    mov AH, 03h                    ;чтение текущей
позиции курсора
   633      071B  CD 10                    int 10h
   634      071D  52                  push DX                             ;помещаем информацию
o              +
   635                                    положении курсора в стек
   636
   637      071E  83 3E 01B6r 00           cmp printPos, 0
   638      0723  74 0E                    je      _printTop
   639
   640      0725  83 3E 01B6r 01           cmp printPos, 1
   641      072A  74 0E                    je      _printCenter
   642
   643      072C  83 3E 01B6r 02           cmp printPos, 2
   644      0731  74 0E                    je      _printBottom
   645
   646                                    ;все числа      подобраны на глаз...
   647      0733                          _printTop:
   648      0733  B6 00                          mov DH, 0
   649      0735  B2 0F                          mov DL, 15
   650      0737  EB 0F 90                       jmp _actualPrint
   651
   652      073A                          _printCenter:
   653      073A  B6 09                          mov DH, 9
   654      073C  B2 0F                          mov DL, 15
   655      073E  EB 08 90                       jmp _actualPrint
   656
   657      0741                          _printBottom:
   658      0741  B6 13                          mov DH, 19
   659      0743  B2 0F                          mov DL, 15
   660      0745  EB 01 90                       jmp _actualPrint
   661
   662      0748                          _actualPrint:
   663      0748  B4 0F                          mov AH, 0Fh                ;чтение текущего
видеорежима. в+
```

```
664                                 BH      - текущая страница
665     074A  CD 10                         int 10h
666
667     074C  0E                    push CS
668     074D  07                    pop ES                          ;указываем      ES на CS
669
670                                 ;вывод 'верхушки' таблицы
671     074E  52                    push DX
672     074F  BD 03F8r                      lea BP, tableTop                        ;помещаем в BP
указатель на    +
673                                 выводимую строку
674     0752  B9 0035                       mov CX, tableTop_length     ;в      CX - длина строки
675     0755  B3 07                         mov BL, 0111b                   ;цвет выводимого текста ref:
          +
676                                 http://en.wikipedia.org/wiki/BIOS_color_attributes
677     0757  B8 1301                       mov AX, 1301h                           ;AH=13h - номер ф-ии,
AL=01h - +
678                                 курсор перемещается при выводе каждого из символов  строки
679     075A  CD 10                         int 10h
680     075C  5A                    pop DX
681     075D  FE C6                         inc DH
682
683
684                                 ;вывод первой линии
```

```
685  075F  52                    push DX
686  0760  BD 01B8r                   lea BP, signatureLine1
687  0763  B9 0035                    mov CX, Line1_length
688  0766  B3 07                      mov BL, 0111b
689  0768  B8 1301                    mov AX, 1301h
690  076B  CD 10                      int 10h
691  076D  5A                    pop DX
692  076E  FE C6                      inc DH
693
694                                   ;вывод второй линии
695  0770  52                    push DX
696  0771  BD 01EDr                   lea BP, signatureLine2
697  0774  B9 0035                    mov CX, Line2_length
698  0777  B3 07                      mov BL, 0111b
699  0779  B8 1301                    mov AX, 1301h
700  077C  CD 10                      int 10h
701  077E  5A                    pop DX
702  077F  FE C6                      inc DH
703
704                                   ;вывод третьей линии
705  0781  52                    push DX
706  0782  BD 0222r                   lea BP, signatureLine3
707  0785  B9 0035                    mov CX, Line3_length
708  0788  B3 07                      mov BL, 0111b
709  078A  B8 1301                    mov AX, 1301h
710  078D  CD 10                      int 10h
711  078F  5A                    pop DX
712  0790  FE C6                      inc DH
713
714                                   ;вывод 'низа' таблицы
715  0792  52                    push DX
716  0793  BD 042Dr                   lea BP, tableBottom
717  0796  B9 0035                    mov CX, tableBottom_length
718  0799  B3 07                      mov BL, 0111b
719  079B  B8 1301                    mov AX, 1301h
720  079E  CD 10                      int 10h
721  07A0  5A                    pop DX
722  07A1  FE C6                      inc DH
```

```
723
724     07A3  33 DB                     xor BX, BX
725     07A5  5A                pop DX                  ;восстанавливаем из
стека     +
726                             прежнее положение курсора
727     07A6  B4 02                     mov AH, 02h             ;меняем положение
курсора на   +
728                             первоначальное
729     07A8  CD 10                     int 10h
730     07AA  E8 FD46                   call changeFx
731
732     07AD  5F                pop DI
733     07AE  5E                pop SI
734     07AF  5D                pop BP
735     07B0  5C                pop SP
736     07B1  07                pop ES
737     07B2  5B                pop BX
738     07B3  59                pop CX
739     07B4  5A                pop DX
740     07B5  58                pop AX
741
```

```
742  07B6  C3                    ret
743  07B7                        printSignature endp
744
745                              ;=== Функция, которая в зависимости от флага cursiveEnabled меняет начертание символа с курсива+
746                              на обычное и наоброт
747                              ;=== Сама смена происходит в процедуре changeFont, а здесь подготавливаются данные
748  07B7                        setCursive proc
749  07B7  06                    push ES ; сохраняем регистры
750  07B8  50                    push AX
751  07B9  0E                    push CS
752  07BA  07                    pop ES
753
754  07BB  80 3E 0184r FF                cmp cursiveEnabled, true
755  07C0  75 30                         jne _restoreSymbol
756                                      ; если флаг равен true, выполняем замену символа на курсивный вариант,
757                                      ; предварительно сохраняя старый символ в savedSymbol
758
759  07C2  E8 004C                       call saveFont
760  07C5  8A 0E 0195r           mov CL, charToCursiveIndex
761  07C9                        _shifTtable:
762                                      ; мы получаем в BP таблицу всех символов. адрес указывает на символ 0
763                                      ; поэтому нуэно совершить сдвиг 16*X - где X - код символа
764  07C9  83 C5 10                      add BP, 16
765  07CC  E2 FB                         loop _shiftTable
766
767                                      ; при savefont смещается регистр ES
768                                      ; поэтому приходится делать такие махинации, чтобы
769                                      ; записать полученный элемент в savedSymbol
770                                      ; swap(ES, DS) и сохранение старого значения DS
771  07CE  1E                    push DS
772  07CF  58                    pop AX
773  07D0  06                    push ES
774  07D1  1F                    pop DS
775  07D2  50                    push AX
776  07D3  07                    pop ES
777  07D4  50                    push AX
778
```

```
07D5  8B F5                     mov SI, BP
07D7  BF 0196r                  lea DI, savedSymbol
                                ; сохраняем в переменную savedSymbol
                                ; таблицу нужного символа
07DA  B9 0010                   mov CX, 16
                                ; movsb из      DS:SI в    ES:DI
07DD  F3> A4                    rep movsb
                                ; исходные      позиции   сегментов возвращены
07DF  1F                    pop DS ; восстановление DS

                                ; заменим написание символа на курсив
07E0  B9 0001                   mov CX, 1
07E3  B6 00                     mov DH, 0
07E5  8A 16 0195r           mov DL, charToCursiveIndex
07E9  BD 0185r                  lea BP, cursiveSymbol
07EC  E8 0015                   call changeFont
07EF  EB 10 90                  jmp _exitSetCursive

07F2                        _restoreSymbol:
                                ; если флаг равен 0, выполняем замену курсивного символа на старый
вариант
```

```
                                
07F2  B9 0001                   mov CX, 1
07F5  B6 00                     mov DH, 0
07F7  8A 16 0195r               mov DL, charToCursiveIndex
07FB  BD 0196r                  lea bp, savedSymbol
07FE  E8 0003                   call changeFont
                                
0801                            _exitSetCursive:
0801  58                        pop AX
0802  07                        pop ES
0803  C3                        ret
0804                            setCursive      endp
                                
                                ;=== Функция смены      начертания символа (курсив/нормальное)
                                ;===
                                ; *** входные данные
                                ; DL = номер символа для замены
                                ; CX = Кол-во символов заменяемых изображений символов
                                ; (начиная      с символа указанного в DX)
                                ; ES:bp = адрес таблицы
                                ;
                                ; *** описание работы процедуры
                                ; Происходит вызов      int 10h   (видеосервис)
                                ; с функцией AH = 11h (функции знакогенератора)
                                ; Параметр      AL = 0 сообщает, что будет заменено изображение
                                ; символа для текущего шрифта
                                ; В случаях, когда      AL = 1 или 2, будет заменено изображение
                                ; только для опредленного шрифта (8x14 и 8x8 соответственно)
                                ; Параметр      BH = 0Eh сообщает, что на опредление каждого изображения символа
                                ; расходуется по 14 байт (режим 8x14 бит как раз 14 байт)
                                ; Параметр      BL = 0 - блок шрифта для загрузки (от 0         до 4)
                                ;
                                ; *** результат
                                ; изображение указанного(ых) символа(ов) будет заменено
                                ; на предложенное пользователем.
                                ; Изменению подвергнутся все символы, находящиеся на экране,
                                ; то есть если изображение заменено, старый вариант нигде уже не проявится
                                
```

```
0804                    changeFont      proc
0804  50                push AX
0805  53                push BX
0806  B8 1100                  mov AX, 1100h
0809  BB 1000                  mov BX, 1000h
080C  CD 10                    int 10h
080E  58                pop AX
080F  5B                pop BX
0810  C3                ret
0811                    changeFont      endp

                        ;=== Функция сохранения нормального начертания символа
                        ;===
                        ; *** входные данные
                        ; BH - тип      возвращаемой символьной  таблицы
                        ;       0 - таблица из  int 1fh
                        ;       1 - таблица из  int 44h
                        ;       2-5 - таблица из 8x14,  8x8, 8x8 (top),  9x14
                        ;       6 - 8x16
```

```
                                ;
                                ; *** описание работы процедуры
                                ; Происходит вызов int 10h (видеосервис)
                                ; с функцией AH = 11h (функции знакогенератора)
                                ; Параметр AL = 30 - подфункция получения информации о EGA
                                ;
                                ; *** результат
                                ; в ES:BP находится таблица символов (полная)
                                ; в CX находится байт на символ
                                ; в DL количество экранных строк
                                ; ВАЖНО! Происходит сдвиг регистра ES
                                ; ( ES становится равным C000h )

0811                            saveFont proc
0811  50                        push AX
0812  53                        push BX
0813  B8 1130                        mov AX, 1130h
0816  BB 0600                        mov BX, 0600h
0819  CD 10                        int 10h
081B  58                        pop AX
081C  5B                        pop BX
081D  C3                        ret
081E                            saveFont endp


                                ;=== Отсюда начинается выполнение основной части программы ===;
                                ;===
081E                            _initTSR: ; старт резидента
081E  B4 03                        mov AH, 03h
0820  CD 10                        int 10h
0822  52                        push DX
0823  B4 00                        mov AH,00h ; установка видеорежима (83h текст 80x25 16/8 CGA,EGA
b800  Comp,RGB, +
                                Enhanced), без очистки экрана
0825  B0 83                        mov AL,83h
0827  CD 10                        int 10h
0829  5A                        pop DX
082A  B4 02                        mov AH, 02h
```

```
082C  CD 10                     int 10h

082E  E8 00B3                   call commandParamsParser
0831  B8 3509                     mov AX,3509h                  ; получить          в ES:BX   вектор 09
0834  CD 21                     int 21h                         ; прерывания

                                ;@      === Удаление резидента из памяти ===
                                ;@      Если по    варианту необходимо выгружать резидент по повторному запуску приложений,
                                ;@      нужно закомментировать следующие 3 строки, а также
                                ;@      содержимое метки _finishTSR ф-ии commandParamsParser, но не саму метку!
0836  80 3E 01B2r FF            cmp unloadTSR, true
083B  74 03                     je      _removingOnParameter
083D  EB 15 90                  jmp _notRemovingNow

0840                            _removingOnParameter:
0840  B4 FF                             mov AH, 0FFh
0842  B0 00                             mov AL, 0
0844  CD 2F                             int 2Fh
0846  80 FC 69                          cmp AH, 'i'  ; проверка того, загружена ли        уже программа
```

```
   913      0849  74 7D                              je        _remove
   914      084B  B4 09                              mov AH, 09h                    ;@      для выгрузки резидента по
повторному+
   915                                  запуску закомментировать эту строку
   916      084D  BA 049Fr                           lea DX, notInstalledMsg ;@ для выгрузки резидента по повторному
запуску      +
   917                                  закомментировать эту строку
   918      0850  CD 21                              int 21h                        ;@      для выгрузки резидента по
повторному+
   919                                  запуску закомментировать эту строку
   920      0852  CD 20                              int 20h                        ;@      для выгрузки резидента по
повторному+
   921                                  запуску закомментировать эту строку
   922
   923      0854                                    _notRemovingNow:
   924
   925      0854  80 3E 01B3r FF                    cmp notLoadTSR, true           ; если была выведена справка
   926      0859  74 0E                              je        _exit_tmp                                ; просто выходим
   927
   928                                              ;@      Если по    варианту необходимо выгружать резидент по повторному запуску,
то           +
   929                                  комментируем 5 строк ниже
   930                                              ;@      если необходимо     выгружать по параметру коммандной строки, то
оставляем их
   931      085B  B4 FF                              mov AH, 0FFh
   932      085D  B0 00                              mov AL, 0
   933      085F  CD 2F                              int 2Fh
   934      0861  80 FC 69                           cmp AH, 'i'  ; проверка того, загружена ли        уже программа
   935      0864  74 6B                              je        _alreadyInstalled
   936
   937      0866  EB 04 90                           jmp _tmp
   938
   939      0869                                    _exit_tmp:
   940      0869  EB 77 90                                    jmp _exit
   941
   942      086C                                    _tmp:
   943      086C  06                            push ES
   944      086D  A1 002C                                mov AX, DS:[2Ch]           ; psp
   945      0870  8E C0                                  mov ES, AX
   946      0872  B4 49                                  mov AH, 49h                        ; хватит памяти чтоб остаться
   947      0874  CD 21                                  int 21h                    ; резидентом?
```

```
0876  07                    pop ES
0877  72 62                 jc _notMem                    ; не хватило -  выходим

                            ;== int 09h ==;

0879  2E: 89 1E  01A6r      mov   word ptr CS:old_int9hOffset, BX
087E  2E: 8C 06  01A8r      mov   word ptr CS:old_int9hSegment, ES
0883  B8 2509               mov AX, 2509h                 ; установим вектор на 09
0886  BA 0595r              mov DX, offset  new_int9h  ; прерывание
0889  CD 21                 int 21h

                            ;== int 1Ch ==;
088B  B8 351C               mov AX,351Ch                  ; получить  в ES:BX  вектор 1C
088E  CD 21                 int 21h                       ; прерывания
0890  2E: 89 1E  01AAr      mov   word ptr CS:old_int1ChOffset, BX
0895  2E: 8C 06  01ACr      mov   word ptr CS:old_int1ChSegment, ES
089A  B8 251C               mov AX, 251Ch                 ; установим вектор  на 1C
089D  BA 064Er              mov DX, offset new_int1Ch     ; прерывание
08A0  CD 21                 int 21h

                            ;== int 2Fh ==;
08A2  B8 352F               mov AX,352Fh                  ; получить  в ES:BX  вектор 1C
```

```
  970    08A5  CD 21                     int 21h                         ; прерывания
  971    08A7  2E: 89 1E    01AEr          mov     word ptr CS:old_int2FhOffset, BX
  972    08AC  2E: 8C 06    01B0r          mov     word ptr CS:old_int2FhSegment, ES
  973    08B1  B8 252F                     mov AX, 252Fh                   ; установим вектор          на 2F
  974    08B4  BA 067Dr                    mov DX, offset new_int2Fh               ; прерывание
  975    08B7  CD 21                       int 21h
  976
  977    08B9  E8 FC37                     call changeFx
  978    08BC  BA 0462r                  mov DX, offset      installedMsg              ; выводим что все ок
  979    08BF  B4 09                     mov AH, 9
  980    08C1  CD 21                     int 21h
  981    08C3  BA 081Er                  mov DX, offset      _initTSR    ; остаемся в памяти резидентом
  982    08C6  CD 27                     int 27h                         ; и выходим
  983                                    ; конец основной программы
  984    08C8                          _remove: ;          выгрузка программы из памяти
  985    08C8  B4 FF                       mov AH, 0FFh
  986    08CA  B0 01                       mov AL, 1
  987    08CC  CD 2F                       int 2Fh
  988    08CE  EB 12 90                    jmp _exit
  989    08D1                          _alreadyInstalled:
  990    08D1  B4 09                       mov AH, 09h
  991    08D3  BA 0475r                    lea DX, alreadyInstalledMsg
  992    08D6  CD 21                       int 21h
  993    08D8  EB 08 90                    jmp _exit
  994    08DB                          _notMem:                                ; не хватает памяти, чтобы остаться
резидентом
  995    08DB  BA 048Br                    mov DX, offset      noMemMsg
  996    08DE  B4 09                       mov AH, 9
  997    08E0  CD 21                       int 21h
  998    08E2                          _exit:                          ; выход
  999    08E2  CD 20                       int 20h
 1000
 1001                                   ;=== Процедура проверки параметров          ком. строки ===;
 1002                                   ;===
 1003    08E4                          commandParamsParser proc
 1004    08E4  0E                      push CS
 1005    08E5  07                      pop ES
 1006    08E6  C6 06 01B2r 00              mov unloadTSR, 0
```

```
1007    08EB  C6 06 01B3r 00              mov notLoadTSR, 0

1008

1009    08F0  BE 0080                    mov SI, 80h                ;SI=смещение командной строки.

1010    08F3  AC                lodsb                               ;Получим кол-во символов.

1011    08F4  0A C0                      or      AL, AL                      ;Если 0 символов введено,

1012    08F6  74 40                      jz      _exitHelp          ;то все в порядке.

1013

1014    08F8                             _nextChar:

1015

1016    08F8  46                inc SI                              ;Теперь SI          указывает на первый
символ  +

1017                            строки.

1018

1019    08F9  80 3C 0D                   cmp [SI], BYTE ptr         13

1020    08FC  74 3A                      je      _exitHelp

1021

1022

1023    08FE  AD                         lodsw                      ;Получаем два символа

1024    08FF  3D 3F2F                            cmp AX, '?/'                ;Это '/?' (данные расположены в
          +

1025                            обратном порядк, т.е. AL:AH вместо AH:AL)

1026    0902  74 03                              je      _question
```

```
1027                                    ;cmp AX, 'u/'
1028                                    ;je _finishTSR
1029
1030                                    ;cmp AH, '/'
1031                                    ;je _errorParam
1032
1033    0904  EB 32 90                  jmp _exitHelp
1034
1035    0907                    _question:
1036                                    ; вывод строки помощи
1037    0907  B4 03                             mov AH,03
1038    0909  CD 10                             int 10h
1039    090B  BD 0257r                          lea BP, helpMsg
1040    090E  B9 017C                           mov CX, helpMsg_length
1041    0911  B3 07                             mov BL, 0111b
1042    0913  B8 1301                           mov AX, 1301h
1043    0916  CD 10                             int 10h
1044                                    ; конец вывода строки помощи
1045    0918  F6 16 01B3r               not notLoadTSR  ;флаг того, что необходимо не загружать резидент
1046    091C  EB DA                     jmp _nextChar
1047
1048                                    ;@      === Удаление резидента из памяти ===
1049                                    ;@      Если по  варианту необходимо выгружать резидент по параметру '/u'
коммандной строки,
1050                                    ;@      нужно использовать следующий код, в остальных случаях необходимо
закомменитровать
1051                                    ;@      этот код, кроме  названия метки!  (по желанию можно избавиться и от
метки, но  +
1052                            аккуратно просмотреть использование)
1053    091E                    _finishTSR:
1054    091E  F6 16 01B2r               not unloadTSR           ;флаг того, что необходимо выгузить резидент
1055    0922  EB D4                     jmp _nextChar
1056
1057    0924  EB 12 90                  jmp _exitHelp
1058
1059    0927                    _errorParam:
1060                                    ;вывод строки
1061    0927  B4 03                             mov AH,03
1062    0929  CD 10                             int 10h
```

```
1063    092B  BD 03D3r                                  lea BP, CS:errorParamMsg
1064    092E  B9 0025                                    mov CX, errorParamMsg_length
1065    0931  B3 07                                      mov BL, 0111b
1066    0933  B8 1301                                    mov AX, 1301h
1067    0936  CD 10                                      int 10h
1068                                                ;конец вывода строки
1069    0938                                  _exitHelp:
1070    0938  C3                        ret
1071    0939                            commandParamsParser endp
1072
1073    0939                            code ends
1074                                    end _start
```

Symbol Table

Symbol Name	Type	Value
??DATE	Text	"05/08/24"
??FILENAME	Text	"tsr "
??TIME	Text	"04:21:29"
??VERSION	Number	030A
@CPU	Text	0101H
@CURSEG	Text	CODE
@FILENAME	Text	TSR
@WORDSIZE	Text	2
ALREADYINSTALLEDMSG	Byte	CODE:0475
CHANGEFONT	Near	CODE:0804
CHANGEFX	Near	CODE:04F3
CHARTOCURSIVEINDEX	Byte	CODE:0195
COMMANDPARAMSPARSER	Near	CODE:08E4
COUNTER	Word	CODE:01B4
CURSIVEENABLED	Byte	CODE:0184
CURSIVESYMBOL	Byte	CODE:0185
ERRORPARAMMSG	Byte	CODE:03D3
ERRORPARAMMSG_LENGTH	Number	0025
F1_TXT	Byte	CODE:04EB
F2_TXT	Byte	CODE:04ED
F3_TXT	Byte	CODE:04EF
F4_TXT	Byte	CODE:04F1
FX_LENGTH	Number	0002
HELPMSG	Byte	CODE:0257
HELPMSG_LENGTH	Number	017C
IGNOREDCHARS	Byte	CODE:013D
IGNOREDLENGTH	Number	003A
IGNOREENABLED	Byte	CODE:0177
INSTALLEDMSG	Byte	CODE:0462
LINE1_LENGTH	Number	0035
LINE2_LENGTH	Number	0035
LINE3_LENGTH	Number	0035
NEW_INT1CH	Far	CODE:064E
NEW_INT2FH	Near	CODE:067D
NEW_INT9H	Far	CODE:0595

NOMEMMSG	Byte	CODE:048B
NOREMOVEMSG	Byte	CODE:04CE
NOREMOVEMSG_LENGTH	Number	001D
NOTINSTALLEDMSG	Byte	CODE:049F
NOTLOADTSR	Byte	CODE:01B3
OLD_INT1CHOFFSET	Word	CODE:01AA
OLD_INT1CHSEGMENT	Word	CODE:01AC
OLD_INT2FHOFFSET	Word	CODE:01AE
OLD_INT2FHSEGMENT	Word	CODE:01B0
OLD_INT9HOFFSET	Word	CODE:01A6
OLD_INT9HSEGMENT	Word	CODE:01A8
PRINTDELAY	Number	0007
PRINTPOS	Word	CODE:01B6
PRINTSIGNATURE	Near	CODE:070A
REMOVEDMSG	Byte	CODE:04BD
REMOVEDMSG_LENGTH	Number	0011
REPLACEWITH	Byte	CODE:0103
SAVEDSYMBOL	Byte	CODE:0196
SAVEFONT	Near	CODE:0811

Symbol Table

SETCURSIVE	Near	CODE:07B7
SIGNATURELINE1	Byte	CODE:01B8
SIGNATURELINE2	Byte	CODE:01ED
SIGNATURELINE3	Byte	CODE:0222
SIGNATUREPRINTINGENABLED	Byte	CODE:0183
TABLEBOTTOM	Byte	CODE:042D
TABLEBOTTOM_LENGTH	Number	0035
TABLETOP	Byte	CODE:03F8
TABLETOP_LENGTH	Number	0035
TRANSLATEENABLED	Byte	CODE:0182
TRANSLATEFROM	Byte	CODE:0178
TRANSLATELENGTH	Number	0005
TRANSLATETO	Byte	CODE:017D
TRUE	Number	00FF
UNLOADTSR	Byte	CODE:01B2
_2FH_EXIT	Near	CODE:0705
_2FH_STD	Near	CODE:068D
_ACTUALPRINT	Near	CODE:0748
_ALREADYINSTALLED	Near	CODE:08D1
_ALREADY_INSTALLED	Near	CODE:0692
_BLOCK	Near	CODE:0618
_CHECKF6	Near	CODE:0502
_CHECKF7	Near	CODE:0526
_CHECKF8	Near	CODE:0547
_CHECKF9	Near	CODE:0568
_CHECK_IGNORED	Near	CODE:060C
_CHECK_TRANSLATE	Near	CODE:0624
_CHECK_TRANSLATE_LOOP	Near	CODE:0631
_DONTPRINT	Near	CODE:0676
_ERRORPARAM	Near	CODE:0927
_EXIT	Near	CODE:08E2
_EXITHELP	Near	CODE:0938
_EXITSETCURSIVE	Near	CODE:0801
_EXIT_TMP	Near	CODE:0869
_F6	Near	CODE:05A7
_F7	Near	CODE:05B5
_F8	Near	CODE:05C6
_F9	Near	CODE:05D4

_FINISHTSR	**Near**	**CODE:091E**
_GO	**Near**	**CODE:05FC**
_GREENF6	**Near**	**CODE:051F**
_GREENF7	**Near**	**CODE:0543**
_GREENF8	**Near**	**CODE:0564**
_GREENF9	**Near**	**CODE:0585**
_INITTSR	**Near**	**CODE:081E**
_LETSPRINT	**Near**	**CODE:0669**
_NEXTCHAR	**Near**	**CODE:08F8**
_NOTMEM	**Near**	**CODE:08DB**
_NOTREMOVE	**Near**	**CODE:06E0**
_NOTREMOVINGNOW	**Near**	**CODE:0854**
_NOTTOPRINT	**Near**	**CODE:067B**
_OUTFX	**Near**	**CODE:0589**
_PRINTBOTTOM	**Near**	**CODE:0741**
_PRINTCENTER	**Near**	**CODE:073A**
_PRINTTOP	**Near**	**CODE:0733**
_QUESTION	**Near**	**CODE:0907**
_QUIT	**Near**	**CODE:0646**

Symbol Table

_REDF6	Near	CODE:0518
_REDF7	Near	CODE:053C
_REDF8	Near	CODE:055D
_REDF9	Near	CODE:057E
_REMOVE	Near	CODE:08C8
_REMOVINGONPARAMETER	Near	CODE:0840
_RESTORESYMBOL	Near	CODE:07F2
_SHIFTTABLE	Near	CODE:07C9
_START	Near	CODE:0100
_TEST_FX	Near	CODE:05A5
_TMP	Near	CODE:086C
_TRANSLATE	Near	CODE:063D
_TRANSLATE_OR_IGNORE	Near	CODE:05E2
_UNINSTALL	Near	CODE:0695
_UNLOADED	Near	CODE:06F4

Groups & Segments	Bit	Size	Align	Combine	Class
CODE	16	0939	Para	none	CODE

2. Файл unloader.lst

```
Turbo Assembler    Version 3.1       05/08/24 04:50:48    Page 1
UNLOADER.asm

                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                     ; unloader.asm
                     ;
                     ; Сборка:
                     ;       tasm.exe /l unloader.asm
                     ;       tlink /t /x unloader.obj
                     ;
                     ; Программа для выгрузки TSR из памяти
                     ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

0000                 code segment 'code'
                         assume    CS:code, DS:code
                         org 100h
0100                 _start:

0100  B4 09              mov AH, 09h             ; Функция DOS    для вывода строки
0102  BA 012Ar           lea DX, prompt_msg   ; Загрузить адрес сообщения приглашения
0105  CD 21              int 21h

0107  B4 01              mov AH, 01h             ; Функция DOS    для чтения символа без отображения
0109  CD 21              int 21h             ; Вызов прерывания DOS для чтения символа
010B  A2 0174r           mov response, AL     ; Сохранить ответ пользователя

                         ; Проверка ответа пользователя
010E  80 3E 0174r 59   cmp response, 'Y'     ; Сравнить с символом      'Y'
0113  74 09              je unload_tsr           ; Если 'Y', выгрузить        TSR
0115  80 3E 0174r 4E   cmp response, 'N'     ; Сравнить с символом      'N'
011A  74 0A              je skip_unload          ; Если 'N', пропустить выгрузку

011C  EB E2              jmp _start          ; В случае ввода другого символа, повторить запрос

```

```
32  011E                        unload_tsr:
33  011E  B4 FF                     mov AH, 0FFh
34  0120  B0 01                     mov AL, 1
35  0122  CD 2F                     int 2Fh             ; наше прерывание
36  0124  CD 20                     int 20h             ; выход
37
38  0126                        skip_unload:
39                                  ; Выход
40  0126  B4 4C                     mov AH, 4Ch                 ; Функция DOS   для завершения программы
41  0128  CD 21                     int 21h             ; Вызов прерывания DOS для завершения
42
43  012A  82 EB A3 E0 E3 A7     A8+ prompt_msg  db 'Выгрузить TSR из памяти? (Y         или N) для выполнения или отказа в
выгрузке: $'
44        E2 EC 20 54 53 52 20+
45        A8 A7 20 AF A0 AC     EF+
46        E2 A8 3F 20 28 59 20+
47        A8 AB A8 20 4E 29     20+
48        A4 AB EF 20 A2 EB     AF+
49        AE AB AD A5 AD A8     EF+
50        20 A8 AB A8 20 AE     E2+
51        AA A0 A7 A0 20 A2     20+
52        A2 EB A3 E0 E3 A7     AA+
53        A5 3A 20 24
54  0174  ??                   response db ?
55  0175                        code ends
56                              end _start
```

Turbo Assembler Version 3.1 05/08/24 04:50:48 Page 2

Symbol Table

Symbol Name	Type	Value
??DATE	Text	"05/08/24"
??FILENAME	Text	"UNLOADER"
??TIME	Text	"04:50:48"
??VERSION	Number	030A
@CPU	Text	0101H
@CURSEG	Text	CODE
@FILENAME	Text	UNLOADER
@WORDSIZE	Text	2
PROMPT_MSG	Byte	CODE:012A
RESPONSE	Byte	CODE:0174
SKIP_UNLOAD	Near	CODE:0126
UNLOAD_TSR	Near	CODE:011E
_START	Near	CODE:0100

Groups & Segments	Bit Size	Align	Combine	Class
CODE	16 0175	Para	none	CODE