Московский государственный технический университет
им. Н.Э. Баумана

УТВЕРЖДАЮ:

Большаков С.А.
"__"_____2023 г.

Курсовая работа по курсу «Системное программирование»

Исходный текст программного продукта
(вид документа)

писчая бумага
(вид носителя)

25
(количество листов)

ИСПОЛНИТЕЛИ:

студенты группы ИУ5-42б _____
Казицин А.Ю.
"__"_____2023 г.

Москва – 2023

Содержание

1. Файл tsr.lst

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 1
tsr.asm



      1                             ;=======================================
      2                             ; tsr.asm
      3                             ;
      4                             ; Сборка:
      5                             ; > tasm.exe /l tsr.asm
      6                             ; > tlink /t /x tsr.obj
      7                             ;=======================================
      8                             ; Казицин А.Ю., ИУ5-42Б, Вариант 8
      9                             ;=======================================
     10
     11 0000                        code segment    'code'
     12                                     assume  CS:code, DS:code
     13                                     org     100h
     14 0100                                _start:
     15
     16 0100  E9 072C                       jmp _initTSR ; на начало программы
     17
     18                                     ; данные
     19 0103  E9 F6 F3 EA E5 ED  E3+        ignoredChars                            DB
'йцукенгшщзхъфывапролджёячсмитьбю'       ;   +
     20       F8 F9 E7 F5 FA F4  FB+  список игнорируемых символов
     21       E2 E0 EF F0 EE EB  E4+
     22       E6 B8 FF F7 F1 EC  E8+
     23       F2 FC E1 FE
     24                                     ;replaceWith                    DB
'qwertyuiop[]asdfghjkl;\zxcvbnm,.QWERTYUIOP          +
     25                             []ASDFGHJKL;\ZXCVBNM,.'
     26       =0020                         ignoredLength                           equ     $-
ignoredChars ; длина строки          +
     27                             ignoredChars
     28 0123  00                    ignoreEnabled                   DB      0       ; флаг
функции игнорирования   +
     29                             ввода
     30 0124  7B 57 58 49 4F                translateFrom                           DB      '{WXIO' ;
символы для замены (ХЦЧШЩ на +
     31                             англ. раскладке)
     32 0129  D5 D6 D7 D8 D9                translateTo                     DB      'ХЦЧШЩ' ; символы на
которые будет   идти+
     33                             замена
     34       =0005                         translateLength                         equ     $-translateTo
; длина строки      +
     35                             trasnlateFrom
     36 012E  00                    translateEnabled                DB      0       ; флаг
функции перевода
     37
     38 012F  00                    signaturePrintingEnabled        DB      0       ; флаг
функции вывода      +
     39                             информации об авторе
     40 0130  00                    cursiveEnabled                  DB      0       ; флаг
перевода символа в курсив
     41
     42 0131  00                    cursiveSymbol                   DB 00000000b    ; символ,
составленный из     +
     43                             единичек (его курсивный вариант)
     44 0132  00                                                                    DB 00000000b
     45 0133  00                                                                    DB 00000000b
     46 0134  0F                                                                    DB 00001111b
     47 0135  11                                                                    DB 00010001b
     48 0136  11                                                                    DB 00010001b
     49 0137  12                                                                    DB 00010010b
     50 0138  22                                                                    DB 00100010b
     51 0139  22                                                                    DB 00100010b
     52 013A  22                                                                    DB 00100010b
     53 013B  44                                                                    DB 01000100b
     54 013C  44                                                                    DB 01000100b
     55 013D  87                                                                    DB 10000111b
     56 013E  00                                                                    DB 00000000b
     57 013F  00                                                                    DB 00000000b
```

```
     58 0140  00                                                                    DB 00000000b
     59
     60 0141  CB                    charToCursiveIndex              DB 'Л' ; символ для замены
     61 0142  10*(FF)                       savedSymbol                             DB 16 dup(0FFh) ;
переменная для      +
     62                             хранения старого символа
     63
     64       =00FF                         true
              equ     0FFh ; +
     65                             константа истинности
     66 0152  ????                          old_int9hOffset                         DW      ?
; адрес старого     +
     67                             обработчика int 9h
     68 0154  ????                          old_int9hSegment                        DW      ?
; сегмент старого            +
     69                             обработчика int 9h
     70 0156  ????                          old_int1ChOffset                        DW      ?
; адрес старого     +
     71                             обработчика int 1Ch
     72 0158  ????                          old_int1ChSegment                 DW      ?       ;
сегмент старого обработчика  +
     73                             int 1Ch
     74 015A  ????                          old_int2FhOffset                        DW      ?
; адрес старого     +
     75                             обработчика int 2Fh
     76 015C  ????                          old_int2FhSegment                 DW      ?       ;
сегмент старого обработчика  +
     77                             int 2Fh
     78
     79 015E  00                    unloadTSR                                       DB      0 ;
1 - выгрузить       +
     80                             резидент
     81 015F  00                    notLoadTSR                              DB      0       ; 1
- не загружать
     82 0160  0000                          counter                                 DW      0
     83       =0005                         printDelay                              equ     5 ;
таймер  вывода сообщения об +
     84                             исполнителе (секунды)
     85 0162  0000                          printPos                                        DW
0            ; положение    +
     86                             подписи на  экране. 0 - верх, 1 - центр, 2 - низ
     87
     88                                    ;@ заменить на собственные  данные. формирование таблицы идет
по строке большей длины   +
     89                             (1я строка).
     90 0164  B3 CA E0 E7 E8 F6  E8+        signatureLine1                  DB      179, 'Казицин
А.Ю.                          +
     91       ED 20 C0 2E DE 2E  20+ ', 179
     92       20 20 20 20 20 20  20+
     93       20 20 20 20 20 20  20+
     94       20 20 20 20 20 20  20+
     95       20 20 20 20 20 20  20+
     96       20 20 20 20 20 20  20+
     97       20 B3
     98       =0033                         Line1_length                            equ     $-
signatureLine1
     99 0197  B3 C8 D3 35 2D 34  32+        signatureLine2                  DB      179, 'ИУ5-42Б
                               +
    100       C1 20 20 20 20 20  20+ ', 179
    101       20 20 20 20 20 20  20+
    102       20 20 20 20 20 20  20+
    103       20 20 20 20 20 20  20+
    104       20 20 20 20 20 20  20+
    105       20 20 20 20 20 20  20+
    106       20 20 B3
    107       =0034                         Line2_length                            equ     $-
signatureLine2
    108 01CB  B3 C2 E0 F0 E8 E0  ED+        signatureLine3                  DB      179, 'Вариант
№8                            +
    109       F2 20 B9 38 20 20  20+ ', 179
    110       20 20 20 20 20 20  20+
    111       20 20 20 20 20 20  20+
```

```
112       20 20 20 20 20 20  20+
113       20 20 20 20 20 20  20+
114       20 20 20 20 20 20  20+
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 3
tsr.asm



    115       20 B3
    116       =0033                         Line3_length                                    equ     $-
signatureLine3
    117 01FE  3E 74 73 72 2E 63  6F+        helpMsg DB  '>tsr.com [/?]', 10, 13
    118       6D 20 5B 2F 3F 5D  0A+
    119       0D
    120 020D  20 5B 2F 3F 5D 20  2D+                                        DB ' [/?] - вывод данной
справки',       10, 13
    121       20 E2 FB E2 EE E4  20+
    122       E4 E0 ED ED EE E9  20+
    123       F1 EF F0 E0 E2 EA  E8+
    124       0A 0D
    125 022B  20 C2 FB E3 F0 F3  E7+                                        DB ' Выгрузку осуществляет
        повторный запуск', 10, 13
    126       EA F3 20 EE F1 F3  F9+
    127       E5 F1 F2 E2 EB FF  E5+
    128       F2 20 EF EE E2 F2  EE+
    129       F0 ED FB E9 20 E7  E0+
    130       EF F3 F1 EA 0A 0D
    131 0254  20 3D 3D 3D 3D 3D  3D+                                        DB '
                            +
    132       3D 3D 3D 3D 3D 3D  3D+
==========================================================================', 10, 13
    133       3D 3D 3D 3D 3D 3D  3D+
    134       3D 3D 3D 3D 3D 3D  3D+
    135       3D 3D 3D 3D 3D 3D  3D+
    136       3D 3D 3D 3D 3D 3D  3D+
    137       3D 3D 3D 3D 3D 3D  3D+
    138       3D 3D 3D 3D 3D 3D  3D+
    139       3D 3D 3D 3D 3D 3D  3D+
    140       3D 3D 3D 3D 3D 3D  3D+
    141       3D 3D 3D 3D 3D 3D  0A+
    142       0D
    143 02A2  20 20 46 32 20 20  2D+                                        DB '  F2  - вывод ФИО и
группы в верх экрана по таймеру+
    144       20 E2 FB E2 EE E4  20+  (5 секунд)', 10, 13
    145       D4 C8 CE 20 E8 20  E3+
    146       F0 F3 EF EF FB 20  E2+
    147       20 E2 E5 F0 F5 20  FD+
    148       EA F0 E0 ED E0 20  EF+
    149       EE 20 F2 E0 E9 EC  E5+
    150       F0 F3 20 28 35 20  F1+
    151       E5 EA F3 ED E4 29  0A+
    152       0D
    153 02E2  20 20 46 33 20 20  2D+                                        DB '  F3  - вкл/откл
курсивного вывода русского символа+
    154       20 E2 EA EB 2F EE  F2+  Л', 10, 13
    155       EA EB 20 EA F3 F0  F1+
    156       E8 E2 ED EE E3 EE  20+
    157       E2 FB E2 EE E4 E0  20+
    158       F0 F3 F1 F1 EA EE  E3+
    159       EE 20 F1 E8 EC E2  EE+
    160       EB E0 20 CB 0A 0D
    161 0319  20 20 46 34 20 20  2D+                                        DB '  F4  - вкл/откл
частичной русификации    клавиатуры  +
    162       20 E2 EA EB 2F EE  F2+  ({WXIO -> ХЦЧШЩ)',  10, 13
    163       EA EB 20 F7 E0 F1  F2+
    164       E8 F7 ED EE E9 20  F0+
    165       F3 F1 E8 F4 E8 EA  E0+
    166       F6 E8 E8 20 EA EB  E0+
    167       E2 E8 E0 F2 F3 F0  FB+
    168       20 28 7B 57 58 49  4F+
    169       20 2D 3E 20 D5 D6  D7+
    170       D8 D9 29 0A 0D
    171 035D  20 20 46 35 20 20  2D+                                        DB '  F5  - вкл/откл режима
ограничения ввода строчных +
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 4
tsr.asm


    172       20 E2 EA EB 2F EE  F2+  русских букв', 10,  13
    173       EA EB 20 F0 E5 E6  E8+
    174       EC E0 20 EE E3 F0  E0+
    175       ED E8 F7 E5 ED E8  FF+
    176       20 E2 E2 EE E4 E0  20+
    177       F1 F2 F0 EE F7 ED  FB+
    178       F5 20 F0 F3 F1 F1  EA+
    179       E8 F5 20 E1 F3 EA  E2+
    180       0A 0D
    181 039E  20 3D 3D 3D 3D 3D  3D+                                        DB '
                           +
    182       3D 3D 3D 3D 3D 3D  3D+
================================================================================', 10, 13
    183       3D 3D 3D 3D 3D 3D  3D+
    184       3D 3D 3D 3D 3D 3D  3D+
    185       3D 3D 3D 3D 3D 3D  3D+
    186       3D 3D 3D 3D 3D 3D  3D+
    187       3D 3D 3D 3D 3D 3D  3D+
    188       3D 3D 3D 3D 3D 3D  3D+
    189       3D 3D 3D 3D 3D 3D  3D+
    190       3D 3D 3D 3D 3D 3D  3D+
    191       3D 3D 3D 3D 3D 3D  0A+
    192       0D
    193
    194       =01EE                           helpMsg_length              equ  $-helpMsg
    195 03EC  CE F8 E8 E1 EA E0  20+          errorParamMsg                   DB      'Ошибка параметров
        коммандной строки', 10,  13
    196       EF E0 F0 E0 EC E5  F2+
    197       F0 EE E2 20 EA EE  EC+
    198       EC E0 ED E4 ED EE  E9+
    199       20 F1 F2 F0 EE EA  E8+
    200       0A 0D
    201       =0025                           errorParamMsg_length    equ     $-errorParamMsg
    202
    203 0411  DA 31*(C4) BF                   tableTop                                DB      218,
Line1_length-2 dup (196), 191
    204       =0033                           tableTop_length                 equ     $-tableTop
    205 0444  C0 31*(C4) D9                   tableBottom                     DB      192, Line1_length-2
dup (196), 217
    206       =0033                           tableBottom_length          equ  $-tableBottom
    207
    208                                       ; сообщения
    209 0477  D0 E5 E7 E8 E4 E5  ED+          installedMsg                    DB  'Резидент загружен!$'
    210       F2 20 E7 E0 E3 F0  F3+
    211       E6 E5 ED 21 24
    212 048A  D0 E5 E7 E8 E4 E5  ED+          alreadyInstalledMsg     DB  'Резидент уже загружен$'
    213       F2 20 F3 E6 E5 20  E7+
    214       E0 E3 F0 F3 E6 E5  ED+
    215       24
    216 04A0  CD E5 E4 EE F1 F2  E0+          noMemMsg                                DB  'Недостаточно
памяти$'
    217       F2 EE F7 ED EE 20  EF+
    218       E0 EC FF F2 E8 24
    219 04B4  CD E5 20 F3 E4 E0  EB+          notInstalledMsg             DB  'Не удалось загрузить
резидент$'
    220       EE F1 FC 20 E7 E0  E3+
    221       F0 F3 E7 E8 F2 FC  20+
    222       F0 E5 E7 E8 E4 E5  ED+
    223       F2 24
    224
    225 04D2  D0 E5 E7 E8 E4 E5  ED+          removedMsg                          DB  'Резидент выгружен!'
    226       F2 20 E2 FB E3 F0  F3+
    227       E6 E5 ED 21
    228       =0012                           removedMsg_length               equ     $-removedMsg
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 5
tsr.asm



    229
    230 04E4  CD E5 20 F3 E4 E0  EB+          noRemoveMsg             DB  'Не удалось выгрузить резидент'
    231       EE F1 FC 20 E2 FB  E3+
    232       F0 F3 E7 E8 F2 FC  20+
    233       F0 E5 E7 E8 E4 E5  ED+
    234       F2
    235       =001D                           noRemoveMsg_length      equ     $-noRemoveMsg
    236
    237 0501  46 32                           f1_txt                                          DB      'F2'
    238 0503  46 33                           f2_txt                                          DB      'F3'
    239 0505  46 34                           f3_txt                                          DB      'F4'
    240 0507  46 35                           f4_txt                                          DB      'F5'
    241       =0002                           fx_length                                       equ     $-
f4_txt
    242
    243 0509                                  changeFx proc
    244 0509  50                              push AX
    245 050A  53                              push BX
    246 050B  51                              push CX
    247 050C  52                              push DX
    248 050D  55                              push BP
    249 050E  06                              push ES
    250 050F  33 DB                                   xor BX, BX
    251
    252 0511  B4 03                                   mov AH, 03h
    253 0513  CD 10                                   int 10h
    254 0515  52                              push DX
    255
    256 0516  0E                              push CS
    257 0517  07                              pop ES
    258
    259 0518                                  _checkF1:
    260 0518  BD 0501r                                lea BP, f1_txt
    261 051B  B9 0002                                 mov CX, fx_length
    262 051E  B7 00                                   mov BH, 0
    263 0520  B6 00                                   mov DH, 0
    264 0522  B2 4E                                   mov DL, 78
    265 0524  B8 1301                                 mov AX, 1301h
    266
    267 0527  80 3E 012Fr FF                          cmp signaturePrintingEnabled, true
    268 052C  74 07                                   je _greenF1
    269
    270 052E                                          _redF1:
    271 052E  B3 4F                                           mov BL, 01001111b ; бирюзовый
    272 0530  CD 10                                           int 10h
    273 0532  EB 08 90                                        jmp _checkF2
    274
    275 0535                                          _greenF1:
    276 0535  BD 0501r                                        lea BP, f1_txt
    277 0538  B3 2F                                           mov BL, 00101111b ; желтый
    278 053A  CD 10                                           int 10h
    279
    280 053C                                  _checkF2:
    281 053C  BD 0503r                                lea BP, f2_txt
    282 053F  B9 0002                                 mov CX, fx_length
    283 0542  B7 00                                   mov BH, 0
    284 0544  B6 01                                   mov DH, 1
    285 0546  B2 4E                                   mov DL, 78
```

```
286 0548  B8 1301                             mov AX, 1301h
287
288 054B  80 3E 0130r FF                      cmp cursiveEnabled, true
289 0550  74 07                               je _greenF2
290
291 0552                                      _redF2:
292 0552  B3 4F                                       mov BL, 01001111b ; бирюзовый
293 0554  CD 10                                       int 10h
294 0556  EB 05 90                                    jmp _checkF3
295
296 0559                                      _greenF2:
297 0559  B3 2F                                       mov BL, 00101111b ; желтый
298 055B  CD 10                                       int 10h
299
300 055D                                  _checkF3:
301 055D  BD 0505r                            lea BP, f3_txt
302 0560  B9 0002                             mov CX, fx_length
303 0563  B7 00                               mov BH, 0
304 0565  B6 02                               mov DH, 2
305 0567  B2 4E                               mov DL, 78
306 0569  B8 1301                             mov AX, 1301h
307
308 056C  80 3E 012Er FF                      cmp translateEnabled, true
309 0571  74 07                               je _greenF3
310
311 0573                                      _redF3:
312 0573  B3 4F                                       mov BL, 01001111b ; бирюзовый
313 0575  CD 10                                       int 10h
314 0577  EB 05 90                                    jmp _checkF4
315
316 057A                                      _greenF3:
317 057A  B3 2F                                       mov BL, 00101111b ; желтый
318 057C  CD 10                                       int 10h
319
320 057E                                  _checkF4:
321 057E  BD 0507r                            lea BP, f4_txt
322 0581  B9 0002                             mov CX, fx_length
323 0584  B7 00                               mov BH, 0
324 0586  B6 03                               mov DH, 3
325 0588  B2 4E                               mov DL, 78
326 058A  B8 1301                             mov AX, 1301h
327
328 058D  80 3E 0123r FF                      cmp ignoreEnabled,  true
329 0592  74 07                               je _greenF4
330
331 0594                                      _redF4:
332 0594  B3 4F                                       mov BL, 01001111b ; бирюзовый
333 0596  CD 10                                       int 10h
334 0598  EB 05 90                                    jmp _outFx
335
336 059B                                      _greenF4:
337 059B  B3 2F                                       mov BL, 00101111b ; желтый
338 059D  CD 10                                       int 10h
339
340 059F                                  _outFx:
341 059F  5A                              pop DX
342 05A0  B4 02                               mov AH, 02h
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 7
tsr.asm



    343 05A2  CD 10                                 int 10h
    344
    345 05A4  07                            pop ES
    346 05A5  5D                            pop BP
    347 05A6  5A                            pop DX
    348 05A7  59                            pop CX
    349 05A8  5B                            pop BX
    350 05A9  58                            pop AX
    351 05AA  C3                            ret
    352 05AB                                changeFx endp
    353                                 ;новый обработчик
    354 05AB                                new_int9h proc  far
    355                                             ; сохраняем значения всех,  изменяемых регистров в
стэке
    356 05AB  56                            push SI
    357 05AC  50                            push AX
    358 05AD  53                            push BX
    359 05AE  51                            push CX
    360 05AF  52                            push DX
    361 05B0  06                            push ES
    362 05B1  1E                            push DS
    363                                             ; синхронизируем CS и DS
    364 05B2  0E                            push CS
    365 05B3  1F                            pop     DS
    366
    367 05B4  B8 0040                               mov     AX, 40h ; 40h-сегмент,где хранятся  флаги
сост-я клавиатуры, кольц.           +
    368                              буфер ввода
    369 05B7  8E C0                                 mov     ES, AX
    370 05B9  E4 60                                 in      AL, 60h ; записываем в AL скан-код  нажатой
        клавиши
    371
    372                                             ;@ проверка на Ctrl+U, только для ИУ5-41
    373
    374                                             ;@ далее - код для всех вариантов
    375
    376                                             ;проверка F1-F4
    377 05BB                                        _test_Fx:
    378 05BB  2C 3A                                 sub AL, 58  ; в AL теперь номер функциональной клавиши
    379 05BD                                        _F1:
    380 05BD  3C 02                                         cmp AL, 2 ; F2
    381 05BF  75 0A                                         jne _F2
    382 05C1  F6 16 012Fr                                   not signaturePrintingEnabled
    383 05C5  E8 FF41                                       call changeFx
    384 05C8  EB 2E 90                                      jmp _translate_or_ignore
    385 05CB                                        _F2:
    386 05CB  3C 03                                         cmp AL, 3 ; F3
    387 05CD  75 0D                                         jne _F3
    388 05CF  F6 16 0130r                                   not cursiveEnabled
    389 05D3  E8 FF33                                       call changeFx
    390 05D6  E8 01EF                                       call setCursive ; перевод символа в курсив  и
обратно в зависимости    от  +
    391                              флага cursiveEnabled
    392 05D9  EB 1D 90                                      jmp _translate_or_ignore
    393 05DC                                        _F3:
    394 05DC  3C 04                                         cmp AL, 4 ; F4
    395 05DE  75 0A                                         jne _F4
    396 05E0  F6 16 012Er                                   not translateEnabled
    397 05E4  E8 FF22                                       call changeFx
    398 05E7  EB 0F 90                                      jmp _translate_or_ignore
    399 05EA                                        _F4:
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 8
tsr.asm



    400 05EA  3C 05                                         cmp AL, 5 ; F5
    401 05EC  75 0A                                         jne _translate_or_ignore
    402 05EE  F6 16 0123r                                   not ignoreEnabled
    403 05F2  E8 FF14                                       call changeFx
    404 05F5  EB 01 90                                      jmp _translate_or_ignore
    405
    406                                                 ;игнорирование и перевод
    407 05F8                                            _translate_or_ignore:
    408
    409 05F8  9C                                pushf
    410 05F9  2E: FF 1E  0152r                              call dword  ptr CS:[old_int9hOffset] ; вызываем
стандартный     обработчик прерывания
    411 05FE  B8 0040                                       mov     AX, 40h         ; 40h-сегмент,где хранятся
        флаги сост-я клавы,кольц.   +
    412                                 буфер ввода
    413 0601  8E C0                                         mov     ES, AX
    414 0603  26: 8B 1E  001C                               mov     BX, ES:[1Ch]    ; адрес хвоста
    415 0608  4B                                dec     BX      ; сместимся назад к последнему
    416 0609  4B                                dec     BX      ; введённому символу
    417 060A  83 FB 1E                                      cmp     BX, 1Eh ; не вышли  ли мы за пределы буфера?
    418 060D  73 03                                         jae     _go
    419 060F  BB 003C                                       mov     BX, 3Ch ; хвост вышел за пределы буфера,
значит последний введённый    +
    420                                 символ
    421                                                                 ; находится в конце буфера
    422
    423 0612                                    _go:
    424 0612  26: 8B 17                                     mov DX, ES:[BX] ; в DX 0 введённый  символ
    425                                                     ;включен ли режим блокировки ввода?
    426 0615  80 3E 0123r FF                                cmp ignoreEnabled,  true
    427 061A  75 1A                                         jne _check_translate
    428
    429                                                     ; да, включен
    430 061C  BE 0000                                       mov SI, 0
    431 061F  B9 0020                                       mov CX, ignoredLength ;кол-во игнорируемых  символов
    432
    433                                                     ; проверяем, присутствует ли текущий символ в списке
игнорируемых
    434 0622                                    _check_ignored:
    435 0622  3A 94 0103r                                   cmp DL,ignoredChars[SI]
    436 0626  74 06                                         je _block
    437 0628  46                                inc SI
    438 0629  E2 F7                             loop _check_ignored
    439 062B  EB 09 90                                      jmp _check_translate
    440
    441                                         ; блокируем
    442 062E                                    _block:
    443 062E  26: 89 1E  001C                               mov ES:[1Ch], BX ;блокировка ввода  символа
    444                                                     ;@ если по варианту нужно не блокировать ввод
символа,
    445                                                     ;@ а заменять одни  символы другими,
    446                                                     ;@ замените строку  выше строкой
    447                                                     ;mov ES:[BX], AX
    448                                                     ;@ на месте AX может быть '*' для замены всех
символов множества ignoredChars  +
    449                                 на звёздочки
    450                                                     ;@ или, для перевода одних  символов в другие -
завести массив
    451                                                     ;@ replaceWith DB '...', где перечислить символы, на
которые пойдёт замена
    452                                                     ;@ и раскомментировать строки ниже:
    453                                                     ;xor AX, AX
    454                                                     ; mov AL, replaceWith[SI]
    455                                                     ; mov ES:[BX], AX       ; замена символа
    456 0633  EB 23 90                                      jmp _quit
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 9
tsr.asm



    457
    458 0636                                    _check_translate:
    459                                                 ; включен ли режим  перевода?
    460 0636  80 3E 012Er FF                            cmp translateEnabled, true
    461 063B  75 1B                                     jne _quit
    462
    463                                                 ; да, включен
    464 063D  BE 0000                                   mov SI, 0
    465 0640  B9 0005                                   mov CX, translateLength ; кол-во символов для перевода
    466                                                 ; проверяем, присутствует ли текущий символ в списке
для перевода
    467 0643                                            _check_translate_loop:
    468 0643  3A 94 0124r                                       cmp DL, translateFrom[SI]
    469 0647  74 06                                             je _translate
    470 0649  46                                        inc SI
    471 064A  E2 F7                                     loop _check_translate_loop
    472 064C  EB 0A 90                                  jmp _quit
    473
    474                                                 ; переводим
    475 064F                                            _translate:
    476 064F  33 C0                                             xor AX, AX
    477 0651  8A 84 0129r                                       mov AL, translateTo[SI]
    478 0655  26: 89 07                                         mov ES:[BX], AX ; замена символа
    479
    480 0658                                    _quit:
    481                                                 ; восстанавливаем все регистры
    482 0658  1F                                pop     DS
    483 0659  07                                pop     ES
    484 065A  5A                                pop DX
    485 065B  59                                pop CX
    486 065C  5B                                pop     BX
    487 065D  58                                pop     AX
    488 065E  5E                                pop SI
    489 065F  CF                                iret
    490 0660                                new_int9h endp
    491
    492                                     ;=== Обработчик прерывания  int 1Ch ===;
    493                                     ;=== Вызывается каждые 55 мс ===;
    494 0660                                new_int1Ch  proc far
    495 0660  50                            push AX
    496 0661  0E                            push CS
    497 0662  1F                            pop DS
    498
    499 0663  9C                            pushf
    500 0664  2E: FF 1E  0156r                      call dword  ptr CS:[old_int1ChOffset]
    501
    502 0669  80 3E 012Fr FF                        cmp signaturePrintingEnabled, true  ; если нажата управляющая
клавиша (в данном случае  +
    503                                     F1)
    504 066E  75 1C                                 jne _notToPrint
    505
    506 0670  83 3E 0160r 5B                        cmp counter, printDelay*1000/55 + 1 ; если  кол-во
"тактов"эквивалентно        +
    507                                     %printDelay% секундам
    508 0675  74 03                                 je _letsPrint
    509
    510 0677  EB 0E 90                              jmp _dontPrint
    511
    512 067A                                        _letsPrint:
    513 067A  F6 16 012Fr                                   not signaturePrintingEnabled
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 10
tsr.asm



    514 067E  C7 06 0160r 0000                              mov counter, 0
    515 0684  E8 0094                                       call printSignature
    516
    517 0687                                        _dontPrint:
    518 0687  83 06 0160r 01                                add counter, 1
    519
    520 068C                                        _notToPrint:
    521
    522 068C  58                        pop AX
    523
    524 068D  CF                        iret
    525 068E                            new_int1Ch  endp
    526
    527                                 ;=== Обработчик прерывания  int 2Fh ===;
    528                                 ;=== Служит для:
    529                                 ;===  1) проверки факта присутствия TSR в памяти (при AH=0FFh, AL=0)
    530                                 ;===     будет возвращён AH='i' в случае, если TSR  уже загружен
    531                                 ;===  2) выгрузки TSR из памяти (при AH=0FFh, AL=1)
    532                                 ;===
    533 068E                            new_int2Fh  proc
    534 068E  80 FC FF                              cmp     AH, 0FFh        ;наша функция?
    535 0691  75 0B                                 jne     _2Fh_std        ;нет - на старый обработчик
    536 0693  3C 00                                 cmp     AL, 0   ;подфункция проверки, загружен ли резидент  в
память?
    537 0695  74 0C                                 je      _already_installed
    538 0697  3C 01                                 cmp     AL, 1   ;подфункция выгрузки из памяти?
    539 0699  74 0B                                 je      _uninstall
    540 069B  EB 01 90                              jmp     _2Fh_std        ;нет - на старый обработчик
    541
    542 069E                            _2Fh_std:
    543 069E  2E: FF 2E  015Ar                      jmp     dword ptr CS:[old_int2FhOffset] ;вызов старого
обработчика
    544
    545 06A3                            _already_installed:
    546 06A3  B4 69                                 mov     AH, 'i' ;вернём 'i', если резидент  загружен
в память
    547 06A5  CF                                    iret
    548
    549 06A6                            _uninstall:
    550 06A6  1E                        push    DS
    551 06A7  06                        push    ES
    552 06A8  52                        push    DX
    553 06A9  53                        push    BX
    554
    555 06AA  33 DB                                 xor BX, BX
    556
    557                                             ; CS = ES,  для доступа к переменным
    558 06AC  0E                        push CS
    559 06AD  07                        pop ES
    560
    561 06AE  B8 2509                               mov     AX, 2509h
    562 06B1  26: 8B 16  0152r                      mov DX, ES:old_int9hOffset          ; возвращаем вектор
прерывания
    563 06B6  26: 8E 1E  0154r                  mov DS, ES:old_int9hSegment         ; на место
    564 06BB  CD 21                                 int     21h
    565
    566 06BD  B8 251C                               mov     AX, 251Ch
    567 06C0  26: 8B 16  0156r                      mov DX, ES:old_int1ChOffset  ; возвращаем вектор прерывания
    568 06C5  26: 8E 1E  0158r                  mov DS, ES:old_int1ChSegment         ; на место
    569 06CA  CD 21                                 int     21h
    570
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 11
tsr.asm


    571 06CC  B8 252F                       mov     AX, 252Fh
    572 06CF  26: 8B 16  015Ar              mov DX, ES:old_int2FhOffset  ; возвращаем вектор прерывания
    573 06D4  26: 8E 1E  015Cr          mov DS, ES:old_int2FhSegment         ; на место
    574 06D9  CD 21                         int     21h
    575
    576 06DB  2E: 8E 06  002C               mov     ES, CS:2Ch       ; загрузим в ES адрес окружения
    577 06E0  B4 49                         mov     AH, 49h          ; выгрузим из памяти окружение
    578 06E2  CD 21                         int     21h
    579 06E4  72 0B                         jc _notRemove
    580
    581 06E6  0E                    push    CS
    582 06E7  07                    pop     ES       ;в ES - адрес резидентной программы
    583 06E8  B4 49                         mov     AH, 49h  ;выгрузим  из памяти резидент
    584 06EA  CD 21                         int     21h
    585 06EC  72 03                         jc _notRemove
    586 06EE  EB 15 90                      jmp _unloaded
    587
    588 06F1                        _notRemove: ; не удалось выполнить  выгрузку
    589                                 ; вывод сообщения о неудачной выгрузке
    590 06F1  B4 03                         mov AH, 03h                             ; получаем позицию
        курсора
    591 06F3  CD 10                         int 10h
    592 06F5  BD 04E4r                      lea BP, noRemoveMsg
    593 06F8  B9 001D                       mov CX, noRemoveMsg_length
    594 06FB  B3 07                         mov BL, 0111b
    595 06FD  B8 1301                       mov AX, 1301h
    596 0700  CD 10                         int 10h
    597 0702  EB 12 90                      jmp _2Fh_exit
    598
    599 0705                        _unloaded:  ; выгрузка прошла успешно
    600                                 ; вывод сообщения об удачной выгрузке
    601 0705  B4 03                         mov AH, 03h                             ; получаем позицию
        курсора
    602 0707  CD 10                         int 10h
    603 0709  BD 04D2r                      lea BP, removedMsg
    604 070C  B9 0012                       mov CX, removedMsg_length
    605 070F  B3 07                         mov BL, 0111b
    606 0711  B8 1301                       mov AX, 1301h
    607 0714  CD 10                         int 10h
    608
    609 0716                        _2Fh_exit:
    610 0716  5B                    pop BX
    611 0717  5A                    pop     DX
    612 0718  07                    pop     ES
    613 0719  1F                    pop     DS
    614 071A  CF                    iret
    615 071B                        new_int2Fh  endp
    616
    617                             ;=== Процедура вывода подписи (ФИО, группа)
    618                             ;=== Настраивается  значениями переменных в  начале исходника
    619                             ;===
    620 071B                        printSignature proc
    621 071B  50                    push AX
    622 071C  52                    push DX
    623 071D  51                    push CX
    624 071E  53                    push BX
    625 071F  06                    push ES
    626 0720  54                    push SP
    627 0721  55                    push BP
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 12
tsr.asm



    628 0722  56                     push SI
    629 0723  57                     push DI
    630
    631 0724  33 C0                          xor AX, AX
    632 0726  33 DB                          xor BX, BX
    633 0728  33 D2                          xor DX, DX
    634
    635 072A  B4 03                          mov AH, 03h                                ;чтение
текущей позиции курсора
    636 072C  CD 10                          int 10h
    637 072E  52                     push DX                                            ;помещаем
информацию о        +
    638                              положении курсора в стек
    639
    640 072F  83 3E 0162r 00                 cmp printPos, 0
    641 0734  74 0E                          je _printTop
    642
    643 0736  83 3E 0162r 01                 cmp printPos, 1
    644 073B  74 0E                          je _printCenter
    645
    646 073D  83 3E 0162r 02                 cmp printPos, 2
    647 0742  74 0E                          je _printBottom
    648
    649                                      ;все числа  подобраны на глаз...
    650 0744                                 _printTop:
    651 0744  B6 00                                  mov DH, 0
    652 0746  B2 0F                                  mov DL, 15
    653 0748  EB 0F 90                               jmp _actualPrint
    654
    655 074B                                 _printCenter:
    656 074B  B6 09                                  mov DH, 9
    657 074D  B2 0F                                  mov DL, 15
    658 074F  EB 08 90                               jmp _actualPrint
    659
    660 0752                                 _printBottom:
    661 0752  B6 13                                  mov DH, 19
    662 0754  B2 0F                                  mov DL, 15
    663 0756  EB 01 90                               jmp _actualPrint
    664
    665 0759                                 _actualPrint:
    666 0759  B4 0F                                  mov AH, 0Fh                        ;чтение
текущего видеорежима. в+
    667                              BH - текущая страница
    668 075B  CD 10                                  int 10h
    669
    670 075D  0E                             push CS
    671 075E  07                             pop ES                                     ;указываем
        ES на CS
    672
    673                                              ;вывод 'верхушки' таблицы
    674 075F  52                             push DX
    675 0760  BD 0411r                               lea BP, tableTop
;помещаем в BP указатель на    +
    676                              выводимую строку
    677 0763  B9 0033                                mov CX, tableTop_length          ;в CX - длина строки
    678 0766  B3 07                                  mov BL, 0111b                      ;цвет
выводимого текста ref:       +
    679                              http://en.wikipedia.org/wiki/BIOS_color_attributes
    680 0768  B8 1301                                mov AX, 1301h
;AH=13h - номер ф-ии, AL=01h - +
    681                              курсор перемещается при выводе каждого из символов  строки
    682 076B  CD 10                                  int 10h
    683 076D  5A                             pop DX
    684 076E  FE C6                                  inc DH
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 13
tsr.asm



    685
    686
    687                                             ;вывод первой линии
    688 0770  52                                push DX
    689 0771  BD 0164r                              lea BP, signatureLine1
    690 0774  B9 0033                               mov CX, Line1_length
    691 0777  B3 07                                 mov BL, 0111b
    692 0779  B8 1301                               mov AX, 1301h
    693 077C  CD 10                                 int 10h
    694 077E  5A                                pop DX
    695 077F  FE C6                                 inc DH
    696
    697                                             ;вывод второй линии
    698 0781  52                                push DX
    699 0782  BD 0197r                              lea BP, signatureLine2
    700 0785  B9 0034                               mov CX, Line2_length
    701 0788  B3 07                                 mov BL, 0111b
    702 078A  B8 1301                               mov AX, 1301h
    703 078D  CD 10                                 int 10h
    704 078F  5A                                pop DX
    705 0790  FE C6                                 inc DH
    706
    707                                             ;вывод третьей линии
    708 0792  52                                push DX
    709 0793  BD 01CBr                              lea BP, signatureLine3
    710 0796  B9 0033                               mov CX, Line3_length
    711 0799  B3 07                                 mov BL, 0111b
    712 079B  B8 1301                               mov AX, 1301h
    713 079E  CD 10                                 int 10h
    714 07A0  5A                                pop DX
    715 07A1  FE C6                                 inc DH
    716
    717                                             ;вывод 'низа' таблицы
    718 07A3  52                                push DX
    719 07A4  BD 0444r                              lea BP, tableBottom
    720 07A7  B9 0033                               mov CX, tableBottom_length
    721 07AA  B3 07                                 mov BL, 0111b
    722 07AC  B8 1301                               mov AX, 1301h
    723 07AF  CD 10                                 int 10h
    724 07B1  5A                                pop DX
    725 07B2  FE C6                                 inc DH
    726
    727 07B4  33 DB                                 xor BX, BX
    728 07B6  5A                                pop DX
;восстанавливаем из стека           +
    729                                  прежнее положение курсора
    730 07B7  B4 02                                 mov AH, 02h                             ;меняем
положение курсора на   +
    731                                  первоначальное
    732 07B9  CD 10                                 int 10h
    733 07BB  E8 FD4B                               call changeFx
    734
    735 07BE  5F                         pop DI
    736 07BF  5E                         pop SI
    737 07C0  5D                         pop BP
    738 07C1  5C                         pop SP
    739 07C2  07                         pop ES
    740 07C3  5B                         pop BX
    741 07C4  59                         pop CX
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 14
tsr.asm



    742 07C5  5A                    pop DX
    743 07C6  58                    pop AX
    744
    745 07C7  C3                    ret
    746 07C8                        printSignature endp
    747
    748                             ;=== Функция, которая в зависимости от флага cursiveEnabled меняет
        начертание символа с курсива+
    749                             на обычное и наоброт
    750                             ;=== Сама смена происходит  в процедуре changeFont,  а здесь
        подготавливаются данные
    751 07C8                        setCursive  proc
    752 07C8  06                    push ES ; сохраняем регистры
    753 07C9  50                    push AX
    754 07CA  0E                    push CS
    755 07CB  07                    pop ES
    756
    757 07CC  80 3E 0130r FF                cmp cursiveEnabled, true
    758 07D1  75 30                         jne _restoreSymbol
    759                                     ; если флаг равен true, выполняем замену символа на курсивный
вариант,
    760                                     ; предварительно сохраняя старый символ в savedSymbol
    761
    762 07D3  E8 004C                       call saveFont
    763 07D6  8A 0E 0141r                   mov CL, charToCursiveIndex
    764 07DA                        _shifTtable:
    765                                     ; мы получаем в BP  таблицу  всех символов. адрес указывает на
символ 0
    766                                     ; поэтому нуэно совершить сдвиг 16*X - где  X - код символа
    767 07DA  83 C5 10                      add BP, 16
    768 07DD  E2 FB                         loop _shiftTable
    769
    770                                     ; при savefont смещается регистр ES
    771                                     ; поэтому приходится делать такие махинации, чтобы
    772                                     ; записать  полученный элемент в savedSymbol
    773                                     ; swap(ES,  DS) и сохранение старого значения DS
    774 07DF  1E                    push DS
    775 07E0  58                    pop AX
    776 07E1  06                    push ES
    777 07E2  1F                    pop DS
    778 07E3  50                    push AX
    779 07E4  07                    pop ES
    780 07E5  50                    push AX
    781
    782 07E6  8B F5                         mov SI, BP
    783 07E8  BF 0142r                      lea DI, savedSymbol
    784                                     ; сохраняем в переменную savedSymbol
    785                                     ; таблицу нужного символа
    786 07EB  B9 0010                       mov CX, 16
    787                                     ; movsb из  DS:SI в  ES:DI
    788 07EE  F3> A4                        rep movsb
    789                                     ; исходные  позиции  сегментов  возвращены
    790 07F0  1F                    pop DS ; восстановление DS
    791
    792                                     ; заменим написание символа на курсив
    793 07F1  B9 0001                       mov CX, 1
    794 07F4  B6 00                         mov DH, 0
    795 07F6  8A 16 0141r                   mov DL, charToCursiveIndex
    796 07FA  BD 0131r                      lea BP, cursiveSymbol
    797 07FD  E8 0015                       call changeFont
    798 0800  EB 10 90                      jmp _exitSetCursive
```

```
    799
    800 0803                        _restoreSymbol:
    801                                     ; если флаг равен 0, выполняем замену курсивного символа на
старый  вариант
    802
    803 0803  B9 0001                       mov CX, 1
    804 0806  B6 00                         mov DH, 0
    805 0808  8A 16 0141r                   mov DL, charToCursiveIndex
    806 080C  BD 0142r                      lea bp, savedSymbol
    807 080F  E8 0003                       call changeFont
    808
    809 0812                        _exitSetCursive:
    810 0812  58                    pop AX
    811 0813  07                    pop ES
    812 0814  C3                    ret
    813 0815                        setCursive  endp
    814
    815                             ;=== Функция смены  начертания символа (курсив/нормальное)
    816                             ;===
    817                             ; *** входные данные
    818                             ; DL = номер символа для замены
    819                             ; CX = Кол-во символов заменяемых изображений символов
    820                             ; (начиная  с символа указанного в DX)
    821                             ; ES:bp = адрес таблицы
    822                             ;
    823                             ; *** описание работы процедуры
    824                             ; Происходит вызов  int 10h (видеосервис)
    825                             ; с функцией AH = 11h (функции знакогенератора)
    826                             ; Параметр  AL = 0 сообщает, что будет заменено изображение
    827                             ; символа для текущего шрифта
    828                             ; В случаях, когда  AL = 1 или 2, будет заменено изображение
    829                             ; только для опредленного шрифта (8x14 и 8x8 соответственно)
    830                             ; Параметр  BH = 0Eh сообщает, что на опредление каждого изображения
символа
    831                             ; расходуется по 14 байт (режим 8x14 бит как раз 14 байт)
    832                             ; Параметр  BL = 0 - блок шрифта для загрузки (от 0  до 4)
    833                             ;
    834                             ; *** результат
    835                             ; изображение указанного(ых) символа(ов) будет заменено
    836                             ; на предложенное пользователем.
    837                             ; Изменению подвергнутся все символы, находящиеся на экране,
    838                             ; то есть если изображение  заменено, старый вариант нигде уже не
проявится
    839
    840 0815                        changeFont  proc
    841 0815  50                    push AX
    842 0816  53                    push BX
    843 0817  B8 1100                       mov AX, 1100h
    844 081A  BB 1000                       mov BX, 1000h
    845 081D  CD 10                         int 10h
    846 081F  58                    pop AX
    847 0820  5B                    pop BX
    848 0821  C3                    ret
    849 0822                        changeFont  endp
    850
    851                             ;=== Функция сохранения нормального начертания символа
    852                             ;===
    853                             ; *** входные данные
    854                             ; BH - тип  возвращаемой символьной  таблицы
    855                             ;   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 )

 0822                           saveFont proc
 0822  50                       push AX
 0823  53                       push BX
 0824  B8 1130                          mov AX, 1130h
 0827  BB 0600                          mov BX, 0600h
 082A  CD 10                            int 10h
 082C  58                       pop AX
 082D  5B                       pop BX
 082E  C3                       ret
 082F                           saveFont endp


                                ;=== Отсюда начинается выполнение основной  части программы ===;
                                ;===
 082F                           _initTSR: ; старт резидента
 082F  B4 03                            mov AH, 03h
 0831  CD 10                            int 10h
 0833  52                       push DX
 0834  B4 00                            mov AH,00h ; установка видеорежима  (83h  текст  80x25  16/8
CGA,EGA  b800    Comp,RGB,  +
                                Enhanced),  без очистки экрана
 0836  B0 83                            mov AL,83h
 0838  CD 10                            int 10h
 083A  5A                       pop DX
 083B  B4 02                            mov AH, 02h
 083D  CD 10                            int 10h


 083F  E8 0095                      call commandParamsParser
 0842  B8 3509                          mov AX,3509h                    ; получить  в ES:BX  вектор 09
 0845  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!
                                        ;cmp unloadTSR, true
                                        ;je _removingOnParameter
                                        ;jmp _notRemovingNow

 0847                                   _removingOnParameter:
 0847  B4 FF                                    mov AH, 0FFh
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 17
tsr.asm



    913 0849  B0 00                                   mov AL, 0
    914 084B  CD 2F                                   int 2Fh
    915 084D  80 FC 69                                cmp AH, 'i'  ; проверка того, загружена ли  уже
программа
    916 0850  74 69                                   je _remove
    917                                               ;mov AH, 09h                          ;@ для
выгрузки резидента по повторному+
    918                                 запуску закомментировать эту строку
    919                                               ;lea DX, notInstalledMsg        ;@ для выгрузки
резидента по повторному запуску+
    920                                 закомментировать эту строку
    921                                               ;int 21h                                    ;@
        для выгрузки резидента по   +
    922                                 повторному  запуску закомментировать эту строку
    923                                               ;int 20h                                    ;@
        для выгрузки резидента по   +
    924                                 повторному  запуску закомментировать эту строку
    925
    926 0852                                      _notRemovingNow:
    927
    928 0852  80 3E 015Fr FF                      cmp notLoadTSR, true                    ; если была выведена
справка
    929 0857  74 03                               je _exit_tmp                                ;
просто выходим
    930
    931                                           ;@ Если по варианту необходимо выгружать резидент по
повторному запуску, то      +
    932                                 комментируем 5 строк ниже
    933                                           ;@ если необходимо  выгружать по параметру коммандной строки,
то оставляем их
    934                                           ;mov AH, 0FFh
    935                                           ;mov AL, 0
    936                                           ;int 2Fh
    937                                           ;cmp AH, 'i'  ; проверка того, загружена ли уже программа
    938                                           ;je _alreadyInstalled
    939
    940 0859  EB 04 90                            jmp _tmp
    941
    942 085C                                      _exit_tmp:
    943 085C  EB 77 90                                    jmp _exit
    944
    945 085F                                      _tmp:
    946 085F  06                                push ES
    947 0860  A1 002C                               mov AX, DS:[2Ch]                ; psp
    948 0863  8E C0                                 mov ES, AX
    949 0865  B4 49                                 mov AH, 49h                     ; хватит памяти чтоб остаться
    950 0867  CD 21                                 int 21h                 ; резидентом?
    951 0869  07                pop ES
    952 086A  72 62                                 jc _notMem                      ; не хватило -  выходим
    953
    954                                             ;== int 09h ==;
    955
    956 086C  2E: 89 1E  0152r                      mov     word ptr CS:old_int9hOffset, BX
    957 0871  2E: 8C 06  0154r                      mov     word ptr CS:old_int9hSegment, ES
    958 0876  B8 2509                               mov AX, 2509h                   ; установим вектор на 09
    959 0879  BA 05ABr                              mov DX, offset  new_int9h           ; прерывание
    960 087C  CD 21                                 int 21h
    961
    962                                             ;== int 1Ch ==;
    963 087E  B8 351C                               mov AX,351Ch                    ; получить  в ES:BX вектор 1C
    964 0881  CD 21                                 int 21h                 ; прерывания
    965 0883  2E: 89 1E  0156r                      mov     word ptr CS:old_int1ChOffset, BX
    966 0888  2E: 8C 06  0158r                      mov     word ptr CS:old_int1ChSegment, ES
    967 088D  B8 251C                               mov AX, 251Ch                   ; установим вектор  на 1C
    968 0890  BA 0660r                              mov DX, offset new_int1Ch           ; прерывание
    969 0893  CD 21                                 int 21h
```

```
    970
    971                                     ;== int 2Fh ==;
    972 0895  B8 352F                       mov AX,352Fh                    ; получить  в ES:BX вектор 1C
    973 0898  CD 21                     int 21h                 ; прерывания
    974 089A  2E: 89 1E  015Ar              mov     word ptr CS:old_int2FhOffset, BX
    975 089F  2E: 8C 06  015Cr              mov     word ptr CS:old_int2FhSegment, ES
    976 08A4  B8 252F                       mov AX, 252Fh                   ; установим вектор  на 2F
    977 08A7  BA 068Er                      mov DX, offset new_int2Fh            ; прерывание
    978 08AA  CD 21                         int 21h
    979
    980 08AC  E8 FC5A                       call changeFx
    981 08AF  BA 0477r                  mov DX, offset  installedMsg        ; выводим что все ок
    982 08B2  B4 09                     mov AH, 9
    983 08B4  CD 21                     int 21h
    984 08B6  BA 082Fr                  mov DX, offset  _initTSR       ; остаемся в памяти резидентом
    985 08B9  CD 27                     int 27h                 ; и выходим
    986                                 ; конец основной программы
    987 08BB                        _remove: ;  выгрузка программы из памяти
    988 08BB  B4 FF                         mov AH, 0FFh
    989 08BD  B0 01                         mov AL, 1
    990 08BF  CD 2F                         int 2Fh
    991 08C1  EB 12 90                      jmp _exit
    992 08C4                        _alreadyInstalled:
    993 08C4  B4 09                         mov AH, 09h
    994 08C6  BA 048Ar                      lea DX, alreadyInstalledMsg
    995 08C9  CD 21                         int 21h
    996 08CB  EB 08 90                      jmp _exit
    997 08CE                        _notMem:                            ; не хватает памяти, чтобы
остаться резидентом
    998 08CE  BA 04A0r                  mov DX, offset  noMemMsg
    999 08D1  B4 09                     mov AH, 9
   1000 08D3  CD 21                     int 21h
   1001 08D5                        _exit:                              ; выход
   1002 08D5  CD 20                     int 20h
   1003
   1004                             ;=== Процедура проверки параметров  ком. строки ===;
   1005                             ;===
   1006 08D7                        commandParamsParser proc
   1007 08D7  0E                    push CS
   1008 08D8  07                    pop ES
   1009 08D9  C6 06 015Er 00               mov unloadTSR, 0
   1010 08DE  C6 06 015Fr 00               mov notLoadTSR, 0
   1011
   1012 08E3  BE 0080                      mov SI, 80h                     ;SI=смещение командной
строки.
   1013 08E6  AC                    lodsb                                           ;Получим кол-во
символов.
   1014 08E7  0A C0                        or AL, AL                               ;Если 0 символов
введено,
   1015 08E9  74 37                        jz _exitHelp                            ;то все в порядке.
   1016
   1017 08EB                               _nextChar:
   1018
   1019 08EB  46                    inc SI                                          ;Теперь SI  указывает
на первый символ  +
   1020                             строки.
   1021
   1022 08EC  80 3C 0D                     cmp [SI], BYTE ptr  13
   1023 08EF  74 31                        je _exitHelp
   1024
   1025
   1026 08F1  AD                           lodsw                                   ;Получаем два символа
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 19
tsr.asm


   1027 08F2  3D 3F2F                                   cmp AX, '?/'                            ;Это '/?'
(данные расположены в       +
   1028                                 обратном порядк, т.е. AL:AH вместо  AH:AL)
   1029 08F5  74 03                                     je _question
   1030                                                 ;cmp AX, 'u/'
   1031                                                 ;je _finishTSR
   1032
   1033                                                 ;cmp AH, '/'
   1034                                                 ;je _errorParam
   1035
   1036 08F7  EB 29 90                                  jmp _exitHelp
   1037
   1038 08FA                                    _question:
   1039                                                 ; вывод строки помощи
   1040 08FA  B4 03                                             mov AH,03
   1041 08FC  CD 10                                             int 10h
   1042 08FE  BD 01FEr                                          lea BP, helpMsg
   1043 0901  B9 01EE                                           mov CX, helpMsg_length
   1044 0904  B3 07                                             mov BL, 0111b
   1045 0906  B8 1301                                           mov AX, 1301h
   1046 0909  CD 10                                             int 10h
   1047                                                 ; конец вывода строки помощи
   1048 090B  F6 16 015Fr                               not notLoadTSR          ;флаг того, что необходимо
        не загружать резидент
   1049 090F  EB DA                                     jmp _nextChar
   1050
   1051                                         ;@ === Удаление резидента из памяти ===
   1052                                         ;@ Если по варианту необходимо выгружать резидент по
параметру '/u' коммандной строки,
   1053                                         ;@ нужно использовать следующий код, в остальных случаях
необходимо закомменитровать
   1054                                         ;@ этот код, кроме  названия метки!  (по желанию можно
избавиться и от метки, но           +
   1055                                 аккуратно просмотреть использование)
   1056 0911                                    _finishTSR:
   1057                                                 ;not unloadTSR                 ;флаг того, что
необходимо выгузить резидент
   1058                                                 ;jmp _nextChar
   1059
   1060                                         ;jmp _exitHelp
   1061
   1062 0911                                    _errorParam:
   1063                                                 ;вывод строки
   1064 0911  B4 03                                             mov AH,03
   1065 0913  CD 10                                             int 10h
   1066 0915  BD 03ECr                                          lea BP, CS:errorParamMsg
   1067 0918  B9 0025                                           mov CX, errorParamMsg_length
   1068 091B  B3 07                                             mov BL, 0111b
   1069 091D  B8 1301                                           mov AX, 1301h
   1070 0920  CD 10                                             int 10h
   1071                                                 ;конец вывода строки
   1072 0922                                    _exitHelp:
   1073 0922  C3                                ret
   1074 0923                                    commandParamsParser endp
   1075
   1076 0923                                    code ends
   1077                                         end _start
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 20
Symbol Table



Symbol Name                       Type   Value

??DATE                            Text   "05/23/23"
??FILENAME                        Text   "tsr     "
??TIME                            Text   "14:21:39"
??VERSION                         Number 030A
@CPU                              Text   0101H
@CURSEG                           Text   CODE
@FILENAME                         Text   TSR
@WORDSIZE                         Text   2
ALREADYINSTALLEDMSG               Byte   CODE:048A
CHANGEFONT                        Near   CODE:0815
CHANGEFX                  Near    CODE:0509
CHARTOCURSIVEINDEX                Byte   CODE:0141
COMMANDPARAMSPARSER               Near   CODE:08D7
COUNTER                           Word   CODE:0160
CURSIVEENABLED                    Byte   CODE:0130
CURSIVESYMBOL                     Byte   CODE:0131
ERRORPARAMMSG                     Byte   CODE:03EC
ERRORPARAMMSG_LENGTH              Number 0025
F1_TXT                            Byte   CODE:0501
F2_TXT                            Byte   CODE:0503
F3_TXT                            Byte   CODE:0505
F4_TXT                            Byte   CODE:0507
FX_LENGTH                         Number 0002
HELPMSG                           Byte   CODE:01FE
HELPMSG_LENGTH                    Number 01EE
IGNOREDCHARS                      Byte   CODE:0103
IGNOREDLENGTH                     Number 0020
IGNOREENABLED                     Byte   CODE:0123
INSTALLEDMSG                      Byte   CODE:0477
LINE1_LENGTH                      Number 0033
LINE2_LENGTH                      Number 0034
LINE3_LENGTH                      Number 0033
NEW_INT1CH                        Far    CODE:0660
NEW_INT2FH                        Near   CODE:068E
NEW_INT9H                         Far    CODE:05AB
NOMEMMSG                  Byte    CODE:04A0
NOREMOVEMSG                       Byte   CODE:04E4
NOREMOVEMSG_LENGTH                Number 001D
NOTINSTALLEDMSG                   Byte   CODE:04B4
NOTLOADTSR                        Byte   CODE:015F
OLD_INT1CHOFFSET          Word    CODE:0156
OLD_INT1CHSEGMENT                 Word   CODE:0158
OLD_INT2FHOFFSET          Word    CODE:015A
OLD_INT2FHSEGMENT                 Word   CODE:015C
OLD_INT9HOFFSET                   Word   CODE:0152
OLD_INT9HSEGMENT          Word    CODE:0154
PRINTDELAY                        Number 0005
PRINTPOS                  Word    CODE:0162
PRINTSIGNATURE                    Near   CODE:071B
REMOVEDMSG                        Byte   CODE:04D2
REMOVEDMSG_LENGTH                 Number 0012
SAVEDSYMBOL                       Byte   CODE:0142
SAVEFONT                  Near    CODE:0822
SETCURSIVE                        Near   CODE:07C8
```

```
Turbo Assembler  Version 3.1        05/23/23 14:21:40       Page 21
Symbol Table


SIGNATURELINE1                     Byte   CODE:0164
SIGNATURELINE2                     Byte   CODE:0197
SIGNATURELINE3                     Byte   CODE:01CB
SIGNATUREPRINTINGENABLED   Byte    CODE:012F
TABLEBOTTOM                        Byte   CODE:0444
TABLEBOTTOM_LENGTH                 Number 0033
TABLETOP                   Byte    CODE:0411
TABLETOP_LENGTH                    Number 0033
TRANSLATEENABLED           Byte    CODE:012E
TRANSLATEFROM                      Byte   CODE:0124
TRANSLATELENGTH                    Number 0005
TRANSLATETO                        Byte   CODE:0129
TRUE                               Number 00FF
UNLOADTSR                          Byte   CODE:015E
_2FH_EXIT                          Near   CODE:0716
_2FH_STD                   Near    CODE:069E
_ACTUALPRINT                       Near   CODE:0759
_ALREADYINSTALLED                  Near   CODE:08C4
_ALREADY_INSTALLED                 Near   CODE:06A3
_BLOCK                             Near   CODE:062E
_CHECKF1                   Near    CODE:0518
_CHECKF2                   Near    CODE:053C
_CHECKF3                   Near    CODE:055D
_CHECKF4                   Near    CODE:057E
_CHECK_IGNORED                     Near   CODE:0622
_CHECK_TRANSLATE           Near    CODE:0636
_CHECK_TRANSLATE_LOOP              Near   CODE:0643
_DONTPRINT                         Near   CODE:0687
_ERRORPARAM                        Near   CODE:0911
_EXIT                              Near   CODE:08D5
_EXITHELP                          Near   CODE:0922
_EXITSETCURSIVE                    Near   CODE:0812
_EXIT_TMP                          Near   CODE:085C
_F1                                Near   CODE:05BD
_F2                                Near   CODE:05CB
_F3                                Near   CODE:05DC
_F4                                Near   CODE:05EA
_FINISHTSR                         Near   CODE:0911
_GO                                Near   CODE:0612
_GREENF1                   Near    CODE:0535
_GREENF2                   Near    CODE:0559
_GREENF3                   Near    CODE:057A
_GREENF4                   Near    CODE:059B
_INITTSR                   Near    CODE:082F
_LETSPRINT                         Near   CODE:067A
_NEXTCHAR                          Near   CODE:08EB
_NOTMEM                            Near   CODE:08CE
_NOTREMOVE                         Near   CODE:06F1
_NOTREMOVINGNOW                    Near   CODE:0852
_NOTTOPRINT                        Near   CODE:068C
_OUTFX                             Near   CODE:059F
_PRINTBOTTOM                       Near   CODE:0752
_PRINTCENTER                       Near   CODE:074B
_PRINTTOP                          Near   CODE:0744
_QUESTION                          Near   CODE:08FA
_QUIT                              Near   CODE:0658
_REDF1                             Near   CODE:052E
```

```
Turbo Assembler   Version 3.1        05/23/23 14:21:40       Page 22
Symbol Table


_REDF2                            Near   CODE:0552
_REDF3                            Near   CODE:0573
_REDF4                            Near   CODE:0594
_REMOVE                           Near   CODE:08BB
_REMOVINGONPARAMETER              Near   CODE:0847
_RESTORESYMBOL                    Near   CODE:0803
_SHIFTTABLE                       Near   CODE:07DA
_START                            Near   CODE:0100
_TEST_FX                  Near    CODE:05BB
_TMP                              Near   CODE:085F
_TRANSLATE                        Near   CODE:064F
_TRANSLATE_OR_IGNORE              Near   CODE:05F8
_UNINSTALL                        Near   CODE:06A6
_UNLOADED                         Near   CODE:0705

Groups & Segments                 Bit Size Align  Combine Class

CODE                              16  0923 Para    none    CODE
```