**Московский государственный технический университет им. Н.Э. Баумана**

|  |  |
| --- | --- |
| Утверждаю: |  |
| Большаков С.А. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г. |

Курсовая работа по курсу «Системное программирование»

**«Резидентная программа (TSR)»**

Исходный текст программного продукта

(вид документа)

писчая бумага

(вид носителя)

44

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5-81 | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Белоусов Е.А. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_ 2021 г. |

Москва – 2021

# СОДЕРЖАНИЕ

[СОДЕРЖАНИЕ 2](#_Toc8560881)

[1. TSR.ASM 2](#_Toc8560882)

# TSR.ASM

Turbo Assembler Version 4.1 05/15/19 15:22:33 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 ;

9 ; Авторы:

10 ; МГТУ им. Н.Э. Баумана, ИУ5-81, 2021 г.

11 ; Белоусов Е. А.

12 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

13

14 0000 code segment 'code'

15 assume CS:code, DS:code

16 org 100h

17 0100 \_start:

18

19 0100 E9 0736 jmp \_initTSR ; на начало программы

20

21 ; данные

22 0103 61 62 63 64 65 66 67+ replaceWith DB +

23 68 69 6A 6B 6C 6D 6E+ 'QWERTYUIOP{}ASDFGHJKL:"ZXCVBNM<>' ;@ список множества латинских символов

24 6F 70 71 72 73 74 75+

25 76 77 78 79 7A 41 42+

26 43 44 45 46 47 48 49+

27 4A 4B 4C 4D 4E 4F 50+

28 51 52 53 54 55 56 57+

29 58 59 5A

30 0137 E4 A8 E1 A2 E3 A0 AF+ ignoredChars DB +

31 E0 E8 AE AB A4 EC E2+ 'ЙЦУКЕНГШЩЗХЪФЫВАПРОЛДЖЭЯЧСМИТЬБЮ' ;@ список игнорируемых символов

32 E9 A7 A9 AA EB A5 A3+

33 AC E6 E7 AD EF 94 88+

34 91 82 93 80 8F 90 98+

35 8E 8B 84 9C 92 99 87+

36 89 8A 9B 85 83 8C 96+

37 97 8D 9F

38 =0034 ignoredLength equ $-ignoredChars +

39 ; длина строки ignoredChars

40 016B 00 ignoreEnabled DB 0 +

41 ; флаг функции игнорирования ввода

42 016C 54 3A 50 42 52 translateFrom DB 'T:PBR' +

43 ;@ символы для замены (АБВГД на англ. раскладке)

44 0171 85 86 87 88 8A translateTo DB 'ЕЖЗИК' +

45 ;@ символы на которые будет идти замена

46 =0005 translateLength equ $-translateTo +

47 ; длина строки trasnlateFrom

48 0176 00 translateEnabled DB 0 +

49 ; флаг функции перевода

50

51 0177 00 signaturePrintingEnabled DB 0 +

52 ; флаг функции вывода информации об авторе

53 0178 00 cursiveEnabled DB 0 +

54 ; флаг перевода символа в курсив

55

56 0179 00 cursiveSymbol DB 00000000b

57 017A 1F DB 00011111b

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 2

tsr.asm

58 017B 18 DB 00011000b

59 017C 18 DB 00011000b

60 017D 18 DB 00011000b

61 017E 30 DB 00110000b

62 017F 30 DB 00110000b

63 0180 3C DB 00111100b

64 0181 36 DB 00110110b

65 0182 66 DB 01100110b

66 0183 66 DB 01100110b

67 0184 66 DB 01100110b

68 0185 7C DB 01111100b

69 0186 00 DB 00000000b

70 0187 00 DB 00000000b

71

72 0188 81 charToCursiveIndex DB 'Б' +

73 ;@ символ для замены

74 0189 10\*(FF) savedSymbol DB 16 dup(0FFh) +

75 ; переменная для хранения старого символа

76

77 =00FF true equ 0FFh +

78 ; константа истинности

79 0199 ???? old\_int9hOffset DW ? +

80 ; адрес старого обработчика int 9h

81 019B ???? old\_int9hSegment DW ? +

82 ; сегмент старого обработчика int 9h

83 019D ???? old\_int1ChOffset DW ? +

84 ; адрес старого обработчика int 1Ch

85 019F ???? old\_int1ChSegment DW ? +

86 ; сегмент старого обработчика int 1Ch

87 01A1 ???? old\_int2FhOffset DW ? +

88 ; адрес старого обработчика int 2Fh

89 01A3 ???? old\_int2FhSegment DW ? +

90 ; сегмент старого обработчика int 2Fh

91

92 01A5 00 unloadTSR DB 0 +

93 ; 1 - выгрузить резидент

94 01A6 00 notLoadTSR DB 0 +

95 ; 1 - не загружать

96 01A7 0000 counter DW 0

97 =0005 printDelay equ 5 +

98 ;@ задержка перед выводом "подписи" в секундах

99 01A9 0000 printPos DW 0 +

100 ;@ положение подписи на экране. 0 - верх, 1 - центр, 2 - низ

101

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

103 (1я строка).

104 01AB B3 81 A5 AB AE E3 E1+ signatureLine1 DB 179, 'Белоусов Евгений +

105 AE A2 20 85 A2 A3 A5+ ', 179

106 AD A8 A9 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 20 20 20 20+

111 20 20 B3

112 =0034 Line1\_length equ $-signatureLine1

113 01DF B3 88 93 35 2D 34 31+ signatureLine2 DB 179, 'ИУ5-41 +

114 20 20 20 20 20 20 20+ ', 179

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 3

tsr.asm

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 20 20 20 20+

120 20 20 B3

121 =0034 Line2\_length equ $-signatureLine2

122 0213 B3 82 A0 E0 A8 A0 AD+ signatureLine3 DB 179, 'Вариант 4 +

123 E2 20 34 20 20 20 20+ ', 179

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 20 20 20 20+

129 20 20 B3

130 =0034 Line3\_length equ $-signatureLine3

131 0247 3E 74 73 72 2E 63 6F+ helpMsg DB '>tsr.com [/?]', 10, 13

132 6D 20 5B 2F 3F 5D 0A+

133 0D

134 0256 20 5B 2F 3F 5D 20 2D+ DB ' [/?] - вывод данной справки', 10, 13

135 20 A2 EB A2 AE A4 20+

136 A4 A0 AD AD AE A9 20+

137 E1 AF E0 A0 A2 AA A8+

138 0A 0D

139 0274 20 43 6E 74 72 2B 55+ DB ' Cntr+U/u - выгрузка резидента из памяти', 10, 13

140 2F 75 20 2D 20 A2 EB+

141 A3 E0 E3 A7 AA A0 20+

142 E0 A5 A7 A8 A4 A5 AD+

143 E2 A0 20 A8 A7 20 AF+

144 A0 AC EF E2 A8 0A 0D

145 029E 20 20 46 32 20 20 2D+ DB ' F2 - вывод ФИО и группы по таймеру в верху экрана', 10, 13

146 20 A2 EB A2 AE A4 20+

147 94 88 8E 20 A8 20 A3+

148 E0 E3 AF AF EB 20 AF+

149 AE 20 E2 A0 A9 AC A5+

150 E0 E3 20 A2 20 A2 A5+

151 E0 E5 E3 20 ED AA E0+

152 A0 AD A0 0A 0D

153 02D4 20 20 46 33 20 20 2D+ DB ' F3 - включение и отключения курсивного вывода русского символа +

154 20 A2 AA AB EE E7 A5+ Б', 10, 13

155 AD A8 A5 20 A8 20 AE+

156 E2 AA AB EE E7 A5 AD+

157 A8 EF 20 AA E3 E0 E1+

158 A8 A2 AD AE A3 AE 20+

159 A2 EB A2 AE A4 A0 20+

160 E0 E3 E1 E1 AA AE A3+

161 AE 20 E1 A8 AC A2 AE+

162 AB A0 20 81 0A 0D

163 0319 20 20 46 34 20 20 2D+ DB ' F4 - включение и отключение частичной русификации клавиатуры +

164 20 A2 AA AB EE E7 A5+ (T:PBR -> ЕЖЗИК)', 10, 13

165 AD A8 A5 20 A8 20 AE+

166 E2 AA AB EE E7 A5 AD+

167 A8 A5 20 E7 A0 E1 E2+

168 A8 E7 AD AE A9 20 E0+

169 E3 E1 A8 E4 A8 AA A0+

170 E6 A8 A8 20 AA AB A0+

171 A2 A8 A0 E2 E3 E0 EB+

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 4

tsr.asm

172 28 54 3A 50 42 52 20+

173 2D 3E 20 85 86 87 88+

174 8A 29 0A 0D

175 036A 20 20 46 35 20 20 2D+ DB ' F5 - включение и отключение режима замены множества русских букв+

176 20 A2 AA AB EE E7 A5+ на множество латинских', 10, 13

177 AD A8 A5 20 A8 20 AE+

178 E2 AA AB EE E7 A5 AD+

179 A8 A5 20 E0 A5 A6 A8+

180 AC A0 20 A7 A0 AC A5+

181 AD EB 20 AC AD AE A6+

182 A5 E1 E2 A2 A0 20 E0+

183 E3 E1 E1 AA A8 E5 20+

184 A1 E3 AA A2 20 AD A0+

185 20 AC AD AE A6 A5 E1+

186 E2 A2 AE 20 AB A0 E2+

187 A8 AD E1 AA A8 E5 0A+

188 0D

189

190 =017F helpMsg\_length equ $-helpMsg

191 03C6 8E E8 A8 A1 AA A0 20+ errorParamMsg DB 'Ошибка параметров коммандной +

192 AF A0 E0 A0 AC A5 E2+ строки', 10, 13

193 E0 AE A2 20 AA AE AC+

194 AC A0 AD A4 AD AE A9+

195 20 E1 E2 E0 AE AA A8+

196 0A 0D

197 =0025 errorParamMsg\_length equ $-errorParamMsg

198

199 03EB DA 32\*(C4) BF tableTop DB 218, Line1\_length-2 dup+

200 (196), 191

201 =0034 tableTop\_length equ $-tableTop

202 041F C0 32\*(C4) D9 tableBottom DB 192, Line1\_length-2 dup (196), +

203 217

204 =0034 tableBottom\_length equ $-tableBottom

205

206 ; сообщения

207 0453 90 A5 A7 A8 A4 A5 AD+ installedMsg DB 'Резидент загружен!$'

208 E2 20 A7 A0 A3 E0 E3+

209 A6 A5 AD 21 24

210 0466 90 A5 A7 A8 A4 A5 AD+ alreadyInstalledMsg DB 'Резидент уже загружен$'

211 E2 20 E3 A6 A5 20 A7+

212 A0 A3 E0 E3 A6 A5 AD+

213 24

214 047C 8D A5 A4 AE E1 E2 A0+ noMemMsg DB 'Недостаточно памяти$'

215 E2 AE E7 AD AE 20 AF+

216 A0 AC EF E2 A8 24

217 0490 8D A5 20 E3 A4 A0 AB+ notInstalledMsg DB 'Не удалось загрузить резидент$'

218 AE E1 EC 20 A7 A0 A3+

219 E0 E3 A7 A8 E2 EC 20+

220 E0 A5 A7 A8 A4 A5 AD+

221 E2 24

222

223 04AE 90 A5 A7 A8 A4 A5 AD+ removedMsg DB 'Резидент выгружен'

224 E2 20 A2 EB A3 E0 E3+

225 A6 A5 AD

226 =0011 removedMsg\_length equ $-removedMsg

227

228 04BF 8D A5 20 E3 A4 A0 AB+ noRemoveMsg DB 'Не удалось выгрузить резидент'

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 5

tsr.asm

229 AE E1 EC 20 A2 EB A3+

230 E0 E3 A7 A8 E2 EC 20+

231 E0 A5 A7 A8 A4 A5 AD+

232 E2

233 =001D noRemoveMsg\_length equ $-noRemoveMsg

234

235 04DC 46 32 f1\_txt DB 'F2'

236 04DE 46 33 f2\_txt DB 'F3'

237 04E0 46 34 f3\_txt DB 'F4'

238 04E2 46 35 f4\_txt DB 'F5'

239 =0002 fx\_length equ $-f4\_txt

240

241 04E4 changeFx proc

242 04E4 50 push AX

243 04E5 53 push BX

244 04E6 51 push CX

245 04E7 52 push DX

246 04E8 55 push BP

247 04E9 06 push ES

248 04EA 33 DB xor BX, BX

249

250 04EC B4 03 mov AH, 03h

251 04EE CD 10 int 10h

252 04F0 52 push DX

253

254 04F1 0E push CS

255 04F2 07 pop ES

256

257 04F3 \_checkF1:

258 04F3 BD 04DCr lea BP, f1\_txt

259 04F6 B9 0002 mov CX, fx\_length

260 04F9 B7 00 mov BH, 0

261 04FB B6 00 mov DH, 0

262 04FD B2 4E mov DL, 78

263 04FF B8 1301 mov AX, 1301h

264

265 0502 80 3E 0177r FF cmp signaturePrintingEnabled, true

266 0507 74 07 je \_greenF1

267

268 0509 \_redF1:

269 0509 B3 4F mov BL, 01001111b ; red

270 050B CD 10 int 10h

271 050D EB 08 90 jmp \_checkF2

272

273 0510 \_greenF1:

274 0510 BD 04DCr lea BP, f1\_txt

275 0513 B3 2F mov BL, 00101111b ; green

276 0515 CD 10 int 10h

277

278 0517 \_checkF2:

279 0517 BD 04DEr lea BP, f2\_txt

280 051A B9 0002 mov CX, fx\_length

281 051D B7 00 mov BH, 0

282 051F B6 01 mov DH, 1

283 0521 B2 4E mov DL, 78

284 0523 B8 1301 mov AX, 1301h

285

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 6

tsr.asm

286 0526 80 3E 0178r FF cmp cursiveEnabled, true

287 052B 74 07 je \_greenF2

288

289 052D \_redF2:

290 052D B3 4F mov BL, 01001111b ; red

291 052F CD 10 int 10h

292 0531 EB 05 90 jmp \_checkF3

293

294 0534 \_greenF2:

295 0534 B3 2F mov BL, 00101111b ; green

296 0536 CD 10 int 10h

297

298 0538 \_checkF3:

299 0538 BD 04E0r lea BP, f3\_txt

300 053B B9 0002 mov CX, fx\_length

301 053E B7 00 mov BH, 0

302 0540 B6 02 mov DH, 2

303 0542 B2 4E mov DL, 78

304 0544 B8 1301 mov AX, 1301h

305

306 0547 80 3E 0176r FF cmp translateEnabled, true

307 054C 74 07 je \_greenF3

308

309 054E \_redF3:

310 054E B3 4F mov BL, 01001111b ; red

311 0550 CD 10 int 10h

312 0552 EB 05 90 jmp \_checkF4

313

314 0555 \_greenF3:

315 0555 B3 2F mov BL, 00101111b ; green

316 0557 CD 10 int 10h

317

318 0559 \_checkF4:

319 0559 BD 04E2r lea BP, f4\_txt

320 055C B9 0002 mov CX, fx\_length

321 055F B7 00 mov BH, 0

322 0561 B6 03 mov DH, 3

323 0563 B2 4E mov DL, 78

324 0565 B8 1301 mov AX, 1301h

325

326 0568 80 3E 016Br FF cmp ignoreEnabled, true

327 056D 74 07 je \_greenF4

328

329 056F \_redF4:

330 056F B3 4F mov BL, 01001111b ; red

331 0571 CD 10 int 10h

332 0573 EB 05 90 jmp \_outFx

333

334 0576 \_greenF4:

335 0576 B3 2F mov BL, 00101111b ; green

336 0578 CD 10 int 10h

337

338 057A \_outFx:

339 057A 5A pop DX

340 057B B4 02 mov AH, 02h

341 057D CD 10 int 10h

342

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 7

tsr.asm

343 057F 07 pop ES

344 0580 5D pop BP

345 0581 5A pop DX

346 0582 59 pop CX

347 0583 5B pop BX

348 0584 58 pop AX

349 0585 C3 ret

350 0586 changeFx endp

351

352 ;новый обработчик

353 0586 new\_int9h proc far

354 ; сохраняем значения всех, изменяемых регистров в стэке

355 0586 56 push SI

356 0587 50 push AX

357 0588 53 push BX

358 0589 51 push CX

359 058A 52 push DX

360 058B 06 push ES

361 058C 1E push DS

362 ; синхронизируем CS и DS

363 058D 0E push CS

364 058E 1F pop DS

365

366 058F B8 0040 mov AX, 40h ; 40h-сегмент,где хранятся флаги сост-я клавиатуры, кольц. +

367 буфер ввода

368 0592 8E C0 mov ES, AX

369 0594 E4 60 in AL, 60h ; записываем в AL скан-код нажатой клавиши

370

371 ;@ проверка на Ctrl+U, только для ИУ5-41

372 0596 3C 16 cmp AL, 22 ; была нажата клавиша U?

373 0598 75 24 jne \_test\_Fx

374 059A 26: 8A 26 0017 mov AH, ES:[17h] ; флаги клавиатуры

375 059F 80 E4 0F and AH, 00001111b

376 05A2 80 FC 04 cmp AH, 00000100b ; был ли нажат ctrl?

377 05A5 75 17 jne \_test\_Fx

378 ; выгрузка

379 05A7 B4 FF mov AH, 0FFh

380 05A9 B0 01 mov AL, 01h

381 05AB CD 2F int 2Fh

382 ; завершаем обработку нажатия

383

384 05AD E4 61 in AL, 61h ;контроллер состояния клавиатуры

385 05AF 0C 80 or AL, 10000000b ;пометим, что клавишу нажали

386 05B1 E6 61 out 61h, AL

387 05B3 24 7F and AL, 01111111b ;пометим, что клавишу отпустили

388 05B5 E6 61 out 61h, AL

389 05B7 B0 20 mov AL, 20h

390 05B9 E6 20 out 20h, AL ;отправим в контроллер прерываний признак конца +

391 прерывания

392

393 ; выходим

394 05BB E9 00A4 jmp \_quit

395

396 ;@ далее - код для всех вариантов

397

398 ;проверка F1-F4

399 05BE \_test\_Fx:

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 8

tsr.asm

400 05BE 2C 3A sub AL, 58 ; в AL теперь номер функциональной клавиши

401 05C0 \_F1:

402 05C0 3C 02 cmp AL, 2 ; F2

403 05C2 75 0A jne \_F2

404 05C4 F6 16 0177r not signaturePrintingEnabled

405 05C8 E8 FF19 call changeFx

406 05CB EB 2E 90 jmp \_translate\_or\_ignore

407 05CE \_F2:

408 05CE 3C 03 cmp AL, 3 ; F3

409 05D0 75 0D jne \_F3

410 05D2 F6 16 0178r not cursiveEnabled

411 05D6 E8 FF0B call changeFx

412 05D9 E8 01F6 call setCursive ; перевод символа в курсив и обратно в зависимости от +

413 флага cursiveEnabled

414 05DC EB 1D 90 jmp \_translate\_or\_ignore

415 05DF \_F3:

416 05DF 3C 04 cmp AL, 4 ; F4

417 05E1 75 0A jne \_F4

418 05E3 F6 16 0176r not translateEnabled

419 05E7 E8 FEFA call changeFx

420 05EA EB 0F 90 jmp \_translate\_or\_ignore

421 05ED \_F4:

422 05ED 3C 05 cmp AL, 5 ; F5

423 05EF 75 0A jne \_translate\_or\_ignore

424 05F1 F6 16 016Br not ignoreEnabled

425 05F5 E8 FEEC call changeFx

426 05F8 EB 01 90 jmp \_translate\_or\_ignore

427

428 ;игнорирование и перевод

429 05FB \_translate\_or\_ignore:

430

431 05FB 9C pushf

432 05FC 2E: FF 1E 0199r call dword ptr CS:[old\_int9hOffset] ; вызываем стандартный обработчик прерывания

433 0601 B8 0040 mov AX, 40h ; 40h-сегмент,где хранятся флаги сост-я клавы,кольц. +

434 буфер ввода

435 0604 8E C0 mov ES, AX

436 0606 26: 8B 1E 001C mov BX, ES:[1Ch] ; адрес хвоста

437 060B 4B dec BX ; сместимся назад к последнему

438 060C 4B dec BX ; введённому символу

439 060D 83 FB 1E cmp BX, 1Eh ; не вышли ли мы за пределы буфера?

440 0610 73 03 jae \_go

441 0612 BB 003C mov BX, 3Ch ; хвост вышел за пределы буфера, значит последний введённый +

442 символ

443 ; находится в конце буфера

444

445 0615 \_go:

446 0615 26: 8B 17 mov DX, ES:[BX] ; в DX 0 введённый символ

447 ;включен ли режим блокировки ввода?

448 0618 80 3E 016Br FF cmp ignoreEnabled, true

449 061D 75 21 jne \_check\_translate

450

451 ; да, включен

452 061F BE 0000 mov SI, 0

453 0622 B9 0034 mov CX, ignoredLength ;кол-во игнорируемых символов

454

455 ; проверяем, присутствует ли текущий символ в списке игнорируемых

456 0625 \_check\_ignored:

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 9

tsr.asm

457 0625 3A 94 0137r cmp DL,ignoredChars[SI]

458 0629 74 06 je \_block

459 062B 46 inc SI

460 062C E2 F7 loop \_check\_ignored

461 062E EB 10 90 jmp \_check\_translate

462

463 ; блокируем

464 0631 \_block:

465 0631 26: 89 07 mov ES:[BX], AX

466 0634 33 C0 xor AX, AX

467 0636 8A 84 0103r mov AL, replaceWith[SI]

468 063A 26: 89 07 mov ES:[BX], AX ; замена символа

469 063D EB 23 90 jmp \_quit

470

471 0640 \_check\_translate:

472 ; включен ли режим перевода?

473 0640 80 3E 0176r FF cmp translateEnabled, true

474 0645 75 1B jne \_quit

475

476 ; да, включен

477 0647 BE 0000 mov SI, 0

478 064A B9 0005 mov CX, translateLength ; кол-во символов для перевода

479 ; проверяем, присутствует ли текущий символ в списке для перевода

480 064D \_check\_translate\_loop:

481 064D 3A 94 016Cr cmp DL, translateFrom[SI]

482 0651 74 06 je \_translate

483 0653 46 inc SI

484 0654 E2 F7 loop \_check\_translate\_loop

485 0656 EB 0A 90 jmp \_quit

486

487 ; переводим

488 0659 \_translate:

489 0659 33 C0 xor AX, AX

490 065B 8A 84 0171r mov AL, translateTo[SI]

491 065F 26: 89 07 mov ES:[BX], AX ; замена символа

492

493 0662 \_quit:

494 ; восстанавливаем все регистры

495 0662 1F pop DS

496 0663 07 pop ES

497 0664 5A pop DX

498 0665 59 pop CX

499 0666 5B pop BX

500 0667 58 pop AX

501 0668 5E pop SI

502 0669 CF iret

503 066A new\_int9h endp

504

505 ;=== Обработчик прерывания int 1Ch ===;

506 ;=== Вызывается каждые 55 мс ===;

507 066A new\_int1Ch proc far

508 066A 50 push AX

509 066B 0E push CS

510 066C 1F pop DS

511

512 066D 9C pushf

513 066E 2E: FF 1E 019Dr call dword ptr CS:[old\_int1ChOffset]

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 10

tsr.asm

514

515 0673 80 3E 0177r FF cmp signaturePrintingEnabled, true ; если нажата управляющая клавиша (в данном случае +

516 F1)

517 0678 75 1C jne \_notToPrint

518

519 067A 83 3E 01A7r 5B cmp counter, printDelay\*1000/55 + 1 ; если кол-во "тактов" эквивалентно +

520 %printDelay% секундам

521 067F 74 03 je \_letsPrint

522

523 0681 EB 0E 90 jmp \_dontPrint

524

525 0684 \_letsPrint:

526 0684 F6 16 0177r not signaturePrintingEnabled

527 0688 C7 06 01A7r 0000 mov counter, 0

528 068E E8 0094 call printSignature

529

530 0691 \_dontPrint:

531 0691 83 06 01A7r 01 add counter, 1

532

533 0696 \_notToPrint:

534

535 0696 58 pop AX

536

537 0697 CF iret

538 0698 new\_int1Ch endp

539

540 ;=== Обработчик прерывания int 2Fh ===;

541 ;=== Служит для:

542 ;=== 1) проверки факта присутствия TSR в памяти (при AH=0FFh, AL=0)

543 ;=== будет возвращён AH='i' в случае, если TSR уже загружен

544 ;=== 2) выгрузки TSR из памяти (при AH=0FFh, AL=1)

545 ;===

546 0698 new\_int2Fh proc

547 0698 80 FC FF cmp AH, 0FFh ;наша функция?

548 069B 75 0B jne \_2Fh\_std ;нет - на старый обработчик

549 069D 3C 00 cmp AL, 0 ;подфункция проверки, загружен ли резидент в память?

550 069F 74 0C je \_already\_installed

551 06A1 3C 01 cmp AL, 1 ;подфункция выгрузки из памяти?

552 06A3 74 0B je \_uninstall

553 06A5 EB 01 90 jmp \_2Fh\_std ;нет - на старый обработчик

554

555 06A8 \_2Fh\_std:

556 06A8 2E: FF 2E 01A1r jmp dword ptr CS:[old\_int2FhOffset] ;вызов старого обработчика

557

558 06AD \_already\_installed:

559 06AD B4 69 mov AH, 'i' ;вернём 'i', если резидент загружен в память

560 06AF CF iret

561

562 06B0 \_uninstall:

563 06B0 1E push DS

564 06B1 06 push ES

565 06B2 52 push DX

566 06B3 53 push BX

567

568 06B4 33 DB xor BX, BX

569

570 ; CS = ES, для доступа к переменным

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 11

tsr.asm

571 06B6 0E push CS

572 06B7 07 pop ES

573

574 06B8 B8 2509 mov AX, 2509h

575 06BB 26: 8B 16 0199r mov DX, ES:old\_int9hOffset ; возвращаем вектор прерывания

576 06C0 26: 8E 1E 019Br mov DS, ES:old\_int9hSegment ; на место

577 06C5 CD 21 int 21h

578

579 06C7 B8 251C mov AX, 251Ch

580 06CA 26: 8B 16 019Dr mov DX, ES:old\_int1ChOffset ; возвращаем вектор прерывания

581 06CF 26: 8E 1E 019Fr mov DS, ES:old\_int1ChSegment ; на место

582 06D4 CD 21 int 21h

583

584 06D6 B8 252F mov AX, 252Fh

585 06D9 26: 8B 16 01A1r mov DX, ES:old\_int2FhOffset ; возвращаем вектор прерывания

586 06DE 26: 8E 1E 01A3r mov DS, ES:old\_int2FhSegment ; на место

587 06E3 CD 21 int 21h

588

589 06E5 2E: 8E 06 002C mov ES, CS:2Ch ; загрузим в ES адрес окружения

590 06EA B4 49 mov AH, 49h ; выгрузим из памяти окружение

591 06EC CD 21 int 21h

592 06EE 72 0B jc \_notRemove

593

594 06F0 0E push CS

595 06F1 07 pop ES ;в ES - адрес резидентной программы

596 06F2 B4 49 mov AH, 49h ;выгрузим из памяти резидент

597 06F4 CD 21 int 21h

598 06F6 72 03 jc \_notRemove

599 06F8 EB 15 90 jmp \_unloaded

600

601 06FB \_notRemove: ; не удалось выполнить выгрузку

602 ; вывод сообщения о неудачной выгрузке

603 06FB B4 03 mov AH, 03h ; получаем позицию курсора

604 06FD CD 10 int 10h

605 06FF BD 04BFr lea BP, noRemoveMsg

606 0702 B9 001D mov CX, noRemoveMsg\_length

607 0705 B3 07 mov BL, 0111b

608 0707 B8 1301 mov AX, 1301h

609 070A CD 10 int 10h

610 070C EB 12 90 jmp \_2Fh\_exit

611

612 070F \_unloaded: ; выгрузка прошла успешно

613 ; вывод сообщения об удачной выгрузке

614 070F B4 03 mov AH, 03h ; получаем позицию курсора

615 0711 CD 10 int 10h

616 0713 BD 04AEr lea BP, removedMsg

617 0716 B9 0011 mov CX, removedMsg\_length

618 0719 B3 07 mov BL, 0111b

619 071B B8 1301 mov AX, 1301h

620 071E CD 10 int 10h

621

622 0720 \_2Fh\_exit:

623 0720 5B pop BX

624 0721 5A pop DX

625 0722 07 pop ES

626 0723 1F pop DS

627 0724 CF iret

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 12

tsr.asm

628 0725 new\_int2Fh endp

629

630 ;=== Процедура вывода подписи (ФИО, группа)

631 ;=== Настраивается значениями переменных в начале исходника

632 ;===

633 0725 printSignature proc

634 0725 50 push AX

635 0726 52 push DX

636 0727 51 push CX

637 0728 53 push BX

638 0729 06 push ES

639 072A 54 push SP

640 072B 55 push BP

641 072C 56 push SI

642 072D 57 push DI

643

644 072E 33 C0 xor AX, AX

645 0730 33 DB xor BX, BX

646 0732 33 D2 xor DX, DX

647

648 0734 B4 03 mov AH, 03h ;чтение текущей позиции курсора

649 0736 CD 10 int 10h

650 0738 52 push DX ;помещаем информацию о +

651 положении курсора в стек

652

653 0739 83 3E 01A9r 00 cmp printPos, 0

654 073E 74 0E je \_printTop

655

656 0740 83 3E 01A9r 01 cmp printPos, 1

657 0745 74 0E je \_printCenter

658

659 0747 83 3E 01A9r 02 cmp printPos, 2

660 074C 74 0E je \_printBottom

661

662 ;все числа подобраны на глаз...

663 074E \_printTop:

664 074E B6 00 mov DH, 0

665 0750 B2 0F mov DL, 15

666 0752 EB 0F 90 jmp \_actualPrint

667

668 0755 \_printCenter:

669 0755 B6 09 mov DH, 9

670 0757 B2 0F mov DL, 15

671 0759 EB 08 90 jmp \_actualPrint

672

673 075C \_printBottom:

674 075C B6 13 mov DH, 19

675 075E B2 0F mov DL, 15

676 0760 EB 01 90 jmp \_actualPrint

677

678 0763 \_actualPrint:

679 0763 B4 0F mov AH, 0Fh ;чтение текущего видеорежима. в+

680 BH - текущая страница

681 0765 CD 10 int 10h

682

683 0767 0E push CS

684 0768 07 pop ES ;указываем ES на CS

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 13

tsr.asm

685

686 ;вывод 'верхушки' таблицы

687 0769 52 push DX

688 076A BD 03EBr lea BP, tableTop ;помещаем в BP указатель на +

689 выводимую строку

690 076D B9 0034 mov CX, tableTop\_length ;в CX - длина строки

691 0770 B3 07 mov BL, 0111b ;цвет выводимого текста ref: +

692 http://en.wikipedia.org/wiki/BIOS\_color\_attributes

693 0772 B8 1301 mov AX, 1301h ;AH=13h - номер ф-ии, AL=01h - +

694 курсор перемещается при выводе каждого из символов строки

695 0775 CD 10 int 10h

696 0777 5A pop DX

697 0778 FE C6 inc DH

698

699

700 ;вывод первой линии

701 077A 52 push DX

702 077B BD 01ABr lea BP, signatureLine1

703 077E B9 0034 mov CX, Line1\_length

704 0781 B3 07 mov BL, 0111b

705 0783 B8 1301 mov AX, 1301h

706 0786 CD 10 int 10h

707 0788 5A pop DX

708 0789 FE C6 inc DH

709

710 ;вывод второй линии

711 078B 52 push DX

712 078C BD 01DFr lea BP, signatureLine2

713 078F B9 0034 mov CX, Line2\_length

714 0792 B3 07 mov BL, 0111b

715 0794 B8 1301 mov AX, 1301h

716 0797 CD 10 int 10h

717 0799 5A pop DX

718 079A FE C6 inc DH

719

720 ;вывод третьей линии

721 079C 52 push DX

722 079D BD 0213r lea BP, signatureLine3

723 07A0 B9 0034 mov CX, Line3\_length

724 07A3 B3 07 mov BL, 0111b

725 07A5 B8 1301 mov AX, 1301h

726 07A8 CD 10 int 10h

727 07AA 5A pop DX

728 07AB FE C6 inc DH

729

730 ;вывод 'низа' таблицы

731 07AD 52 push DX

732 07AE BD 041Fr lea BP, tableBottom

733 07B1 B9 0034 mov CX, tableBottom\_length

734 07B4 B3 07 mov BL, 0111b

735 07B6 B8 1301 mov AX, 1301h

736 07B9 CD 10 int 10h

737 07BB 5A pop DX

738 07BC FE C6 inc DH

739

740 07BE 33 DB xor BX, BX

741 07C0 5A pop DX ;восстанавливаем из стека +

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 14

tsr.asm

742 прежнее положение курсора

743 07C1 B4 02 mov AH, 02h ;меняем положение курсора на +

744 первоначальное

745 07C3 CD 10 int 10h

746 07C5 E8 FD1C call changeFx

747

748 07C8 5F pop DI

749 07C9 5E pop SI

750 07CA 5D pop BP

751 07CB 5C pop SP

752 07CC 07 pop ES

753 07CD 5B pop BX

754 07CE 59 pop CX

755 07CF 5A pop DX

756 07D0 58 pop AX

757

758 07D1 C3 ret

759 07D2 printSignature endp

760

761 ;=== Функция, которая в зависимости от флага cursiveEnabled меняет начертание символа с курсива+

762 на обычное и наоброт

763 ;=== Сама смена происходит в процедуре changeFont, а здесь подготавливаются данные

764 07D2 setCursive proc

765 07D2 06 push ES ; сохраняем регистры

766 07D3 50 push AX

767 07D4 0E push CS

768 07D5 07 pop ES

769

770 07D6 80 3E 0178r FF cmp cursiveEnabled, true

771 07DB 75 30 jne \_restoreSymbol

772 ; если флаг равен true, выполняем замену символа на курсивный вариант,

773 ; предварительно сохраняя старый символ в savedSymbol

774

775 07DD E8 004C call saveFont

776 07E0 8A 0E 0188r mov CL, charToCursiveIndex

777 07E4 \_shifTtable:

778 ; мы получаем в BP таблицу всех символов. адрес указывает на символ 0

779 ; поэтому нуэно совершить сдвиг 16\*X - где X - код символа

780 07E4 83 C5 10 add BP, 16

781 07E7 E2 FB loop \_shiftTable

782

783 ; пpи savefont смещается pегистp ES

784 ; поэтомy пpиходится делать такие махинации, чтобы

785 ; записать полyченный элемент в savedSymbol

786 ; swap(ES, DS) и сохранение старого значения DS

787 07E9 1E push DS

788 07EA 58 pop AX

789 07EB 06 push ES

790 07EC 1F pop DS

791 07ED 50 push AX

792 07EE 07 pop ES

793 07EF 50 push AX

794

795 07F0 8B F5 mov SI, BP

796 07F2 BF 0189r lea DI, savedSymbol

797 ; сохpаняем в пеpеменнyю savedSymbol

798 ; таблицy нyжного символа

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 15

tsr.asm

799 07F5 B9 0010 mov CX, 16

800 ; movsb из DS:SI в ES:DI

801 07F8 F3> A4 rep movsb

802 ; исходные позиции сегментов возвpащены

803 07FA 1F pop DS ; восстановление DS

804

805 ; заменим написание символа на кypсив

806 07FB B9 0001 mov CX, 1

807 07FE B6 00 mov DH, 0

808 0800 8A 16 0188r mov DL, charToCursiveIndex

809 0804 BD 0179r lea BP, cursiveSymbol

810 0807 E8 0015 call changeFont

811 080A EB 10 90 jmp \_exitSetCursive

812

813 080D \_restoreSymbol:

814 ; если флаг равен 0, выполняем замену курсивного символа на старый вариант

815

816 080D B9 0001 mov CX, 1

817 0810 B6 00 mov DH, 0

818 0812 8A 16 0188r mov DL, charToCursiveIndex

819 0816 BD 0189r lea bp, savedSymbol

820 0819 E8 0003 call changeFont

821

822 081C \_exitSetCursive:

823 081C 58 pop AX

824 081D 07 pop ES

825 081E C3 ret

826 081F setCursive endp

827

828 ;=== Функция смены начертания символа (курсив/нормальное)

829 ;===

830 ; \*\*\* входные данные

831 ; DL = номер символа для замены

832 ; CX = Кол-во символов заменяемых изображений символов

833 ; (начиная с символа указанного в DX)

834 ; ES:bp = адрес таблицы

835 ;

836 ; \*\*\* описание работы процедуры

837 ; Происходит вызов int 10h (видеосервис)

838 ; с функцией AH = 11h (функции знакогенератора)

839 ; Параметр AL = 0 сообщает, что будет заменено изображение

840 ; символа для текущего шрифта

841 ; В случаях, когда AL = 1 или 2, будет заменено изображение

842 ; только для опредленного шрифта (8x14 и 8x8 соответственно)

843 ; Параметр BH = 0Eh сообщает, что на опредление каждого изображения символа

844 ; расходуется по 14 байт (режим 8x14 бит как раз 14 байт)

845 ; Параметр BL = 0 - блок шрифта для загрузки (от 0 до 4)

846 ;

847 ; \*\*\* результат

848 ; изображение указанного(ых) символа(ов) будет заменено

849 ; на предложенное пользователем.

850 ; Изменению подвергнутся все символы, находящиеся на экране,

851 ; то есть если изображение заменено, старый вариант нигде уже не проявится

852

853 081F changeFont proc

854 081F 50 push AX

855 0820 53 push BX

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 16

tsr.asm

856 0821 B8 1100 mov AX, 1100h

857 0824 BB 1000 mov BX, 1000h

858 0827 CD 10 int 10h

859 0829 58 pop AX

860 082A 5B pop BX

861 082B C3 ret

862 082C changeFont endp

863

864 ;=== Функция сохранения нормального начертания символа

865 ;===

866 ; \*\*\* входные данные

867 ; BH - тип возвращаемой символьной таблицы

868 ; 0 - таблица из int 1fh

869 ; 1 - таблица из int 44h

870 ; 2-5 - таблица из 8x14, 8x8, 8x8 (top), 9x14

871 ; 6 - 8x16

872 ;

873 ; \*\*\* описание работы процедуры

874 ; Происходит вызов int 10h (видеосервис)

875 ; с функцией AH = 11h (функции знакогенератора)

876 ; Параметр AL = 30 - подфункция получения информации о EGA

877 ;

878 ; \*\*\* результат

879 ; в ES:BP находится таблица символов (полная)

880 ; в CX находится байт на символ

881 ; в DL количество экранных строк

882 ; ВАЖНО! Происходит сдвиг регистра ES

883 ; ( ES становится равным C000h )

884

885 082C saveFont proc

886 082C 50 push AX

887 082D 53 push BX

888 082E B8 1130 mov AX, 1130h

889 0831 BB 0600 mov BX, 0600h

890 0834 CD 10 int 10h

891 0836 58 pop AX

892 0837 5B pop BX

893 0838 C3 ret

894 0839 saveFont endp

895

896

897 ;=== Отсюда начинается выполнение основной части программы ===;

898 ;===

899 0839 \_initTSR: ; старт резидента

900 0839 B4 03 mov AH, 03h

901 083B CD 10 int 10h

902 083D 52 push DX

903 083E B4 00 mov AH,00h ; установка видеорежима (83h текст +

904 80x25 16/8 CGA,EGA b800 Comp,RGB,Enhanced), без очистки экрана

905 0840 B0 83 mov AL,83h

906 0842 CD 10 int 10h

907 0844 5A pop DX

908 0845 B4 02 mov AH, 02h

909 0847 CD 10 int 10h

910

911

912 0849 E8 00B3 call commandParamsParser

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 17

tsr.asm

913 084C B8 3509 mov AX,3509h ; получить в ES:BX вектор 09

914 084F CD 21 int 21h ; прерывания

915

916 ;@ === Удаление резидента из памяти ===

917 0851 80 3E 01A5r FF cmp unloadTSR, true

918 0856 74 03 je \_removingOnParameter

919 0858 EB 15 90 jmp \_notRemovingNow

920

921 085B \_removingOnParameter:

922 085B B4 FF mov AH, 0FFh

923 085D B0 00 mov AL, 0

924 085F CD 2F int 2Fh

925 0861 80 FC 69 cmp AH, 'i' ; проверка того, загружена ли уже программа

926 0864 74 7D je \_remove

927 0866 B4 09 mov AH, 09h

928 0868 BA 0490r lea DX, notInstalledMsg

929 086B CD 21 int 21h

930 086D CD 20 int 20h

931

932 086F \_notRemovingNow:

933

934 086F 80 3E 01A6r FF cmp notLoadTSR, true ; если была выведена справка

935 0874 74 0E je \_exit\_tmp ; просто выходим

936

937 0876 B4 FF mov AH, 0FFh

938 0878 B0 00 mov AL, 0

939 087A CD 2F int 2Fh

940 087C 80 FC 69 cmp AH, 'i' ; проверка того, загружена ли уже программа

941 087F 74 6B je \_alreadyInstalled

942

943 0881 EB 04 90 jmp \_tmp

944

945 0884 \_exit\_tmp:

946 0884 EB 77 90 jmp \_exit

947

948 0887 \_tmp:

949 0887 06 push ES

950 0888 A1 002C mov AX, DS:[2Ch] ; psp

951 088B 8E C0 mov ES, AX

952 088D B4 49 mov AH, 49h ; хватит памяти чтоб остаться

953 088F CD 21 int 21h ; резидентом?

954 0891 07 pop ES

955 0892 72 62 jc \_notMem ; не хватило - выходим

956

957 ;== int 09h ==;

958

959 0894 2E: 89 1E 0199r mov word ptr CS:old\_int9hOffset, BX

960 0899 2E: 8C 06 019Br mov word ptr CS:old\_int9hSegment, ES

961 089E B8 2509 mov AX, 2509h ; установим вектор на 09

962 08A1 BA 0586r mov DX, offset new\_int9h ; прерывание

963 08A4 CD 21 int 21h

964

965 ;== int 1Ch ==;

966 08A6 B8 351C mov AX,351Ch ; получить в ES:BX вектор 1C

967 08A9 CD 21 int 21h ; прерывания

968 08AB 2E: 89 1E 019Dr mov word ptr CS:old\_int1ChOffset, BX

969 08B0 2E: 8C 06 019Fr mov word ptr CS:old\_int1ChSegment, ES

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 18

tsr.asm

970 08B5 B8 251C mov AX, 251Ch ; установим вектор на 1C

971 08B8 BA 066Ar mov DX, offset new\_int1Ch ; прерывание

972 08BB CD 21 int 21h

973

974 ;== int 2Fh ==;

975 08BD B8 352F mov AX,352Fh ; получить в ES:BX вектор 1C

976 08C0 CD 21 int 21h ; прерывания

977 08C2 2E: 89 1E 01A1r mov word ptr CS:old\_int2FhOffset, BX

978 08C7 2E: 8C 06 01A3r mov word ptr CS:old\_int2FhSegment, ES

979 08CC B8 252F mov AX, 252Fh ; установим вектор на 2F

980 08CF BA 0698r mov DX, offset new\_int2Fh ; прерывание

981 08D2 CD 21 int 21h

982

983 08D4 E8 FC0D call changeFx

984 08D7 BA 0453r mov DX, offset installedMsg ; выводим что все ок

985 08DA B4 09 mov AH, 9

986 08DC CD 21 int 21h

987 08DE BA 0839r mov DX, offset \_initTSR ; остаемся в памяти резидентом

988 08E1 CD 27 int 27h ; и выходим

989 ; конец основной программы

990 08E3 \_remove: ; выгрузка программы из памяти

991 08E3 B4 FF mov AH, 0FFh

992 08E5 B0 01 mov AL, 1

993 08E7 CD 2F int 2Fh

994 08E9 EB 12 90 jmp \_exit

995 08EC \_alreadyInstalled:

996 08EC B4 09 mov AH, 09h

997 08EE BA 0466r lea DX, alreadyInstalledMsg

998 08F1 CD 21 int 21h

999 08F3 EB 08 90 jmp \_exit

1000 08F6 \_notMem: ; не хватает памяти, чтобы остаться резидентом

1001 08F6 BA 047Cr mov DX, offset noMemMsg

1002 08F9 B4 09 mov AH, 9

1003 08FB CD 21 int 21h

1004 08FD \_exit: ; выход

1005 08FD CD 20 int 20h

1006

1007 ;=== Процедура проверки параметров ком. строки ===;

1008 ;===

1009 08FF commandParamsParser proc

1010 08FF 0E push CS

1011 0900 07 pop ES

1012 0901 C6 06 01A5r 00 mov unloadTSR, 0

1013 0906 C6 06 01A6r 00 mov notLoadTSR, 0

1014

1015 090B BE 0080 mov SI, 80h ;SI=смещение командной строки.

1016 090E AC lodsb ;Получим кол-во символов.

1017 090F 0A C0 or AL, AL ;Если 0 символов введено,

1018 0911 74 3F jz \_exitHelp ;то все в порядке.

1019

1020 0913 \_nextChar:

1021

1022 0913 46 inc SI ;Теперь SI указывает на первый символ +

1023 строки.

1024

1025 0914 80 3C 0D cmp [SI], BYTE ptr 13

1026 0917 74 39 je \_exitHelp

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 19

tsr.asm

1027

1028

1029 0919 AD lodsw ;Получаем два символа

1030 091A 3D 3F2F cmp AX, '?/' ;Это '/?' (данные расположены в +

1031 обратном порядк, т.е. AL:AH вместо AH:AL)

1032 091D 74 08 je \_question

1033 091F 3D 752F cmp AX, 'u/'

1034 0922 74 1A je \_finishTSR

1035

1036 ;cmp AH, '/'

1037 ;je \_errorParam

1038

1039 0924 EB 2C 90 jmp \_exitHelp

1040

1041 0927 \_question:

1042 ; вывод строки помощи

1043 0927 B4 03 mov AH,03

1044 0929 CD 10 int 10h

1045 092B BD 0247r lea BP, helpMsg

1046 092E B9 017F mov CX, helpMsg\_length

1047 0931 B3 07 mov BL, 0111b

1048 0933 B8 1301 mov AX, 1301h

1049 0936 CD 10 int 10h

1050 ; конец вывода строки помощи

1051 0938 F6 16 01A6r not notLoadTSR ;флаг того, что необходимо не загружать резидент

1052 093C EB D5 jmp \_nextChar

1053

1054 ;@ === Удаление резидента из памяти ===

1055

1056 093E \_finishTSR:

1057 ;not unloadTSR ;флаг того, что необходимо выгузить резидент

1058 ;jmp \_nextChar

1059

1060 093E EB 12 90 jmp \_exitHelp

1061

1062 0941 \_errorParam:

1063 ;вывод строки

1064 0941 B4 03 mov AH,03

1065 0943 CD 10 int 10h

1066 0945 BD 03C6r lea BP, CS:errorParamMsg

1067 0948 B9 0025 mov CX, errorParamMsg\_length

1068 094B B3 07 mov BL, 0111b

1069 094D B8 1301 mov AX, 1301h

1070 0950 CD 10 int 10h

1071 ;конец вывода строки

1072 0952 \_exitHelp:

1073 0952 C3 ret

1074 0953 commandParamsParser endp

1075

1076 0953 code ends

1077 end \_start

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 20

Symbol Table

Symbol Name Type Value

??DATE Text "05/15/19"

??FILENAME Text "tsr "

??TIME Text "15:22:32"

??VERSION Number 040A

@CPU Text 0101H

@CURSEG Text CODE

@FILENAME Text TSR

@WORDSIZE Text 2

ALREADYINSTALLEDMSG Byte CODE:0466

CHANGEFONT Near CODE:081F

CHANGEFX Near CODE:04E4

CHARTOCURSIVEINDEX Byte CODE:0188

COMMANDPARAMSPARSER Near CODE:08FF

COUNTER Word CODE:01A7

CURSIVEENABLED Byte CODE:0178

CURSIVESYMBOL Byte CODE:0179

ERRORPARAMMSG Byte CODE:03C6

ERRORPARAMMSG\_LENGTH Number 0025

F1\_TXT Byte CODE:04DC

F2\_TXT Byte CODE:04DE

F3\_TXT Byte CODE:04E0

F4\_TXT Byte CODE:04E2

FX\_LENGTH Number 0002

HELPMSG Byte CODE:0247

HELPMSG\_LENGTH Number 017F

IGNOREDCHARS Byte CODE:0137

IGNOREDLENGTH Number 0034

IGNOREENABLED Byte CODE:016B

INSTALLEDMSG Byte CODE:0453

LINE1\_LENGTH Number 0034

LINE2\_LENGTH Number 0034

LINE3\_LENGTH Number 0034

NEW\_INT1CH Far CODE:066A

NEW\_INT2FH Near CODE:0698

NEW\_INT9H Far CODE:0586

NOMEMMSG Byte CODE:047C

NOREMOVEMSG Byte CODE:04BF

NOREMOVEMSG\_LENGTH Number 001D

NOTINSTALLEDMSG Byte CODE:0490

NOTLOADTSR Byte CODE:01A6

OLD\_INT1CHOFFSET Word CODE:019D

OLD\_INT1CHSEGMENT Word CODE:019F

OLD\_INT2FHOFFSET Word CODE:01A1

OLD\_INT2FHSEGMENT Word CODE:01A3

OLD\_INT9HOFFSET Word CODE:0199

OLD\_INT9HSEGMENT Word CODE:019B

PRINTDELAY Number 0005

PRINTPOS Word CODE:01A9

PRINTSIGNATURE Near CODE:0725

REMOVEDMSG Byte CODE:04AE

REMOVEDMSG\_LENGTH Number 0011

REPLACEWITH Byte CODE:0103

SAVEDSYMBOL Byte CODE:0189

SAVEFONT Near CODE:082C

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 21

Symbol Table

SETCURSIVE Near CODE:07D2

SIGNATURELINE1 Byte CODE:01AB

SIGNATURELINE2 Byte CODE:01DF

SIGNATURELINE3 Byte CODE:0213

SIGNATUREPRINTINGENABLED Byte CODE:0177

TABLEBOTTOM Byte CODE:041F

TABLEBOTTOM\_LENGTH Number 0034

TABLETOP Byte CODE:03EB

TABLETOP\_LENGTH Number 0034

TRANSLATEENABLED Byte CODE:0176

TRANSLATEFROM Byte CODE:016C

TRANSLATELENGTH Number 0005

TRANSLATETO Byte CODE:0171

TRUE Number 00FF

UNLOADTSR Byte CODE:01A5

\_2FH\_EXIT Near CODE:0720

\_2FH\_STD Near CODE:06A8

\_ACTUALPRINT Near CODE:0763

\_ALREADYINSTALLED Near CODE:08EC

\_ALREADY\_INSTALLED Near CODE:06AD

\_BLOCK Near CODE:0631

\_CHECKF1 Near CODE:04F3

\_CHECKF2 Near CODE:0517

\_CHECKF3 Near CODE:0538

\_CHECKF4 Near CODE:0559

\_CHECK\_IGNORED Near CODE:0625

\_CHECK\_TRANSLATE Near CODE:0640

\_CHECK\_TRANSLATE\_LOOP Near CODE:064D

\_DONTPRINT Near CODE:0691

\_ERRORPARAM Near CODE:0941

\_EXIT Near CODE:08FD

\_EXITHELP Near CODE:0952

\_EXITSETCURSIVE Near CODE:081C

\_EXIT\_TMP Near CODE:0884

\_F1 Near CODE:05C0

\_F2 Near CODE:05CE

\_F3 Near CODE:05DF

\_F4 Near CODE:05ED

\_FINISHTSR Near CODE:093E

\_GO Near CODE:0615

\_GREENF1 Near CODE:0510

\_GREENF2 Near CODE:0534

\_GREENF3 Near CODE:0555

\_GREENF4 Near CODE:0576

\_INITTSR Near CODE:0839

\_LETSPRINT Near CODE:0684

\_NEXTCHAR Near CODE:0913

\_NOTMEM Near CODE:08F6

\_NOTREMOVE Near CODE:06FB

\_NOTREMOVINGNOW Near CODE:086F

\_NOTTOPRINT Near CODE:0696

\_OUTFX Near CODE:057A

\_PRINTBOTTOM Near CODE:075C

\_PRINTCENTER Near CODE:0755

\_PRINTTOP Near CODE:074E

\_QUESTION Near CODE:0927

\_QUIT Near CODE:0662

Turbo Assembler Version 4.1 05/15/19 15:22:33 Page 22

Symbol Table

\_REDF1 Near CODE:0509

\_REDF2 Near CODE:052D

\_REDF3 Near CODE:054E

\_REDF4 Near CODE:056F

\_REMOVE Near CODE:08E3

\_REMOVINGONPARAMETER Near CODE:085B

\_RESTORESYMBOL Near CODE:080D

\_SHIFTTABLE Near CODE:07E4

\_START Near CODE:0100

\_TEST\_FX Near CODE:05BE

\_TMP Near CODE:0887

\_TRANSLATE Near CODE:0659

\_TRANSLATE\_OR\_IGNORE Near CODE:05FB

\_UNINSTALL Near CODE:06B0

\_UNLOADED Near CODE:070F

Groups & Segments Bit Size Align Combine Class

CODE 16 0953 Para none CODE