**Московский государственный технический университет**

**им. Н.Э. Баумана**

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

Курсовая работа по курсу «Системное программирование»

Листинг

(вид документа)

писчая бумага

(вид носителя)

26

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-61Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Рябкин А.В. | "\_3\_"\_\_\_\_06\_\_\_\_\_ 2021 г. |

Москва – 2021

Содержание

1.  Файл CWC.lst............................................................................................................................................3

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 1

cwc.asm

1 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

2 ;

3 ; комменатрии, начинающиеся с символа @ - места, где код зависит от варианта

4 ;

5 ; Автор:

6 ; МГТУ им. Н.Э. Баумана, ИУ5Ц-61Б, 2021 г.

7 ; Рябкин А.В.

8 ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

9

10 0000 code segment 'code'

11 assume CS:code, DS:code

12 org 100h

13 0100 \_start:

14

15 0100 E9 06CB jmp \_initTSR ; на начало программы

16

17 ; данные

18 0103 41 42 43 44 45 46 47+ ignoredChars DB +

19 48 49 4A 4B 4C 4D 4E+ 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz' ;@ список игнорируемых символов

20 4F 50 51 52 53 54 55+

21 56 57 58 59 5A 61 62+

22 63 64 65 66 67 68 69+

23 6A 6B 6C 6D 6E 6F 70+

24 71 72 73 74 75 76 77+

25 78 79 7A

26 0137 94 88 91 82 93 80 8F+ replaceWith DB 'ФИСВУАПРШОЛДЬТЩЗЙКЫЕГМЦЧНЯфисвуапршолдьтщзйкыегмцчня'

27 90 98 8E 8B 84 9C 92+

28 99 87 89 8A 9B 85 83+

29 8C 96 97 8D 9F E4 A8+

30 E1 A2 E3 A0 AF E0 E8+

31 AE AB A4 EC E2 E9 A7+

32 A9 AA EB A5 A3 AC E6+

33 E7 AD EF

34 =0068 ignoredLength equ $-ignoredChars +

35 ; длина строки ignoredChars

36 016B 00 ignoreEnabled DB 0 +

37 ; флаг функции игнорирования ввода

38 016C 48 43 4E 45 41 translateFrom DB 'HCNEA' +

39 ;@ символы для замены (ЛМНОП на англ. раскладке)

40 0171 90 91 92 93 94 translateTo DB 'РСТУФ' +

41 ;@ символы на которые будет идти замена

42 =0005 translateLength equ $-translateTo +

43 ; длина строки trasnlateFrom

44 0176 00 translateEnabled DB 0 +

45 ; флаг функции перевода

46

47 0177 00 signaturePrintingEnabled DB 0 +

48 ; флаг функции вывода информации об авторе

49 0178 00 cursiveEnabled DB 0 +

50 ; флаг перевода символа в курсив

51

52 0179 00 cursiveSymbol DB 00000000b +

53 ;@ символ, составленный из единичек (его курсивный вариант)

54 017A 00 DB 00000000b

55 017B 00 DB 00000000b

56 017C 3E DB 00111110b

57 017D 41 DB 01000001b

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 2

cwc.asm

58 017E 40 DB 01000000b

59 017F 40 DB 01000000b

60 0180 40 DB 01000000b

61 0181 40 DB 01000000b

62 0182 40 DB 01000000b

63 0183 40 DB 01000000b

64 0184 40 DB 01000000b

65 0185 00 DB 00000000b

66 0186 00 DB 00000000b

67 0187 00 DB 00000000b

68 0188 00 DB 00000000b

69

70 0189 83 charToCursiveIndex DB 'Г' +

71 ;@ символ для замены

72 018A 10\*(FF) savedSymbol DB 16 dup(0FFh) +

73 ; переменная для хранения старого символа

74

75 =00FF true equ 0FFh +

76 ; константа истинности

77 019A ???? old\_int9hOffset DW ? +

78 ; адрес старого обработчика int 9h

79 019C ???? old\_int9hSegment DW ? +

80 ; сегмент старого обработчика int 9h

81 019E ???? old\_int1ChOffset DW ? +

82 ; адрес старого обработчика int 1Ch

83 01A0 ???? old\_int1ChSegment DW ? +

84 ; сегмент старого обработчика int 1Ch

85 01A2 ???? old\_int2FhOffset DW ? +

86 ; адрес старого обработчика int 2Fh

87 01A4 ???? old\_int2FhSegment DW ? +

88 ; сегмент старого обработчика int 2Fh

89

90 01A6 00 unloadTSR DB 0 +

91 ; 1 - выгрузить резидент

92 01A7 00 notLoadTSR DB 0 +

93 ; 1 - не загружать

94 01A8 0000 counter DW 0

95 =0003 printDelay equ 3 +

96 ;@ задержка перед выводом "подписи" в секундах

97 01AA 0000 printPos DW 0 +

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

99

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

101 (1я строка).

102 01AC B3 20 20 20 20 20 20+ signatureLine1 DB 179, ' Рябкин Александр +

103 20 20 20 20 20 20 90+ ', 179

104 EF A1 AA A8 AD 20 80+

105 AB A5 AA E1 A0 AD A4+

106 E0 20 20 20 20 20 20+

107 20 20 20 20 20 B3

108 =0029 Line1\_length equ $-signatureLine1

109 01D5 B3 20 20 20 20 20 20+ signatureLine2 DB 179, ' ИУ5Ц-61Б +

110 20 20 20 20 20 20 20+ ', 179

111 20 20 20 20 88 93 35+

112 96 2D 36 31 81 20 20+

113 20 20 20 20 20 20 20+

114 20 20 20 20 20 20 20+

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 3

cwc.asm

115 B3

116 =002B Line2\_length equ $-signatureLine2

117 0200 B3 20 20 20 20 20 20+ signatureLine3 DB 179, ' Вариант: 3 +

118 20 20 20 20 20 20 20+ ', 179

119 20 20 82 A0 E0 A8 A0+

120 AD E2 3A 20 33 20 20+

121 20 20 20 20 20 20 20+

122 20 20 20 20 20 B3

123 =0029 Line3\_length equ $-signatureLine3

124 0229 3E 6B 72 2E 63 6F 6D+ helpMsg DB '>cwc.com [/?] [/u]', 10, 13

125 20 5B 2F 3F 5D 20 5B+

126 2F 75 5D 0A 0D

127 023C 20 5B 2F 3F 5D 20 2D+ DB ' [/?] - вывод данной справки', 10, 13

128 20 A2 EB A2 AE A4 20+

129 A4 A0 AD AD AE A9 20+

130 E1 AF E0 A0 A2 AA A8+

131 0A 0D

132 025A 20 5B 2F 75 5D 20 2D+ DB ' [/u] - выгрузка резидента из памяти', 10, 13

133 20 A2 EB A3 E0 E3 A7+

134 AA A0 20 E0 A5 A7 A8+

135 A4 A5 AD E2 A0 20 A8+

136 A7 20 AF A0 AC EF E2+

137 A8 0A 0D

138 0280 20 20 46 34 20 20 2D+ DB ' F4 - вывод ФИО и группы по таймеру в центре экрана', 10, 13

139 20 A2 EB A2 AE A4 20+

140 94 88 8E 20 A8 20 A3+

141 E0 E3 AF AF EB 20 AF+

142 AE 20 E2 A0 A9 AC A5+

143 E0 E3 20 A2 20 E6 A5+

144 AD E2 E0 A5 20 ED AA+

145 E0 A0 AD A0 0A 0D

146 02B7 20 20 46 35 20 20 2D+ DB ' F5 - включение и отключения курсивного вывода русского символа +

147 20 A2 AA AB EE E7 A5+ Г', 10, 13

148 AD A8 A5 20 A8 20 AE+

149 E2 AA AB EE E7 A5 AD+

150 A8 EF 20 AA E3 E0 E1+

151 A8 A2 AD AE A3 AE 20+

152 A2 EB A2 AE A4 A0 20+

153 E0 E3 E1 E1 AA AE A3+

154 AE 20 E1 A8 AC A2 AE+

155 AB A0 20 83 0A 0D

156 02FC 20 20 46 36 20 20 2D+ DB ' F6 - включение и отключение частичной русификации клавиатуры +

157 20 A2 AA AB EE E7 A5+ (HCNEA -> РСТУФ)', 10, 13

158 AD A8 A5 20 A8 20 AE+

159 E2 AA AB EE E7 A5 AD+

160 A8 A5 20 E7 A0 E1 E2+

161 A8 E7 AD AE A9 20 E0+

162 E3 E1 A8 E4 A8 AA A0+

163 E6 A8 A8 20 AA AB A0+

164 A2 A8 A0 E2 E3 E0 EB+

165 28 48 43 4E 45 41 20+

166 2D 3E 20 90 91 92 93+

167 94 29 0A 0D

168 034D 20 20 46 37 20 20 2D+ DB ' F7 - включение и отключение режима замены латинских символов на +

169 20 A2 AA AB EE E7 A5+ русские', 10, 13

170 AD A8 A5 20 A8 20 AE+

171 E2 AA AB EE E7 A5 AD+

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 4

cwc.asm

172 A8 A5 20 E0 A5 A6 A8+

173 AC A0 20 A7 A0 AC A5+

174 AD EB 20 AB A0 E2 A8+

175 AD E1 AA A8 E5 20 E1+

176 A8 AC A2 AE AB AE A2+

177 20 AD A0 20 E0 E3 E1+

178 E1 AA A8 A5 0A 0D

180 =0170 helpMsg\_length equ $-helpMsg

181 0399 8E E8 A8 A1 AA A0 20+ errorParamMsg DB 'Ошибка параметров коммандной +

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

183 E0 AE A2 20 AA AE AC+

184 AC A0 AD A4 AD AE A9+

185 20 E1 E2 E0 AE AA A8+

186 0A 0D

187 =0025 errorParamMsg\_length equ $-errorParamMsg

188

189 03BE DA 27\*(C4) BF tableTop DB 218, Line1\_length-2 dup+

190 (196), 191

191 =0029 tableTop\_length equ $-tableTop

192 03E7 C0 27\*(C4) D9 tableBottom DB 192, Line1\_length-2 dup (196), +

193 217

194 =0029 tableBottom\_length equ $-tableBottom

196 ; сообщения

197 0410 90 A5 A7 A8 A4 A5 AD+ installedMsg DB 'Резидент загружен!$'

198 E2 20 A7 A0 A3 E0 E3+

199 A6 A5 AD 21 24

200 0423 90 A5 A7 A8 A4 A5 AD+ alreadyInstalledMsg DB 'Резидент уже загружен$'

201 E2 20 E3 A6 A5 20 A7+

202 A0 A3 E0 E3 A6 A5 AD+

203 24

204 0439 8D A5 A4 AE E1 E2 A0+ noMemMsg DB 'Недостаточно памяти$'

205 E2 AE E7 AD AE 20 AF+

206 A0 AC EF E2 A8 24

207 044D 8D A5 20 E3 A4 A0 AB+ notInstalledMsg DB 'Не удалось загрузить резидент$'

208 AE E1 EC 20 A7 A0 A3+

209 E0 E3 A7 A8 E2 EC 20+

210 E0 A5 A7 A8 A4 A5 AD+

211 E2 24

212

213 046B 90 A5 A7 A8 A4 A5 AD+ removedMsg DB 'Резидент выгружен'

214 E2 20 A2 EB A3 E0 E3+

215 A6 A5 AD

216 =0011 removedMsg\_length equ $-removedMsg

217

218 047C 8D A5 20 E3 A4 A0 AB+ noRemoveMsg DB 'Не удалось выгрузить резидент'

219 AE E1 EC 20 A2 EB A3+

220 E0 E3 A7 A8 E2 EC 20+

221 E0 A5 A7 A8 A4 A5 AD+

222 E2

223 =001D noRemoveMsg\_length equ $-noRemoveMsg

225 0499 46 34 F3\_txt DB 'F4'

226 049B 46 35 F4\_txt DB 'F5'

227 049D 46 36 F5\_txt DB 'F6'

228 049F 46 37 F6\_txt DB 'F7'

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 5

cwc.asm

229 =0002 fx\_length equ $-F6\_txt

230

231 04A1 changeFx proc

232 04A1 50 push AX

233 04A2 53 push BX

234 04A3 51 push CX

235 04A4 52 push DX

236 04A5 55 push BP

237 04A6 06 push ES

238 04A7 33 DB xor BX, BX

239

240 04A9 B4 03 mov AH, 03h

241 04AB CD 10 int 10h

242 04AD 52 push DX

243

244 04AE 0E push CS

245 04AF 07 pop ES

246

247 04B0 \_checkF3:

248 04B0 BD 0499r lea BP, F3\_txt

249 04B3 B9 0002 mov CX, fx\_length

250 04B6 B7 00 mov BH, 0

251 04B8 B6 00 mov DH, 0

252 04BA B2 4E mov DL, 78

253 04BC B8 1301 mov AX, 1301h

254

255 04BF 80 3E 0177r FF cmp signaturePrintingEnabled, true

256 04C4 74 07 je \_greenF3

257

258 04C6 \_redF3:

259 04C6 B3 4F mov BL, 01001111b ; red

260 04C8 CD 10 int 10h

261 04CA EB 08 90 jmp \_checkF4

262

263 04CD \_greenF3:

264 04CD BD 0499r lea BP, F3\_txt

265 04D0 B3 2F mov BL, 00101111b ; green

266 04D2 CD 10 int 10h

267

268 04D4 \_checkF4:

269 04D4 BD 049Br lea BP, F4\_txt

270 04D7 B9 0002 mov CX, fx\_length

271 04DA B7 00 mov BH, 0

272 04DC B6 01 mov DH, 1

273 04DE B2 4E mov DL, 78

274 04E0 B8 1301 mov AX, 1301h

275

276 04E3 80 3E 0178r FF cmp cursiveEnabled, true

277 04E8 74 07 je \_greenF4

278

279 04EA \_redF4:

280 04EA B3 4F mov BL, 01001111b ; red

281 04EC CD 10 int 10h

282 04EE EB 05 90 jmp \_checkF5

283

284 04F1 \_greenF4:

285 04F1 B3 2F mov BL, 00101111b ; green

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 6

cwc.asm

286 04F3 CD 10 int 10h

287

288 04F5 \_checkF5:

289 04F5 BD 049Dr lea BP, F5\_txt

290 04F8 B9 0002 mov CX, fx\_length

291 04FB B7 00 mov BH, 0

292 04FD B6 02 mov DH, 2

293 04FF B2 4E mov DL, 78

294 0501 B8 1301 mov AX, 1301h

295

296 0504 80 3E 0176r FF cmp translateEnabled, true

297 0509 74 07 je \_greenF5

298

299 050B \_redF5:

300 050B B3 4F mov BL, 01001111b ; red

301 050D CD 10 int 10h

302 050F EB 05 90 jmp \_checkF6

303

304 0512 \_greenF5:

305 0512 B3 2F mov BL, 00101111b ; green

306 0514 CD 10 int 10h

307

308 0516 \_checkF6:

309 0516 BD 049Fr lea BP, F6\_txt

310 0519 B9 0002 mov CX, fx\_length

311 051C B7 00 mov BH, 0

312 051E B6 03 mov DH, 3

313 0520 B2 4E mov DL, 78

314 0522 B8 1301 mov AX, 1301h

315

316 0525 80 3E 016Br FF cmp ignoreEnabled, true

317 052A 74 07 je \_greenF6

318

319 052C \_redF6:

320 052C B3 4F mov BL, 01001111b ; red

321 052E CD 10 int 10h

322 0530 EB 05 90 jmp \_outFx

323

324 0533 \_greenF6:

325 0533 B3 2F mov BL, 00101111b ; green

326 0535 CD 10 int 10h

327

328 0537 \_outFx:

329 0537 5A pop DX

330 0538 B4 02 mov AH, 02h

331 053A CD 10 int 10h

332

333 053C 07 pop ES

334 053D 5D pop BP

335 053E 5A pop DX

336 053F 59 pop CX

337 0540 5B pop BX

338 0541 58 pop AX

339 0542 C3 ret

340 0543 changeFx endp

341

342 ;новый обработчик

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 7

cwc.asm

343 0543 new\_int9h proc far

344 ; сохраняем значения всех, изменяемых регистров в стэке

345 0543 56 push SI

346 0544 50 push AX

347 0545 53 push BX

348 0546 51 push CX

349 0547 52 push DX

350 0548 06 push ES

351 0549 1E push DS

352 ; синхронизируем CS и DS

353 054A 0E push CS

354 054B 1F pop DS

355

356 054C B8 0040 mov AX, 40h ; 40h-сегмент,где хранятся флаги сост-я клавиатуры, кольц. +

357 буфер ввода

358 054F 8E C0 mov ES, AX

359 0551 E4 60 in AL, 60h ; записываем в AL скан-код нажатой клавиши

362 ;проверка F4-F7

363 0553 \_test\_Fx:

364 0553 2C 3A sub AL, 58 ; в AL теперь номер функциональной клавиши

365 0555 \_F3:

366 0555 3C 04 cmp AL, 4 ; F4

367 0557 75 0A jne \_F4

368 0559 F6 16 0177r not signaturePrintingEnabled

369 055D E8 FF41 call changeFx

370 0560 EB 2E 90 jmp \_translate\_or\_ignore

371 0563 \_F4:

372 0563 3C 05 cmp AL, 5 ; F5

373 0565 75 0D jne \_F5

374 0567 F6 16 0178r not cursiveEnabled

375 056B E8 FF33 call changeFx

376 056E E8 01F6 call setCursive ; перевод символа в курсив и обратно в зависимости от +

377 флага cursiveEnabled

378 0571 EB 1D 90 jmp \_translate\_or\_ignore

379 0574 \_F5:

380 0574 3C 06 cmp AL, 6 ; F6

381 0576 75 0A jne \_F6

382 0578 F6 16 0176r not translateEnabled

383 057C E8 FF22 call changeFx

384 057F EB 0F 90 jmp \_translate\_or\_ignore

385 0582 \_F6:

386 0582 3C 07 cmp AL, 7 ; F7

387 0584 75 0A jne \_translate\_or\_ignore

388 0586 F6 16 016Br not ignoreEnabled

389 058A E8 FF14 call changeFx

390 058D EB 01 90 jmp \_translate\_or\_ignore

391

392 ;игнорирование и перевод

393 0590 \_translate\_or\_ignore:

394

395 0590 9C pushf

396 0591 2E: FF 1E 019Ar call dword ptr CS:[old\_int9hOffset] ; вызываем стандартный обработчик прерывания

397 0596 B8 0040 mov AX, 40h ; 40h-сегмент,где хранятся флаги сост-я клавы,кольц. +

398 буфер ввода

399 0599 8E C0 mov ES, AX

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 8

cwc.asm

400 059B 26: 8B 1E 001C mov BX, ES:[1Ch] ; адрес хвоста

401 05A0 4B dec BX ; сместимся назад к последнему

402 05A1 4B dec BX ; введённому символу

403 05A2 83 FB 1E cmp BX, 1Eh ; не вышли ли мы за пределы буфера?

404 05A5 73 03 jae \_go

405 05A7 BB 003C mov BX, 3Ch ; хвост вышел за пределы буфера, значит последний введённый +

406 символ

407 ; находится в конце буфера

408

409 05AA \_go:

410 05AA 26: 8B 17 mov DX, ES:[BX] ; в DX 0 введённый символ

411 ;включен ли режим блокировки ввода?

412 05AD 80 3E 016Br FF cmp ignoreEnabled, true

413 05B2 75 21 jne \_check\_translate

414

415 ; да, включен

416 05B4 BE 0000 mov SI, 0

417 05B7 B9 0068 mov CX, ignoredLength ;кол-во игнорируемых символов

418

419 ; проверяем, присутствует ли текущий символ в списке игнорируемых

420 05BA \_check\_ignored:

421 05BA 3A 94 0103r cmp DL,ignoredChars[SI]

422 05BE 74 06 je \_block

423 05C0 46 inc SI

424 05C1 E2 F7 loop \_check\_ignored

425 05C3 EB 10 90 jmp \_check\_translate

426

427 ; блокируем

428 05C6 \_block:

429 05C6 26: 89 07 mov ES:[BX], AX

430 05C9 33 C0 xor AX, AX

431 05CB 8A 84 0137r mov AL, replaceWith[SI]

432 05CF 26: 89 07 mov ES:[BX], AX ; замена символа

433 05D2 EB 23 90 jmp \_quit

434

435 05D5 \_check\_translate:

436 ; включен ли режим перевода?

437 05D5 80 3E 0176r FF cmp translateEnabled, true

438 05DA 75 1B jne \_quit

439

440 ; да, включен

441 05DC BE 0000 mov SI, 0

442 05DF B9 0005 mov CX, translateLength ; кол-во символов для перевода

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

444 05E2 \_check\_translate\_loop:

445 05E2 3A 94 016Cr cmp DL, translateFrom[SI]

446 05E6 74 06 je \_translate

447 05E8 46 inc SI

448 05E9 E2 F7 loop \_check\_translate\_loop

449 05EB EB 0A 90 jmp \_quit

450

451 ; переводим

452 05EE \_translate:

453 05EE 33 C0 xor AX, AX

454 05F0 8A 84 0171r mov AL, translateTo[SI]

455 05F4 26: 89 07 mov ES:[BX], AX ; замена символа

456

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 9

cwc.asm

457 05F7 \_quit:

458 ; восстанавливаем все регистры

459 05F7 1F pop DS

460 05F8 07 pop ES

461 05F9 5A pop DX

462 05FA 59 pop CX

463 05FB 5B pop BX

464 05FC 58 pop AX

465 05FD 5E pop SI

466 05FE CF iret

467 05FF new\_int9h endp

468

469 ;=== Обработчик прерывания int 1Ch ===;

470 ;=== Вызывается каждые 55 мс ===;

471 05FF new\_int1Ch proc far

472 05FF 50 push AX

473 0600 0E push CS

474 0601 1F pop DS

475

476 0602 9C pushf

477 0603 2E: FF 1E 019Er call dword ptr CS:[old\_int1ChOffset]

478

479 0608 80 3E 0177r FF cmp signaturePrintingEnabled, true ; если нажата управляющая клавиша (в данном случае +

480 F3)

481 060D 75 1C jne \_notToPrint

482

483 060F 83 3E 01A8r 37 cmp counter, printDelay\*1000/55 + 1 ; если кол-во "тактов" эквивалентно +

484 %printDelay% секундам

485 0614 74 03 je \_letsPrint

486

487 0616 EB 0E 90 jmp \_dontPrint

488

489 0619 \_letsPrint:

490 0619 F6 16 0177r not signaturePrintingEnabled

491 061D C7 06 01A8r 0000 mov counter, 0

492 0623 E8 0094 call printSignature

493

494 0626 \_dontPrint:

495 0626 83 06 01A8r 01 add counter, 1

496

497 062B \_notToPrint:

498

499 062B 58 pop AX

500

501 062C CF iret

502 062D new\_int1Ch endp

503

504 ;=== Обработчик прерывания int 2Fh ===;

505 ;=== Служит для:

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

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

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

509 ;===

510 062D new\_int2Fh proc

511 062D 80 FC FF cmp AH, 0FFh ;наша функция?

512 0630 75 0B jne \_2Fh\_std ;нет - на старый обработчик

513 0632 3C 00 cmp AL, 0 ;подфункция проверки, загружен ли резидент в память?

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 10

cwc.asm

514 0634 74 0C je \_already\_installed

515 0636 3C 01 cmp AL, 1 ;подфункция выгрузки из памяти?

516 0638 74 0B je \_uninstall

517 063A EB 01 90 jmp \_2Fh\_std ;нет - на старый обработчик

518

519 063D \_2Fh\_std:

520 063D 2E: FF 2E 01A2r jmp dword ptr CS:[old\_int2FhOffset] ;вызов старого обработчика

521

522 0642 \_already\_installed:

523 0642 B4 69 mov AH, 'i' ;вернём 'i', если резидент загружен в память

524 0644 CF iret

525

526 0645 \_uninstall:

527 0645 1E push DS

528 0646 06 push ES

529 0647 52 push DX

530 0648 53 push BX

531

532 0649 33 DB xor BX, BX

533

534 ; CS = ES, для доступа к переменным

535 064B 0E push CS

536 064C 07 pop ES

537

538 064D B8 2509 mov AX, 2509h

539 0650 26: 8B 16 019Ar mov DX, ES:old\_int9hOffset ; возвращаем вектор прерывания

540 0655 26: 8E 1E 019Cr mov DS, ES:old\_int9hSegment ; на место

541 065A CD 21 int 21h

542

543 065C B8 251C mov AX, 251Ch

544 065F 26: 8B 16 019Er mov DX, ES:old\_int1ChOffset ; возвращаем вектор прерывания

545 0664 26: 8E 1E 01A0r mov DS, ES:old\_int1ChSegment ; на место

546 0669 CD 21 int 21h

547

548 066B B8 252F mov AX, 252Fh

549 066E 26: 8B 16 01A2r mov DX, ES:old\_int2FhOffset ; возвращаем вектор прерывания

550 0673 26: 8E 1E 01A4r mov DS, ES:old\_int2FhSegment ; на место

551 0678 CD 21 int 21h

552

553 067A 2E: 8E 06 002C mov ES, CS:2Ch ; загрузим в ES адрес окружения

554 067F B4 49 mov AH, 49h ; выгрузим из памяти окружение

555 0681 CD 21 int 21h

556 0683 72 0B jc \_notRemove

557

558 0685 0E push CS

559 0686 07 pop ES ;в ES - адрес резидентной программы

560 0687 B4 49 mov AH, 49h ;выгрузим из памяти резидент

561 0689 CD 21 int 21h

562 068B 72 03 jc \_notRemove

563 068D EB 15 90 jmp \_unloaded

564

565 0690 \_notRemove: ; не удалось выполнить выгрузку

566 ; вывод сообщения о неудачной выгрузке

567 0690 B4 03 mov AH, 03h ; получаем позицию курсора

568 0692 CD 10 int 10h

569 0694 BD 047Cr lea BP, noRemoveMsg

570 0697 B9 001D mov CX, noRemoveMsg\_length

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 11

cwc.asm

571 069A B3 07 mov BL, 0111b

572 069C B8 1301 mov AX, 1301h

573 069F CD 10 int 10h

574 06A1 EB 12 90 jmp \_2Fh\_exit

575

576 06A4 \_unloaded: ; выгрузка прошла успешно

577 ; вывод сообщения об удачной выгрузке

578 06A4 B4 03 mov AH, 03h ; получаем позицию курсора

579 06A6 CD 10 int 10h

580 06A8 BD 046Br lea BP, removedMsg

581 06AB B9 0011 mov CX, removedMsg\_length

582 06AE B3 07 mov BL, 0111b

583 06B0 B8 1301 mov AX, 1301h

584 06B3 CD 10 int 10h

585

586 06B5 \_2Fh\_exit:

587 06B5 5B pop BX

588 06B6 5A pop DX

589 06B7 07 pop ES

590 06B8 1F pop DS

591 06B9 CF iret

592 06BA new\_int2Fh endp

593

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

595 ;=== Настраивается значениями переменных в начале исходника

596 ;===

597 06BA printSignature proc

598 06BA 50 push AX

599 06BB 52 push DX

600 06BC 51 push CX

601 06BD 53 push BX

602 06BE 06 push ES

603 06BF 54 push SP

604 06C0 55 push BP

605 06C1 56 push SI

606 06C2 57 push DI

607

608 06C3 33 C0 xor AX, AX

609 06C5 33 DB xor BX, BX

610 06C7 33 D2 xor DX, DX

611

612 06C9 B4 03 mov AH, 03h ;чтение текущей позиции курсора

613 06CB CD 10 int 10h

614 06CD 52 push DX ;помещаем информацию о +

615 положении курсора в стек

616

617 06CE 83 3E 01AAr 00 cmp printPos, 0

618 06D3 74 0E je \_printTop

619

620 06D5 83 3E 01AAr 01 cmp printPos, 1

621 06DA 74 0E je \_printCenter

622

623 06DC 83 3E 01AAr 02 cmp printPos, 2

624 06E1 74 0E je \_printBottom

625

626 ;все числа подобраны на глаз...

627 06E3 \_printTop:

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 12

cwc.asm

628 06E3 B6 00 mov DH, 0

629 06E5 B2 0F mov DL, 15

630 06E7 EB 0F 90 jmp \_actualPrint

632 06EA \_printCenter:

633 06EA B6 09 mov DH, 9

634 06EC B2 0F mov DL, 15

635 06EE EB 08 90 jmp \_actualPrint

636

637 06F1 \_printBottom:

638 06F1 B6 13 mov DH, 19

639 06F3 B2 0F mov DL, 15

640 06F5 EB 01 90 jmp \_actualPrint

641

642 06F8 \_actualPrint:

643 06F8 B4 0F mov AH, 0Fh ;чтение текущего видеорежима. в+

644 BH - текущая страница

645 06FA CD 10 int 10h

646

647 06FC 0E push CS

648 06FD 07 pop ES ;указываем ES на CS

650 ;вывод 'верхушки' таблицы

651 06FE 52 push DX

652 06FF BD 03BEr lea BP, tableTop ;помещаем в BP указатель на +

653 выводимую строку

654 0702 B9 0029 mov CX, tableTop\_length ;в CX - длина строки

655 0705 B3 07 mov BL, 0111b ;цвет выводимого текста ref: +

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

657 0707 B8 1301 mov AX, 1301h ;AH=13h - номер ф-ии, AL=01h - +

658 курсор перемещается при выводе каждого из символов строки

659 070A CD 10 int 10h

660 070C 5A pop DX

661 070D FE C6 inc DH

662

663

664 ;вывод первой линии

665 070F 52 push DX

666 0710 BD 01ACr lea BP, signatureLine1

667 0713 B9 0029 mov CX, Line1\_length

668 0716 B3 07 mov BL, 0111b

669 0718 B8 1301 mov AX, 1301h

670 071B CD 10 int 10h

671 071D 5A pop DX

672 071E FE C6 inc DH

673

674 ;вывод второй линии

675 0720 52 push DX

676 0721 BD 01D5r lea BP, signatureLine2

677 0724 B9 002B mov CX, Line2\_length

678 0727 B3 07 mov BL, 0111b

679 0729 B8 1301 mov AX, 1301h

680 072C CD 10 int 10h

681 072E 5A pop DX

682 072F FE C6 inc DH

683

684 ;вывод третьей линии

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 13

cwc.asm

685 0731 52 push DX

686 0732 BD 0200r lea BP, signatureLine3

687 0735 B9 0029 mov CX, Line3\_length

688 0738 B3 07 mov BL, 0111b

689 073A B8 1301 mov AX, 1301h

690 073D CD 10 int 10h

691 073F 5A pop DX

692 0740 FE C6 inc DH

694 ;вывод 'низа' таблицы

695 0742 52 push DX

696 0743 BD 03E7r lea BP, tableBottom

697 0746 B9 0029 mov CX, tableBottom\_length

698 0749 B3 07 mov BL, 0111b

699 074B B8 1301 mov AX, 1301h

700 074E CD 10 int 10h

701 0750 5A pop DX

702 0751 FE C6 inc DH

703

704 0753 33 DB xor BX, BX

705 0755 5A pop DX ;восстанавливаем из стека +

706 прежнее положение курсора

707 0756 B4 02 mov AH, 02h ;меняем положение курсора на +

708 первоначальное

709 0758 CD 10 int 10h

710 075A E8 FD44 call changeFx

711

712 075D 5F pop DI

713 075E 5E pop SI

714 075F 5D pop BP

715 0760 5C pop SP

716 0761 07 pop ES

717 0762 5B pop BX

718 0763 59 pop CX

719 0764 5A pop DX

720 0765 58 pop AX

721

722 0766 C3 ret

723 0767 printSignature endp

724

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

726 на обычное и наоброт

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

728 0767 setCursive proc

729 0767 06 push ES ; сохраняем регистры

730 0768 50 push AX

731 0769 0E push CS

732 076A 07 pop ES

733

734 076B 80 3E 0178r FF cmp cursiveEnabled, true

735 0770 75 30 jne \_restoreSymbol

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

737 ; предварительно сохраняя старый символ в savedSymbol

738

739 0772 E8 004C call saveFont

740 0775 8A 0E 0189r mov CL, charToCursiveIndex

741 0779 \_shifTtable:

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 14

cwc.asm

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

743 ; поэтому нужно совершить сдвиг 16\*X - где X - код символа

744 0779 83 C5 10 add BP, 16

745 077C E2 FB loop \_shiftTable

746

747 ; пpи savefont смещается pегистp ES

748 ; поэтомy пpиходится делать такие махинации, чтобы

749 ; записать полyченный элемент в savedSymbol

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

751 077E 1E push DS

752 077F 58 pop AX

753 0780 06 push ES

754 0781 1F pop DS

755 0782 50 push AX

756 0783 07 pop ES

757 0784 50 push AX

758

759 0785 8B F5 mov SI, BP

760 0787 BF 018Ar lea DI, savedSymbol

761 ; сохpаняем в пеpеменнyю savedSymbol

762 ; таблицy нyжного символа

763 078A B9 0010 mov CX, 16

764 ; movsb из DS:SI в ES:DI

765 078D F3> A4 rep movsb

766 ; исходные позиции сегментов возвpащены

767 078F 1F pop DS ; восстановление DS

768

769 ; заменим написание символа на кypсив

770 0790 B9 0001 mov CX, 1

771 0793 B6 00 mov DH, 0

772 0795 8A 16 0189r mov DL, charToCursiveIndex

773 0799 BD 0179r lea BP, cursiveSymbol

774 079C E8 0015 call changeFont

775 079F EB 10 90 jmp \_exitSetCursive

776

777 07A2 \_restoreSymbol:

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

779

780 07A2 B9 0001 mov CX, 1

781 07A5 B6 00 mov DH, 0

782 07A7 8A 16 0189r mov DL, charToCursiveIndex

783 07AB BD 018Ar lea bp, savedSymbol

784 07AE E8 0003 call changeFont

785

786 07B1 \_exitSetCursive:

787 07B1 58 pop AX

788 07B2 07 pop ES

789 07B3 C3 ret

790 07B4 setCursive endp

791

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

793 ;===

794 07B4 changeFont proc

795 07B4 50 push AX

796 07B5 53 push BX

797 07B6 B8 1100 mov AX, 1100h

798 07B9 BB 1000 mov BX, 1000h

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 15

cwc.asm

799 07BC CD 10 int 10h

800 07BE 58 pop AX

801 07BF 5B pop BX

802 07C0 C3 ret

803 07C1 changeFont endp

804

805 ;=== Функция сохранения нормального начертания символа

806 ;===

807 07C1 saveFont proc

808 07C1 50 push AX

809 07C2 53 push BX

810 07C3 B8 1130 mov AX, 1130h

811 07C6 BB 0600 mov BX, 0600h

812 07C9 CD 10 int 10h

813 07CB 58 pop AX

814 07CC 5B pop BX

815 07CD C3 ret

816 07CE saveFont endp

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

820 ;===

821 07CE \_initTSR: ; старт резидента

822 07CE B4 03 mov AH, 03h

823 07D0 CD 10 int 10h

824 07D2 52 push DX

825 07D3 B4 00 mov AH,00h ; установка видеорежима (83h текст +

826 80x25 16/8 CGA,EGA b800 Comp,RGB,Enhanced), без очистки экрана

827 07D5 B0 83 mov AL,83h

828 07D7 CD 10 int 10h

829 07D9 5A pop DX

830 07DA B4 02 mov AH, 02h

831 07DC CD 10 int 10h

832

833

834 07DE E8 00B3 call commandParamsParser

835 07E1 B8 3509 mov AX,3509h ; получить в ES:BX вектор 09

836 07E4 CD 21 int 21h ; прерывания

837

838 ;@ === Удаление резидента из памяти ===

839 ;@ Если по варианту необходимо выгружать резидент по повторному запуску приложений,

840 ;@ нужно закомментировать следующие 3 строки, а также

841 ;@ содержимое метки \_finishTSR ф-ии commandParamsParser, но не саму метку!

842 07E6 80 3E 01A6r FF cmp unloadTSR, true

843 07EB 74 03 je \_removingOnParameter

844 07ED EB 15 90 jmp \_notRemovingNow

845

846 07F0 \_removingOnParameter:

847 07F0 B4 FF mov AH, 0FFh

848 07F2 B0 00 mov AL, 0

849 07F4 CD 2F int 2Fh

850 07F6 80 FC 69 cmp AH, 'i' ; проверка того, загружена ли уже программа

851 07F9 74 7D je \_remove

852 07FB B4 09 mov AH, 09h ;@ для выгрузки резидента по повторному+

853 запуску закомментировать эту строку

854 07FD BA 044Dr lea DX, notInstalledMsg ;@ для выгрузки резидента по повторному запуску +

855 закомментировать эту строку

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 16

cwc.asm

856 0800 CD 21 int 21h ;@ для выгрузки резидента по повторному+

857 запуску закомментировать эту строку

858 0802 CD 20 int 20h ;@ для выгрузки резидента по повторному+

859 запуску закомментировать эту строку

860

861 0804 \_notRemovingNow:

862

863 0804 80 3E 01A7r FF cmp notLoadTSR, true ; если была выведена справка

864 0809 74 0E je \_exit\_tmp ; просто выходим

865

866 ;@ Если по варианту необходимо выгружать резидент по повторному запуску, то +

867 комментируем 5 строк ниже

868 ;@ если необходимо выгружать по параметру коммандной строки, то оставляем их

869 080B B4 FF mov AH, 0FFh

870 080D B0 00 mov AL, 0

871 080F CD 2F int 2Fh

872 0811 80 FC 69 cmp AH, 'i' ; проверка того, загружена ли уже программа

873 0814 74 6B je \_alreadyInstalled

874

875 0816 EB 04 90 jmp \_tmp

876

877 0819 \_exit\_tmp:

878 0819 EB 77 90 jmp \_exit

879

880 081C \_tmp:

881 081C 06 push ES

882 081D A1 002C mov AX, DS:[2Ch] ; psp

883 0820 8E C0 mov ES, AX

884 0822 B4 49 mov AH, 49h ; хватит памяти чтоб остаться

885 0824 CD 21 int 21h ; резидентом?

886 0826 07 pop ES

887 0827 72 62 jc \_notMem ; не хватило - выходим

888

889 ;== int 09h ==;

891 0829 2E: 89 1E 019Ar mov word ptr CS:old\_int9hOffset, BX

892 082E 2E: 8C 06 019Cr mov word ptr CS:old\_int9hSegment, ES

893 0833 B8 2509 mov AX, 2509h ; установим вектор на 09

894 0836 BA 0543r mov DX, offset new\_int9h ; прерывание

895 0839 CD 21 int 21h

896

897 ;== int 1Ch ==;

898 083B B8 351C mov AX,351Ch ; получить в ES:BX вектор 1C

899 083E CD 21 int 21h ; прерывания

900 0840 2E: 89 1E 019Er mov word ptr CS:old\_int1ChOffset, BX

901 0845 2E: 8C 06 01A0r mov word ptr CS:old\_int1ChSegment, ES

902 084A B8 251C mov AX, 251Ch ; установим вектор на 1C

903 084D BA 05FFr mov DX, offset new\_int1Ch ; прерывание

904 0850 CD 21 int 21h

905

906 ;== int 2Fh ==;

907 0852 B8 352F mov AX,352Fh ; получить в ES:BX вектор 1C

908 0855 CD 21 int 21h ; прерывания

909 0857 2E: 89 1E 01A2r mov word ptr CS:old\_int2FhOffset, BX

910 085C 2E: 8C 06 01A4r mov word ptr CS:old\_int2FhSegment, ES

911 0861 B8 252F mov AX, 252Fh ; установим вектор на 2F

912 0864 BA 062Dr mov DX, offset new\_int2Fh ; прерывание

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 17

cwc.asm

913 0867 CD 21 int 21h

914

915 0869 E8 FC35 call changeFx

916 086C BA 0410r mov DX, offset installedMsg ; выводим что все ок

917 086F B4 09 mov AH, 9

918 0871 CD 21 int 21h

919 0873 BA 07CEr mov DX, offset \_initTSR ; остаемся в памяти резидентом

920 0876 CD 27 int 27h ; и выходим

921 ; конец основной программы

922 0878 \_remove: ; выгрузка программы из памяти

923 0878 B4 FF mov AH, 0FFh

924 087A B0 01 mov AL, 1

925 087C CD 2F int 2Fh

926 087E EB 12 90 jmp \_exit

927 0881 \_alreadyInstalled:

928 0881 B4 09 mov AH, 09h

929 0883 BA 0423r lea DX, alreadyInstalledMsg

930 0886 CD 21 int 21h

931 0888 EB 08 90 jmp \_exit

932 088B \_notMem: ; не хватает памяти, чтобы остаться резидентом

933 088B BA 0439r mov DX, offset noMemMsg

934 088E B4 09 mov AH, 9

935 0890 CD 21 int 21h

936 0892 \_exit: ; выход

937 0892 CD 20 int 20h

938

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

940 ;===

941 0894 commandParamsParser proc

942 0894 0E push CS

943 0895 07 pop ES

944 0896 C6 06 01A6r 00 mov unloadTSR, 0

945 089B C6 06 01A7r 00 mov notLoadTSR, 0

946

947 08A0 BE 0080 mov SI, 80h ;SI=смещение командной строки.

948 08A3 AC lodsb ;Получим кол-во символов.

949 08A4 0A C0 or AL, AL ;Если 0 символов введено,

950 08A6 74 45 jz \_exitHelp ;то все в порядке.

951

952 08A8 \_nextChar:

953

954 08A8 46 inc SI ;Теперь SI указывает на первый символ +

955 строки.

956

957 08A9 80 3C 0D cmp [SI], BYTE ptr 13

958 08AC 74 3F je \_exitHelp

959

960

961 08AE AD lodsw ;Получаем два символа

962 08AF 3D 3F2F cmp AX, '?/' ;Это '/?' (данные расположены в +

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

964 08B2 74 08 je \_question

965 08B4 3D 752F cmp AX, 'u/'

966 08B7 74 1A je \_finishTSR

967

968 ;cmp AH, '/'

969 ;je \_errorParam

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 18

cwc.asm

970

971 08B9 EB 32 90 jmp \_exitHelp

972

973 08BC \_question:

974 ; вывод строки помощи

975 08BC B4 03 mov AH,03

976 08BE CD 10 int 10h

977 08C0 BD 0229r lea BP, helpMsg

978 08C3 B9 0170 mov CX, helpMsg\_length

979 08C6 B3 07 mov BL, 0111b

980 08C8 B8 1301 mov AX, 1301h

981 08CB CD 10 int 10h

982 ; конец вывода строки помощи

983 08CD F6 16 01A7r not notLoadTSR ;флаг того, что необходимо не загружать резидент

984 08D1 EB D5 jmp \_nextChar

985

986 ;@ === Удаление резидента из памяти ===

987 ;@ Если по варианту необходимо выгружать резидент по параметру '/u' коммандной строки,

988 ;@ нужно использовать следующий код, в остальных случаях необходимо закомменитровать

989 ;@ этот код, кроме названия метки! (по желанию можно избавиться и от метки, но +

990 аккуратно просмотреть использование)

991 08D3 \_finishTSR:

992 08D3 F6 16 01A6r not unloadTSR ;флаг того, что необходимо выгузить резидент

993 08D7 EB CF jmp \_nextChar

994

995 08D9 EB 12 90 jmp \_exitHelp

996

997 08DC \_errorParam:

998 ;вывод строки

999 08DC B4 03 mov AH,03

1000 08DE CD 10 int 10h

1001 08E0 BD 0399r lea BP, CS:errorParamMsg

1002 08E3 B9 0025 mov CX, errorParamMsg\_length

1003 08E6 B3 07 mov BL, 0111b

1004 08E8 B8 1301 mov AX, 1301h

1005 08EB CD 10 int 10h

1006 ;конец вывода строки

1007 08ED \_exitHelp:

1008 08ED C3 ret

1009 08EE commandParamsParser endp

1010

1011 08EE code ends

1012 end \_start

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 19

Symbol Table

Symbol Name Type Value Cref (defined at #)

??DATE Text "06/03/21"

??FILENAME Text "cwc "

??TIME Text "15:03:00"

??VERSION Number 030A

@CPU Text 0101H

@CURSEG Text CODE #10

@FILENAME Text CWC

@WORDSIZE Text 2 #10

ALREADYINSTALLEDMSG Byte CODE:0423 #200 929

CHANGEFONT Near CODE:07B4 774 784 #794

CHANGEFX Near CODE:04A1 #231 369 375 383 389 710 915

CHARTOCURSIVEINDEX Byte CODE:0189 #70 740 772 782

COMMANDPARAMSPARSER Near CODE:0894 834 #941

COUNTER Word CODE:01A8 #94 483 491 495

CURSIVEENABLED Byte CODE:0178 #49 276 374 734

CURSIVESYMBOL Byte CODE:0179 #52 773

ERRORPARAMMSG Byte CODE:0399 #181 187 1001

ERRORPARAMMSG\_LENGTH Number 0025 #187 1002

F3\_TXT Byte CODE:0499 #225 248 264

F4\_TXT Byte CODE:049B #226 269

F5\_TXT Byte CODE:049D #227 289

F6\_TXT Byte CODE:049F #228 229 309

FX\_LENGTH Number 0002 #229 249 270 290 310

HELPMSG Byte CODE:0229 #124 180 977

HELPMSG\_LENGTH Number 0170 #180 978

IGNOREDCHARS Byte CODE:0103 #18 34 421

IGNOREDLENGTH Number 0068 #34 417

IGNOREENABLED Byte CODE:016B #36 316 388 412

INSTALLEDMSG Byte CODE:0410 #197 916

LINE1\_LENGTH Number 0029 #108 189 192 667

LINE2\_LENGTH Number 002B #116 677

LINE3\_LENGTH Number 0029 #123 687

NEW\_INT1CH Far CODE:05FF #471 903

NEW\_INT2FH Near CODE:062D #510 912

NEW\_INT9H Far CODE:0543 #343 894

NOMEMMSG Byte CODE:0439 #204 933

NOREMOVEMSG Byte CODE:047C #218 223 569

NOREMOVEMSG\_LENGTH Number 001D #223 570

NOTINSTALLEDMSG Byte CODE:044D #207 854

NOTLOADTSR Byte CODE:01A7 #92 863 945 983

OLD\_INT1CHOFFSET Word CODE:019E #81 477 544 900

OLD\_INT1CHSEGMENT Word CODE:01A0 #83 545 901

OLD\_INT2FHOFFSET Word CODE:01A2 #85 520 549 909

OLD\_INT2FHSEGMENT Word CODE:01A4 #87 550 910

OLD\_INT9HOFFSET Word CODE:019A #77 396 539 891

OLD\_INT9HSEGMENT Word CODE:019C #79 540 892

PRINTDELAY Number 0003 #95 483

PRINTPOS Word CODE:01AA #97 617 620 623

PRINTSIGNATURE Near CODE:06BA 492 #597

REMOVEDMSG Byte CODE:046B #213 216 580

REMOVEDMSG\_LENGTH Number 0011 #216 581

REPLACEWITH Byte CODE:0137 #26 431

SAVEDSYMBOL Byte CODE:018A #72 760 783

SAVEFONT Near CODE:07C1 739 #807

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 20

Symbol Table

SETCURSIVE Near CODE:0767 376 #728

SIGNATURELINE1 Byte CODE:01AC #102 108 666

SIGNATURELINE2 Byte CODE:01D5 #109 116 676

SIGNATURELINE3 Byte CODE:0200 #117 123 686

SIGNATUREPRINTINGENABL Byte CODE:0177 #47 255 368 479 490

ED

TABLEBOTTOM Byte CODE:03E7 #192 194 696

TABLEBOTTOM\_LENGTH Number 0029 #194 697

TABLETOP Byte CODE:03BE #189 191 652

TABLETOP\_LENGTH Number 0029 #191 654

TRANSLATEENABLED Byte CODE:0176 #44 296 382 437

TRANSLATEFROM Byte CODE:016C #38 445

TRANSLATELENGTH Number 0005 #42 442

TRANSLATETO Byte CODE:0171 #40 42 454

TRUE Number 00FF #75 255 276 296 316 412 437 479 734 842 863

UNLOADTSR Byte CODE:01A6 #90 842 944 992

\_2FH\_EXIT Near CODE:06B5 574 #586

\_2FH\_STD Near CODE:063D 512 517 #519

\_ACTUALPRINT Near CODE:06F8 630 635 640 #642

\_ALREADYINSTALLED Near CODE:0881 873 #927

\_ALREADY\_INSTALLED Near CODE:0642 514 #522

\_BLOCK Near CODE:05C6 422 #428

\_CHECKF3 Near CODE:04B0 #247

\_CHECKF4 Near CODE:04D4 261 #268

\_CHECKF5 Near CODE:04F5 282 #288

\_CHECKF6 Near CODE:0516 302 #308

\_CHECK\_IGNORED Near CODE:05BA #420 424

\_CHECK\_TRANSLATE Near CODE:05D5 413 425 #435

\_CHECK\_TRANSLATE\_LOOP Near CODE:05E2 #444 448

\_DONTPRINT Near CODE:0626 487 #494

\_ERRORPARAM Near CODE:08DC #997

\_EXIT Near CODE:0892 878 926 931 #936

\_EXITHELP Near CODE:08ED 950 958 971 995 #1007

\_EXITSETCURSIVE Near CODE:07B1 775 #786

\_EXIT\_TMP Near CODE:0819 864 #877

\_F3 Near CODE:0555 #365

\_F4 Near CODE:0563 367 #371

\_F5 Near CODE:0574 373 #379

\_F6 Near CODE:0582 381 #385

\_FINISHTSR Near CODE:08D3 966 #991

\_GO Near CODE:05AA 404 #409

\_GREENF3 Near CODE:04CD 256 #263

\_GREENF4 Near CODE:04F1 277 #284

\_GREENF5 Near CODE:0512 297 #304

\_GREENF6 Near CODE:0533 317 #324

\_INITTSR Near CODE:07CE 15 #821 919

\_LETSPRINT Near CODE:0619 485 #489

\_NEXTCHAR Near CODE:08A8 #952 984 993

\_NOTMEM Near CODE:088B 887 #932

\_NOTREMOVE Near CODE:0690 556 562 #565

\_NOTREMOVINGNOW Near CODE:0804 844 #861

\_NOTTOPRINT Near CODE:062B 481 #497

\_OUTFX Near CODE:0537 322 #328

\_PRINTBOTTOM Near CODE:06F1 624 #637

\_PRINTCENTER Near CODE:06EA 621 #632

\_PRINTTOP Near CODE:06E3 618 #627

\_QUESTION Near CODE:08BC 964 #973

Turbo Assembler Version 3.1 06/03/21 15:03:01 Page 21

Symbol Table

\_QUIT Near CODE:05F7 433 438 449 #457

\_REDF3 Near CODE:04C6 #258

\_REDF4 Near CODE:04EA #279

\_REDF5 Near CODE:050B #299

\_REDF6 Near CODE:052C #319

\_REMOVE Near CODE:0878 851 #922

\_REMOVINGONPARAMETER Near CODE:07F0 843 #846

\_RESTORESYMBOL Near CODE:07A2 735 #777

\_SHIFTTABLE Near CODE:0779 #741 745

\_START Near CODE:0100 #13 1012

\_TEST\_FX Near CODE:0553 #363

\_TMP Near CODE:081C 875 #880

\_TRANSLATE Near CODE:05EE 446 #452

\_TRANSLATE\_OR\_IGNORE Near CODE:0590 370 378 384 387 390 #393

\_UNINSTALL Near CODE:0645 516 #526

\_UNLOADED Near CODE:06A4 563 #576

Groups & Segments Bit Size Align Combine Class Cref (defined at #)

CODE 16 08EE Para none CODE #10 11 11