Московский государственный технический университет им. Н.Э.Баумана

| Защищено:  Большаков С.А.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |  | Демонстрация ЛР:  Большаков С.А.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |
| --- | --- | --- |

**Отчет по лабораторной работе № 4 по курсу**

**Системное программирование**

#### "Циклы и перевод символов"

#### (есть ли дополнительные требования - НЕТ)

15

(количество листов)

Вариант № 20

| ИСПОЛНИТЕЛЬ: |  |
| --- | --- |
| студент группы **ИУ5-41Б** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| **Цыпышев Т.А.** | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Москва, МГТУ - 2024

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**СОДЕРЖАНИЕ**

[1. Цель выполнения лабораторной работы № 4 3](#_gjdgxs)

[2. Порядок и условия проведения работы № 4 3](#_30j0zll)

[3. Описание ошибок, возникших при отладке № 4 4](#_1fob9te)

[4. Блок-схема программы 5](#_3znysh7)

[5. Текст программы на языке Ассемблера (.LST) 6](#_2et92p0)

[6. Скриншот программы в TD.exe 14](#_tyjcwt)

[7. Результаты работы программы 14](#_3dy6vkm)

[8. Выводы по ЛР № 4 15](#_1t3h5sf)

# Цель выполнения лабораторной работы № 4

Лабораторная работа №4 выполняется для получения навыков разработки циклических программ и процедур на Ассемблере, получения знаний о перекодировке символов в среде.

# Порядок и условия проведения работы № 4

Разработать и отладить циклическую программу на языке Ассемблер для вывода на экран **20** последовательных прописных букв русского алфавита (начиная с символа “А” или другого символа, введенного с клавиатуры. Символы должны быть представлены в символьном (печатном) и шестнадцатеричном представлении (через черточку) в виде столбчатой таблицы (см. ниже). Каждая буква выводится в виде ее символьного представления и его 2-х разрядного шестнадцатеричного числа на одной строке. Например (СИМВОЛ – Шестнадцатеричный код):

**А – 80h**.

**Б – 81h**.

**В – 82h**.

**Г – 83h**.

**…**

В программе должна быть выполнена автоматическая шестнадцатеричная перекодировка, на основе преобразования машинного представления кода символа.

Шестнадцатеричная перекодировка (перевод одного представления в другое) должна выполняться командой **XLAT** по специальной таблице перекодировки вида: 0123456789ABCDEF. Переведенные представления русских букв выводятся на экран дисплея последовательно. В каждой строке выводиться только одна буква с переводом (например, "**А – 80h**" – пример для кодировки ДОС - ASCII). Для организации цикла использовать команду **LOOP**. Разработать блок-схему программы. Использовать MS VISIO для блок-схемы или другой доступный графический редактор.

После завершения вывода таблицы нужно организовать ожидание ввода нового символа с клавиатуры для вывода новой таблицы (процедура - **GETCH**). Если вводиться заранее предопределенный символ (например, символ “**\***”), то программа должна завершаться с сообщением о своем завершении. В противном случае циклически выводиться новая таблица для нового введенного символа. В программе разработать и использовать **четыре** отдельные процедуры:

* для ввода символа(без эха) (1 - **GETCH**),
* вывода одного символа (2 - **PUTCH**),
* для перевода буквы в двух символьное шестнадцатеричное представление (3-я процедура **HEX**) и перевода строки и возврата “каретки” экрана дисплея (4 - **CLRF**) и
* для очистки экрана (процедура - **CLSSCR**).

Выход из программы выполнить посредством прерывания 21H - 04CH после нажатия любой клавиши, с заданием кода завершения – 5.

# 

# 

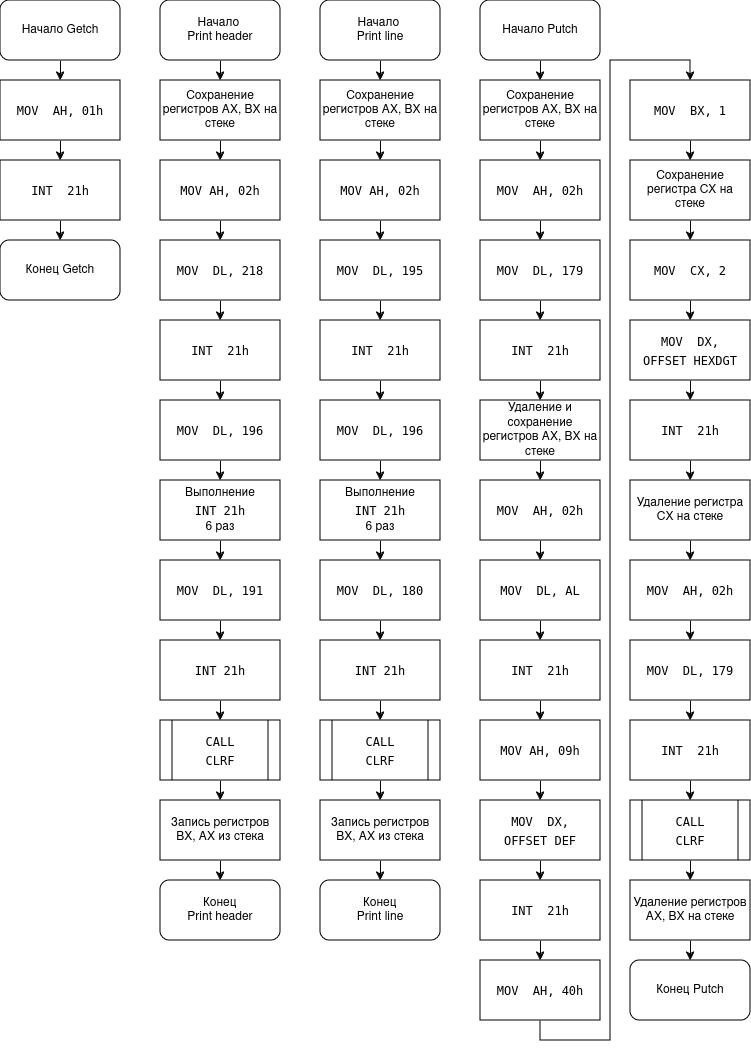
# 

# Описание ошибок, возникших при отладке № 4

| № п/п | Проявление ошибки | Причина ошибки | Способ устранения |
| --- | --- | --- | --- |
|  | Вывод мусора, вместо ввывода спецзнаков рамки | Использование кодов символов неправильной кодировки | Изменение заполнения регистра DL подобный способом  MOV DL, 180 |

# Блок-схема программы

# 



# Текст программы на языке Ассемблера (.LST)

Turbo Assembler Version 3.1 04/02/24 21:52:30 Page 1

c:\lab4\lab.asm

1 ;==================================================================================

2 0000 PRGR SEGMENT ; Начало сегмента PRGR

3 ASSUME CS:PRGR, DS:DATA ; Предполагаемые сегменты+

4 кода и данных

5

6 0000 BEGIN: ; Метка начала программы

7 0000 B8 0000s MOV AX, DATA ; Помещаем адрес сегмента +

8 данных в регистр AX

9 0003 8E D8 MOV DS, AX ; Загружаем регистр DS +

10 адресом сегмента данных

11 0005 B8 0002 MOV AX, 02h ; Помещаем значение 02h в +

12 регистр AX

13 0008 CD 10 INT 10h ; Вызываем прерывание 10h +

14 для установки видеорежима

15 000A 8C DA MOV DX, DS ; Помещаем адрес сегмента +

16 данных в регистр DX

17 000C 8E C2 MOV ES, DX ; Копируем адрес сегмента +

18 данных из DX в ES

19 000E BB 0000r MOV BX, OFFSET HEXTB ; Помещаем смещение HEXTB в +

20 регистр BX

21 0011 B9 0014 MOV CX, 20 ; Помещаем значение 20 в +

22 регистр CX

23 0014 B8 0080 MOV AX, 0080h ; Помещаем значение 0080h в +

24 регистр AX

25 0017 E8 0040 CALL PRINT\_HEADER ; Вызываем процедуру +

26 печати заголовка

27

28 001A СYCLE: ; Метка цикла СYCLE

29 001A E8 00D8 CALL HEX ; Вызываем процедуру HEX

30 001D E8 0078 CALL PUTCH ; Вызываем процедуру PUTCH

31 0020 83 F9 01 CMP CX, 1 ; Сравниваем значение в CX +

32 с 1

33 0023 74 08 JE END\_СYCLE ; Если CX равен 1, +

34 переходим к метке TEST1

35

36 0025 E8 0050 CALL PRINT\_LINE ; Вызываем процедуру +

37 печати линии

38 0028 B4 00 MOV AH, 00 ; Помещаем значение 00 в +

39 регистр AH

40 002A 40 INC AX ; Увеличиваем значение в AX+

41 на 1

42 002B E2 ED LOOP СYCLE ; Уменьшаем значение +

43 в CX и переходим к СYCLE

44

45 002D END\_СYCLE: ; Метка END\_СYCLE

46 002D E8 009C CALL PRINT\_FOOTER ; Вызываем процедуру +

47 печати нижнего колонтитула

48 0030 B4 09 MOV AH, 09h ; Помещаем значение 09h в +

49 регистр AH

50 0032 BA 0012r MOV DX, OFFSET MSG ; Помещаем смещение MSG в +

51 регистр DX

52 0035 CD 21 INT 21h ; Вызываем прерывание 21h +

53 (для вывода сообщения)

54 0037 E8 00AE CALL CLRF ; Вызываем процедуру +

55 очистки экрана

56

57 003A CHECK\_INPUT: ; Метка CHECK\_INPUT

Turbo Assembler Version 3.1 04/02/24 21:52:30 Page 2

c:\lab4\lab.asm

58 003A E8 0018 CALL GETCH ; Вызываем процедуру +

59 получения символа

60 003D 3C 2A CMP AL, '\*' ; Сравниваем символ с '\*'

61 003F 74 07 JE EXIT\_PROGRAM ; Если символ - '\*', +

62 переходим к выходу из программы

63 0041 B9 0014 MOV CX, 20 ; Помещаем значение 20 в +

64 регистр CX

65 0044 B4 00 MOV AH, 0 ; Помещаем значение 0 в +

66 регистр AH

67 0046 EB D2 JMP СYCLE ; Переходим к метке +

68 СYCLE

69

70 0048 EXIT\_PROGRAM: ; Метка EXIT\_PROGRAM

71 0048 B4 09 MOV AH, 09h ; Помещаем значение 09h в +

72 регистр AH

73 004A BA 003Fr MOV DX, OFFSET EXIT\_MSG ; Помещаем смещение EXIT\_MSG в+

74 регистр DX

75 004D CD 21 INT 21h ; Вызываем прерывание 21h +

76 (для вывода сообщения)

77 004F B4 4C MOV AH, 4CH ; Устанавливаем функцию +

78 завершения программы

79 0051 B0 00 MOV AL, 0 ; Устанавливаем код +

80 возврата 0

81 0053 CD 21 INT 21H ; Вызываем прерывание для +

82 завершения программы

83

84 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

85 0055 GETCH PROC

86 0055 B4 01 MOV AH, 01h ; Установка номера +

87 функции для чтения символа с клавиатуры

88 0057 CD 21 INT 21h ; Вызов прерывания 21h для +

89 чтения символа

90 0059 C3 RET ; Возвращение из процедуры

91 005A GETCH ENDP

92

93 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

94 005A PRINT\_HEADER PROC

95 005A 50 PUSH AX ; Сохранение регистра AX на+

96 стеке

97 005B 53 PUSH BX ; Сохранение регистра BX на+

98 стеке

99 005C B4 02 MOV AH, 02h

100 005E B2 DA MOV DL, 218

101 0060 CD 21 INT 21h ; Вывод символа ┌

102 0062 B2 C4 MOV DL, 196

103 0064 CD 21 INT 21h ; Вывод символа ─

104 0066 CD 21 INT 21h ; Вывод символа ─

105 0068 CD 21 INT 21h ; Вывод символа ─

106 006A CD 21 INT 21h ; Вывод символа ─

107 006C CD 21 INT 21h ; Вывод символа ─

108 006E B2 BF MOV DL, 191

109 0070 CD 21 INT 21h ; Вывод символа ┐

110 0072 E8 0073 CALL CLRF ; Вызов процедуры для +

111 очистки экрана

112 0075 5B POP BX ; Восстановление регистра+

113 BX из стека

114 0076 58 POP AX ; Восстановление регистра+

Turbo Assembler Version 3.1 04/02/24 21:52:30 Page 3

c:\lab4\lab.asm

115 AX из стека

116 0077 C3 RET ; Возвращение из процедуры

117 0078 PRINT\_HEADER ENDP

118

119 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

120 0078 PRINT\_LINE PROC

121 0078 50 PUSH AX

122 0079 53 PUSH BX

123 007A B4 02 MOV AH, 02h

124 007C B2 C3 MOV DL, 195

125 007E CD 21 INT 21h ; Вывод символа ┬

126 0080 B2 C4 MOV DL, 196

127 0082 CD 21 INT 21h ; Вывод символа ─

128 0084 CD 21 INT 21h ; Вывод символа ─

129 0086 CD 21 INT 21h ; Вывод символа ─

130 0088 CD 21 INT 21h ; Вывод символа ─

131 008A CD 21 INT 21h ; Вывод символа ─

132 008C CD 21 INT 21h ; Вывод символа ─

133 008E B2 B4 MOV DL, 180

134 0090 CD 21 INT 21h ; Вывод символа ├

135 0092 E8 0053 CALL CLRF ; Очистка экрана

136 0095 5B POP BX

137 0096 58 POP AX

138 0097 C3 RET

139 0098 PRINT\_LINE ENDP

140

141 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

142 0098 PUTCH PROC

143 0098 50 PUSH AX

144 0099 53 PUSH BX

145 009A B4 02 MOV AH, 02h

146 009C B2 B3 MOV DL, 179

147 009E CD 21 INT 21h ; Вывод символа │

148 00A0 5B POP BX

149 00A1 58 POP AX

150 00A2 50 PUSH AX

151 00A3 53 PUSH BX

152 00A4 B4 02 MOV AH, 02h

153 00A6 8A D0 MOV DL, AL

154 00A8 CD 21 INT 21h ; Вывод символа

155 00AA B4 09 MOV AH, 09h

156 00AC BA 0057r MOV DX, OFFSET DEF

157 00AF CD 21 INT 21h ; Вывод текста " - "

158 00B1 B4 40 MOV AH, 40h

159 00B3 BB 0001 MOV BX, 1

160 00B6 51 PUSH CX

161 00B7 B9 0002 MOV CX, 2

162 00BA BA 0010r MOV DX, OFFSET HEXDGT

163 00BD CD 21 INT 21h ; Вывод шестнадцатеричных+

164 символов

165 00BF 59 POP CX

166 00C0 B4 02 MOV AH, 02h

167 00C2 B2 B3 MOV DL, 179

168 00C4 CD 21 INT 21h ; Вывод символа │

169 00C6 E8 001F CALL CLRF ; Очистка экрана

170 00C9 5B POP BX

171 00CA 58 POP AX

Turbo Assembler Version 3.1 04/02/24 21:52:30 Page 4

c:\lab4\lab.asm

172 00CB C3 RET

173 00CC PUTCH ENDP

174

175 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

176 00CC PRINT\_FOOTER PROC

177 00CC B4 02 MOV AH, 02h

178 00CE B2 C0 MOV DL, 192

179 00D0 CD 21 INT 21h ; Вывод символа └

180 00D2 B2 C4 MOV DL, 196

181 00D4 CD 21 INT 21h ; Вывод символа ─

182 00D6 CD 21 INT 21h ; Вывод символа ─

183 00D8 CD 21 INT 21h ; Вывод символа ─

184 00DA CD 21 INT 21h ; Вывод символа ─

185 00DC CD 21 INT 21h ; Вывод символа ─

186 00DE CD 21 INT 21h ; Вывод символа ─

187 00E0 B2 D9 MOV DL, 217

188 00E2 CD 21 INT 21h ; Вывод символа ┘

189 00E4 E8 0001 CALL CLRF ; Очистка экрана

190 00E7 C3 RET

191 00E8 PRINT\_FOOTER ENDP

192

193 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

194 00E8 CLRF PROC

195 00E8 B4 02 MOV AH, 02h

196 00EA B2 0A MOV DL, 10

197 00EC CD 21 INT 21h ; Перевод каретки

198 00EE B4 02 MOV AH, 02h

199 00F0 B2 0D MOV DL, 13

200 00F2 CD 21 INT 21h ; Перевод строки

201 00F4 C3 RET

202 00F5 CLRF ENDP

203

204 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

205 00F5 HEX PROC

206 00F5 50 PUSH AX

207 00F6 BE 0000 MOV SI, 0

208 00F9 50 PUSH AX

209 00FA D0 E8 SHR AL, 1

210 00FC D1 E8 SHR AX, 1

211 00FE D1 E8 SHR AX, 1

212 0100 D1 E8 SHR AX, 1

213 0102 D7 XLAT

214 0103 88 84 0010r MOV HEXDGT[SI], AL ; Сохранение старших 4 бит +

215 в массиве HEXDGT

216 0107 46 INC SI

217 0108 58 POP AX

218 0109 24 0F AND AL, 00001111b

219 010B D7 XLAT

220 010C 88 84 0010r MOV HEXDGT[SI], AL ; Сохранение младших 4 бит +

221 в массиве HEXDGT

222 0110 58 POP AX

223 0111 C3 RET

224 0112 HEX ENDP

225

226 0112 PRGR ENDS

227 ;==================================================================================

228

Turbo Assembler Version 3.1 04/02/24 21:52:30 Page 5

c:\lab4\lab.asm

229 0000 DATA SEGMENT

230 0000 30 31 32 33 34 35 36+ HEXTB DB '0123456789ABCDEF'

231 37 38 39 41 42 43 44+

232 45 46

233 0010 02\*(2A) HEXDGT DB 2 DUP ('\*')

234 0012 50 72 65 73 73 20 61+ MSG DB 'Press any key to continue or "\*" to exit... $'

235 6E 79 20 6B 65 79 20+

236 74 6F 20 63 6F 6E 74+

237 69 6E 75 65 20 6F 72+

238 20 22 2A 22 20 74 6F+

239 20 65 78 69 74 2E 2E+

240 2E 20 24

241 003F 45 78 69 74 69 6E 67+ EXIT\_MSG DB 'Exiting the program... $'

242 20 74 68 65 20 70 72+

243 6F 67 72 61 6D 2E 2E+

244 2E 20 24

245 0057 20 2D 20 24 DEF DB ' - $'

246 005B DATA ENDS

247 ;==================================================================================

248

249 0000 STK SEGMENT STACK

250 0000 0100\*(00) DB 256 DUP (0)

251 0100 STK ENDS

252 ;==================================================================================

253

254 END BEGIN

Turbo Assembler Version 3.1 04/02/24 21:52:30 Page 6

Symbol Table

Symbol Name Type Value Cref (defined at #)

??DATE Text "04/02/24"

??FILENAME Text "lab "

??TIME Text "21:52:30"

??VERSION Number 030A

@CPU Text 0101H

@CURSEG Text STK #2 #229 #249

@FILENAME Text LAB

@WORDSIZE Text 2 #2 #229 #249

BEGIN Near PRGR:0000 #6 254

CHECK\_INPUT Near PRGR:003A #57

CLRF Near PRGR:00E8 54 110 135 169 189 #194

DEF Byte DATA:0057 156 #245

END\_СYCLE Near PRGR:002D 33 #45

EXIT\_MSG Byte DATA:003F 73 #241

EXIT\_PROGRAM Near PRGR:0048 61 #70

GETCH Near PRGR:0055 58 #85

HEX Near PRGR:00F5 29 #205

HEXDGT Byte DATA:0010 162 214 220 #233

HEXTB Byte DATA:0000 19 #230

MSG Byte DATA:0012 50 #234

PRINT\_FOOTER Near PRGR:00CC 46 #176

PRINT\_HEADER Near PRGR:005A 25 #94

PRINT\_LINE Near PRGR:0078 36 #120

PUTCH Near PRGR:0098 30 #142

YCLE Near PRGR:001A #28 42 67

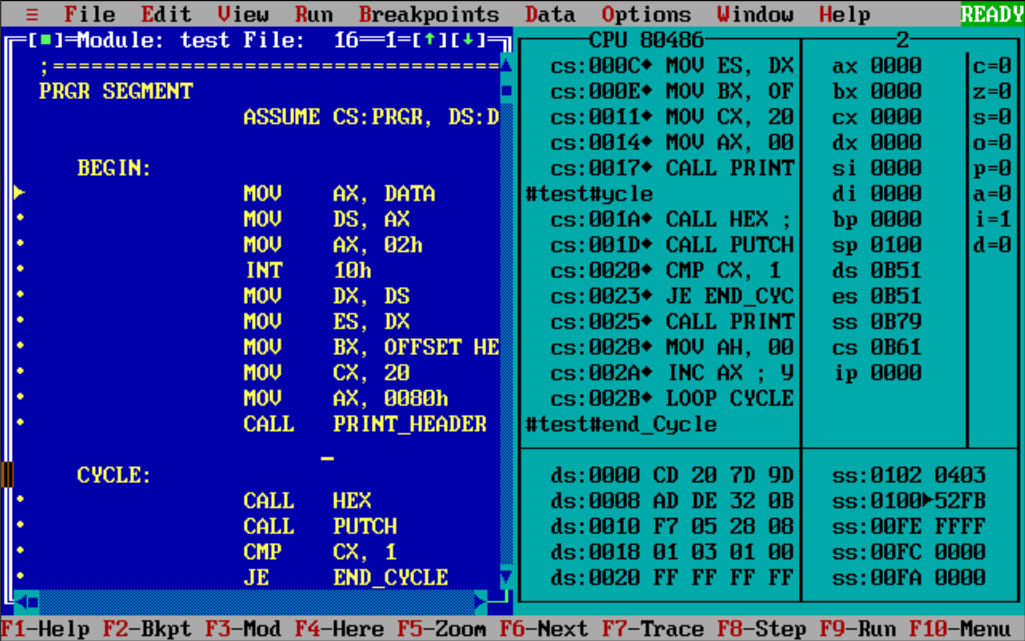
Groups & Segments Bit Size Align Combine Class Cref (defined at #)

DATA 16 005B Para none 3 7 #229

PRGR 16 0112 Para none #2 3

STK 16 0100 Para Stack #249

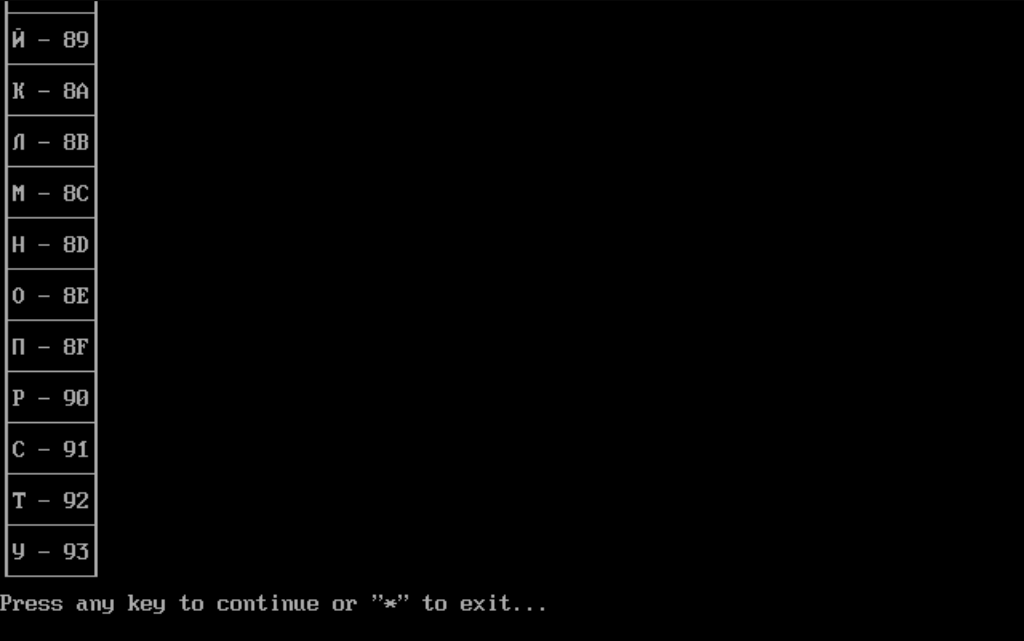
# Скриншот программы в TD.exe



# Результаты работы программы

Программа выводит 20 букв начиная с русской “A”

Далее можно ввести любой другой символ или завершить программу, написав символ “\*”



А - 80

Б - 81

В - 82

Г - 83

Д - 84

Е - 85

Ж - 86

З - 87

И - 88

Й - 89

К - 8A

Л - 8B

М - 8C

Н - 8D

О - 8E

П - 8F

Р - 90

С - 91

Т - 92

У - 93

# Выводы по ЛР № 4

Разработан файл .ASM и соответствующие файлы приложения и листинга на языке Ассемблер. Программа выполняется в циклическом режиме до ввода ‘\*’, выводя по каждому нажатию клавиши 20 символов на экран в виде “Символ-16-ричный код”, где первый из 20 символов – введенный, остальные 19 – следующие за первым по возрастанию кодировки на 1. Программа работает корректно, мы изучили циклы и перекодировку символов.