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

|  |  |
| --- | --- |
| Защищено: Большаков С.А.  " " 2025 г. | Демонстрация ЛР: Большаков С.А.  " " 2025 г. |

**Отчет по лабораторной работе № 5 по курсу Системное программирование**

**" Ввод/вывод в адреса и числа"**

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

12

(количество листов)

Вариант № **2**

|  |
| --- |
| ИСПОЛНИТЕЛЬ: |
| студент группы **ИУ5-42Б** |
| (подпись) |
| **Афонин И. И.** |
| "23" апреля 2025 г. |

Москва, МГТУ - 2025

**СОДЕРЖАНИЕ**

[1. Цель выполнения лабораторной работы № 5 3](#_Toc196269411)

[2. Порядок и условия проведения работы № 5 3](#_Toc196269412)

[3. Описание ошибок, возникших при отладке № 5 3](#_Toc196269413)

[4. Блок-схема программы 4](#_Toc196269414)

[5. Скриншот программы ЛР № 5 в TD.exe 5](#_Toc196269415)

[6. Текст программы на языке Ассемблера 5](#_Toc196269416)

[7. Результаты работы программы 12](#_Toc196269417)

[8. Выводы по ЛР № 5 12](#_Toc196269418)

# Цель выполнения лабораторной работы № 5

Целью лабораторной работы №5 является освоение работы с массивами и строками в языке Ассемблер, включая использование буферизации ввода, команд цепочек (LODSB, STOSB, MOVSB), а также преобразование символов в их шестнадцатеричное представление.

# Порядок и условия проведения работы № 5

Разработать и отладить программу на языке Ассемблер для ввода и буферизации строки символов с клавиатуры (последовательности символов) и затем последовательного их вывода на экран в шестнадцатеричном представлении (через пробел). В данной программе для корректной работы необходимо предусмотреть запоминание строки символов в байтовом массиве. Программа и блок-схема должны содержать вложенные циклы (двойные циклы).

Признак завершения ввода отдельной строки с клавиатуры – это символ "**$**" (он вводиться с клавиатуры для завершения ввода строки). Между введенной строкой символов и их шестнадцатеричным представлением должен располагаться знак равенства (“**=**”). Максимальное число вводимых символов не должно превышать 20-ти. В данной программе цикл ввода (с клавиатуры) организуется с помощью команд условного (JE, JNE) перехода и команды безусловного перехода (JMP). После завершения ввода строки выполняется ее автоматический вывод. Организовать цикл ввода строк до ввода специального символа ('\*'). Пример результата работы одного цикла программы показан ниже:

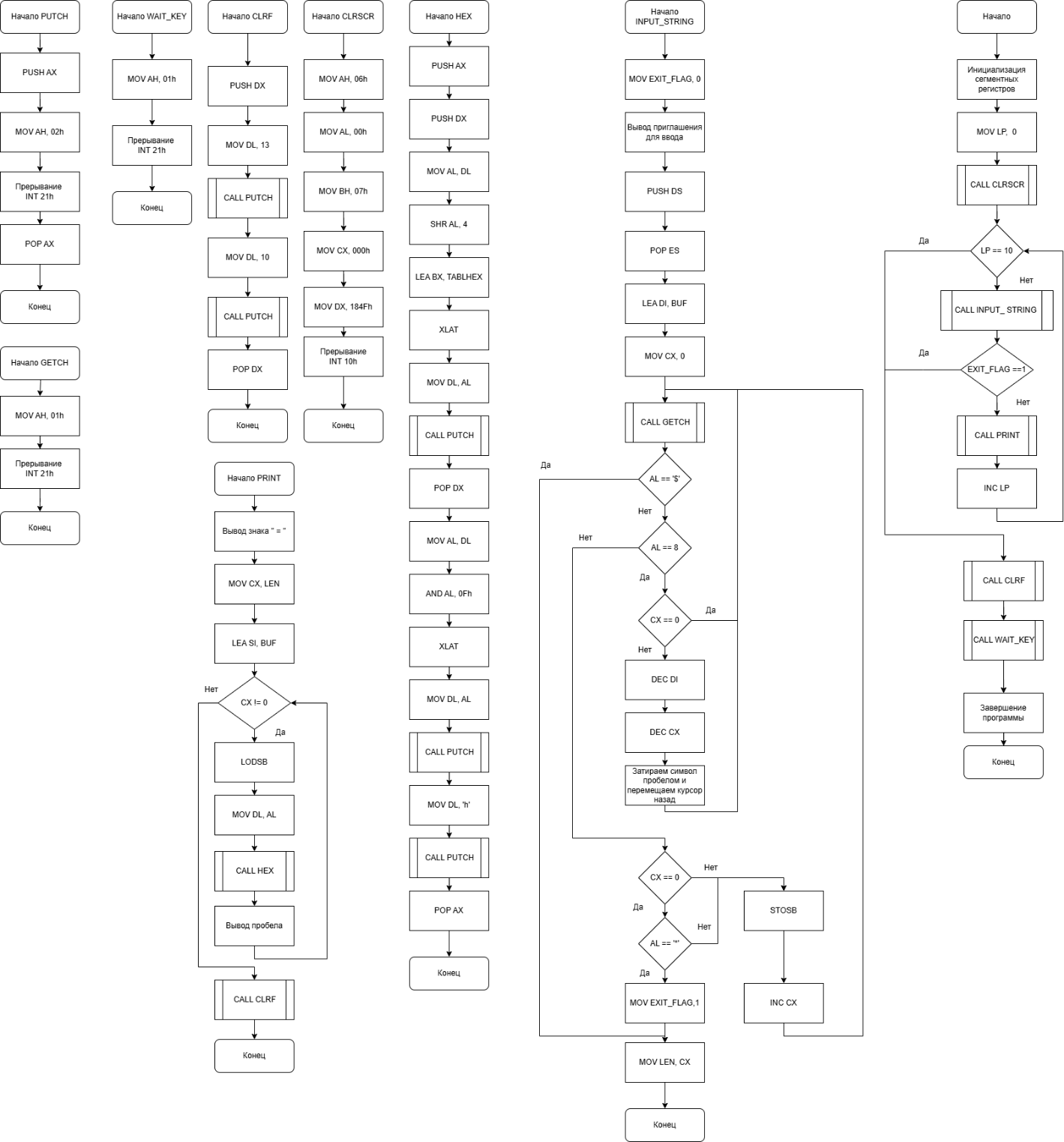
**АБВ$ = 80 81 82**

Оформить отчет по ЛР.

# Описание ошибок, возникших при отладке № 5

Ошибки отсутствуют.

# Блок-схема программы

****

# Скриншот программы ЛР № 5 в TD.exe

****

# Текст программы на языке Ассемблера

Turbo Assembler Version 3.1 04/23/25 02:04:25 Page 1

lab5\_25.asm

1 ; Афонин Иван Игоревич Вариант 2 Группа ИУ5-42Б

2 ;СЕГМЕНТ ДАННЫХ

3 0000 DTSEG SEGMENT PARA 'DATA'

4 ; Справочные сообщения для пользователя.

5 0000 8C A0 AA E1 A8 AC A0+ MES0 DB "Максимальное кол-во символов в строке - 20", 0Dh, 0Ah, '$'

6 AB EC AD AE A5 20 AA+

7 AE AB 2D A2 AE 20 E1+

8 A8 AC A2 AE AB AE A2+

9 20 A2 20 E1 E2 E0 AE+

10 AA A5 20 2D 20 32 30+

11 0D 0A 24

12 002D 91 A8 AC A2 AE AB 20+ MES1 DB "Символ окончание ввода строки - `знак доллара`", 0Dh, 0Ah, '$'

13 AE AA AE AD E7 A0 AD+

14 A8 A5 20 A2 A2 AE A4+

15 A0 20 E1 E2 E0 AE AA+

16 A8 20 2D 20 60 A7 AD+

17 A0 AA 20 A4 AE AB AB+

18 A0 E0 A0 60 0D 0A 24

19 005E 8C A0 AA E1 A8 AC A0+ MES2 DB "Максимальное кол-во строк - 10", 0Dh, 0Ah, '$'

20 AB EC AD AE A5 20 AA+

21 AE AB 2D A2 AE 20 E1+

22 E2 E0 AE AA 20 2D 20+

23 31 30 0D 0A 24

24 007F 91 A8 AC A2 AE AB 20+ MES3 DB "Символ окончания ввода строк - \*", 0Dh, 0Ah, '$'

25 AE AA AE AD E7 A0 AD+

26 A8 EF 20 A2 A2 AE A4+

27 A0 20 E1 E2 E0 AE AA+

28 20 2D 20 2A 0D 0A 24

29 00A2 28\*(??) BUF\_SYMBS DB 40 DUP(?) ; Символьный буфер для введенной строки.

30 00CA 30 31 32 33 34 35 36+ HEX\_TABLE DB '0123456789ABCDEF' ; Таблица перекодировки для процедуры

31 37 38 39 41 42 43 44+

32 45 46

33 00DA ???? BUF\_CX DW ? ; Буфер для счетчика внешнего цикла.

34 00DC 0028 ROW\_SYMBS DW 40 ; Максимальное кол-во символов в одной строке.

35 00DE 000A ROWS DW 10 ; Максимальное кол-во вводимых строк.

36 00E0 DTSEG ENDS

37 ; СЕГМЕНТ СТЕКА

38 0000 STSEG SEGMENT PARA 'STACK'

39 0000 C8\*(00) DB 200 DUP(0)

40 00C8 STSEG ENDS

41 ; СЕГМЕНТ КОДА

42 0000 CDSEG SEGMENT PARA 'CODE'

43 ASSUME DS:DTSEG, SS:STSEG, CS:CDSEG

44 ; СЕГМЕНТ КОДА

45 ; Вход в прСограмму

46 0000 MAIN PROC

47 0000 B8 0000s MOV AX, DTSEG

48 0003 8E D8 MOV DS, AX

49 ; Вывод справочных сообщений.

50 0005 BA 0000r MOV DX, OFFSET MES0

51 0008 E8 0078 CALL PUTMES

52 000B BA 002Dr MOV DX, OFFSET MES1

53 000E E8 0072 CALL PUTMES

54 0011 BA 005Er MOV DX, OFFSET MES2

55 0014 E8 006C CALL PUTMES

56 0017 BA 007Fr MOV DX, OFFSET MES3

57 001A E8 0066 CALL PUTMES

Turbo Assembler Version 3.1 04/23/25 02:04:25 Page 2

lab5\_25.asm

58 001D E8 0068 CALL CLRF

59 0020 BB 00CAr MOV BX, OFFSET HEX\_TABLE

60 0023 8B 0E 00DEr MOV CX, ROWS ; Счетчик для цикла LOOP0.

61 ; ЦИКЛ ПО СТРОКАМ

62 0027 LOOP0:

63 ; Цикл ввода строк.

64 0027 89 0E 00DAr MOV BUF\_CX, CX

65 002B 8B 0E 00DCr MOV CX, ROW\_SYMBS ; Счетчик для цикла LOOP1.

66 002F BE 0000 MOV SI, 0 ; Счетчик символьного буфера.

67 0032 LOOP1:

68 ; Цикл ввода одной строки.

69 0032 E8 005E CALL GETCH ; Ввод символа.

70 ; Проверка окончания ввода строк.

71 0035 3C 2A CMP AL, '\*'

72 0037 74 0B JE GO0

73 ; Проверка окончания ввода строки.

74 0039 3C 24 CMP AL, '$'

75 003B 74 07 JE GO0

76 003D 88 84 00A2r MOV BUF\_SYMBS[SI], AL ; Запись считанного символа в символьный буфер.

77 0041 46 INC SI

78 0042 E2 EE LOOP LOOP1

79 0044 GO0:

80 ;

81 ; Проверка, что строка пустая или является конечной (введен символ "\*"

82 0044 83 FE 00 CMP SI, 0

83 0047 74 1D JE GO1

84 ; Ввывод строки " = ".

85 0049 E8 0030 CALL PUTSPACE

86 004C B2 3D MOV DL, 3DH

87 ; вывод символа пробела

88 004E E8 0026 CALL PUTCH

89 0051 E8 0028 CALL PUTSPACE

90 0054 8B CE MOV CX, SI ; Счетчик для цикла LOOP2 (равен кол-во введенных ;символов).

91 0056 BE 0000 MOV SI, 0 ; Счетчик символьного буфера.

92 ;ЦИКЛ чтения и преобразования

93 0059 LOOP2:

94 ; Цикл преобразования символов введеной строки.

95 0059 8A 84 00A2r MOV AL, BUF\_SYMBS[SI] ; Чтение символа из символьного буфера.

96 005D 46 INC SI

97 005E E8 0037 CALL HEX ; Перекодировка и печать

98 ;Пробел

99 0061 E8 0018 CALL PUTSPACE

100 ; Конец цикла вывода из буфера

101 0064 E2 F3 LOOP LOOP2

102 0066 GO1:

103 ; Проверка, что введенная строка является конечной.

104 0066 3C 2A CMP AL, '\*'

105 0068 74 09 JE END\_PROG

106

107 006A E8 001B CALL CLRF

108 ; для цикла ввода строки

109 006D 8B 0E 00DAr MOV CX, BUF\_CX

110 ; Конец цмкла ввода строк

111 0071 E2 B4 LOOP LOOP0

112

113 0073 END\_PROG:

114 ; Завершение работы программы.

Turbo Assembler Version 3.1 04/23/25 02:04:25 Page 3

lab5\_25.asm

115

116 0073 B4 4C MOV AH, 4CH

117 0075 CD 21 INT 21H

118 0077 MAIN ENDP

119 ;ПРОЦЕДУРЫ ПРОГРАММЫ

120 0077 PUTCH PROC

121 ; Процедура вывода символа.

122 0077 B4 02 MOV AH, 02H

123 0079 CD 21 INT 21H

124 007B C3 RET

125 007C PUTCH ENDP

126 007C PUTSPACE PROC

127 ; Процедура вывода пробела.

128 007C B2 20 MOV DL, 20H

129 007E B4 02 MOV AH, 02H

130 0080 CD 21 INT 21H

131 0082 C3 RET

132 0083 PUTSPACE ENDP

133 0083 PUTMES PROC

134 ; Процедура вывода текстового сообщения.

135 0083 B4 09 MOV AH, 09H

136 0085 CD 21 INT 21H

137 0087 C3 RET

138 0088 PUTMES ENDP

139 0088 CLRF PROC

140 ; Процедура перевода строки и возврата каретки.

141 0088 B4 02 MOV AH, 02H

142 008A B2 0A MOV DL, 0AH

143 008C CD 21 INT 21H

144 008E B2 0D MOV DL, 0DH

145 0090 CD 21 INT 21H

146 0092 C3 RET

147 0093 CLRF ENDP

148 0093 GETCH PROC

149 ; Процедура ввода символа.

150 0093 B4 01 MOV AH, 01H

151 0095 CD 21 INT 21H

152 0097 C3 RET

153 0098 GETCH ENDP

154 0098 HEX PROC

155 ; Процедура перекадировки

156 0098 50 PUSH AX

157 ; Получаем значение старшего полубайта.

158 0099 8A D0 MOV DL, AL

159 009B D0 EA D0 EA D0 EA D0+ SHR DL, 4

160 EA

161 00A3 8A C2 MOV AL, DL

162 00A5 D7 XLAT

163 ; Выводим значение старшего полубайта.

164 00A6 8A D0 MOV DL, AL

165 00A8 E8 FFCC CALL PUTCH

166 ; Получаем значение младшего полубайта.

167 00AB 58 POP AX

168 00AC 8A D0 MOV DL, AL

169 00AE D0 E2 D0 E2 D0 E2 D0+ SHL DL, 4

170 E2

171 00B6 D0 EA D0 EA D0 EA D0+ SHR DL, 4

Turbo Assembler Version 3.1 04/23/25 02:04:25 Page 4

lab5\_25.asm

172 EA

173 00BE 8A C2 MOV AL, DL

174 00C0 D7 XLAT

175 ; Выводим значение младшего полубайта.

176 00C1 8A D0 MOV DL, AL

177 00C3 E8 FFB1 CALL PUTCH

178 00C6 C3 RET

179 00C7 ENDP HEX

180

181 00C7 CDSEG ENDS

182

183 END MAIN

Turbo Assembler Version 3.1 04/23/25 02:04:25 Page 5

Symbol Table

Symbol Name Type Value Cref (defined at #)

??DATE Text "04/23/25"

??FILENAME Text "lab5\_25 "

??TIME Text "02:04:25"

??VERSION Number 030A

@CPU Text 0101H

@CURSEG Text CDSEG #3 #38 #42

@FILENAME Text LAB5\_25

@WORDSIZE Text 2 #3 #38 #42

BUF\_CX Word DTSEG:00DA #33 64 109

BUF\_SYMBS Byte DTSEG:00A2 #29 76 95

CLRF Near CDSEG:0088 58 107 #139

END\_PROG Near CDSEG:0073 105 #113

GETCH Near CDSEG:0093 69 #148

GO0 Near CDSEG:0044 72 75 #79

GO1 Near CDSEG:0066 83 #102

HEX Near CDSEG:0098 97 #154

HEX\_TABLE Byte DTSEG:00CA #30 59

LOOP0 Near CDSEG:0027 #62 111

LOOP1 Near CDSEG:0032 #67 78

LOOP2 Near CDSEG:0059 #93 101

MAIN Near CDSEG:0000 #46 183

MES0 Byte DTSEG:0000 #5 50

MES1 Byte DTSEG:002D #12 52

MES2 Byte DTSEG:005E #19 54

MES3 Byte DTSEG:007F #24 56

PUTCH Near CDSEG:0077 88 #120 165 177

PUTMES Near CDSEG:0083 51 53 55 57 #133

PUTSPACE Near CDSEG:007C 85 89 99 #126

ROWS Word DTSEG:00DE #35 60

ROW\_SYMBS Word DTSEG:00DC #34 65

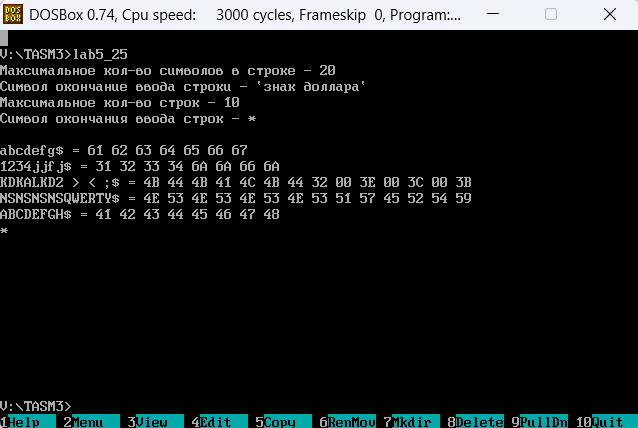
Groups & Segments Bit Size Align Combine Class Cref (defined at #)

CDSEG 16 00C7 Para none CODE #42 43

DTSEG 16 00E0 Para none DATA #3 43 47

STSEG 16 00C8 Para none STACK #38 43

# Результаты работы программы



# Выводы по ЛР № 5

В ходе выполнения лабораторной работы были освоены методы работы с массивами и строками в Ассемблере, включая буферизацию ввода с использованием команд STOSB и LODSB. Реализована программа, которая корректно сохраняет введённые символы в буфер и преобразует их в шестнадцатеричное представление с помощью процедуры HEX.