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 № 5 по курсу Системное программирование

" Ввод/вывод в адреса и числа "

(есть ли дополнительные требования - НЕТ)

16
(количество листов)
Вариант № <3>

ИСПОЛНИТЕЛЬ:

студент группы **ИУ5-41Б** _____
(подпись)

Бирюкова Е.И.

"__"_____2024 г.

Москва, МГТУ - 2024

СОДЕРЖАНИЕ

1. Цель выполнения лабораторной работы № 5.....3
2. Порядок и условия проведения работы № 5 3
3. Описание ошибок, возникших при отладке № 5 3
4. Блок-схема программы.....4
5. Скриншот программы в TD.exe.....5
6. Текст программы на языке Ассемблера 6
7. Результаты работы программы.....14
8. Выводы по ЛР № 5.....15

1. Цель выполнения лабораторной 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).

2. Порядок и условия проведения работы № 5

Признак завершения ввода отдельной строки с клавиатуры – это символ "$" (он вводится с клавиатуры для завершения ввода строки). Между введенной строкой символов и их шестнадцатеричным представлением должен располагаться знак равенства (“=”). Максимальное число вводимых символов не должно превышать 20-ти. В данной программе цикл ввода (с клавиатуры) организуется с помощью команд условного (JE, JNE) перехода и команды безусловного перехода (JMP). После завершения ввода строки выполняется ее автоматический вывод. Организовать цикл ввода строк до ввода специального символа ('*'). Пример результата работы одного цикла программы показан ниже: АБВ$ = 80 81 82.

Программа должна работать в циклическом режиме ввода строк (для внешнего цикла используется команда LOOP): после ввода одной строки запрашивается следующая (максимальное число вводимых строк для одного запуска программы равно 10). Завершение цикла ввода строк может быть выполнено при вводе символа звездочка (“*”), который должен быть введен в первой позиции строки. Вводимые символы строки записываются в символьный массив (буфер символов), максимальное число введенных символов равно 20-ти. Цикл ввода строки организуется командами условного и безусловного перехода. При вводе нужно подсчитать число введенных символов, включая символ доллара (“$”). Для вывода организуется цикл с помощью команды цикла (LOOP). В программе использовать процедуры предыдущих лабораторных данного цикла (ввода символа, печати, перевода строки и др.).

Для ввода/вывода строки и ее шестнадцатеричного представления разрабатываются дополнительная процедура HEX. Организовать очистку экрана до начала работы программы, а также после ее завершения.

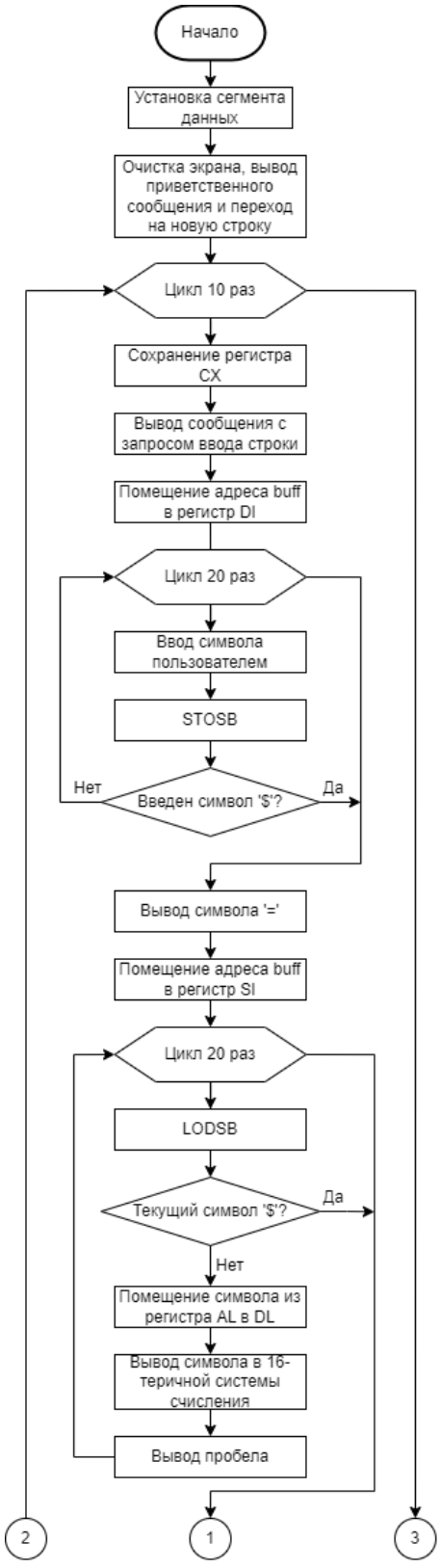
Оформить отчет по ЛР. Процедура HEX для перевода символа может быть использована из 4-й ЛР. Для очистки экрана использовать отдельную процедуру – CLRSCR, а в ней нужно использовать прерывание BIOS 010h, для очистки экрана. В программе должно быть построено три цикла: цикл ввода символов, цикл вывода их шестнадцатеричного представления и общий цикл ввода строк. При организации вложенных циклов необходимо сохранять регистр CX. Проверка завершения внешнего цикла может быть выполнена командой CMP и командами условного перехода, например JE .

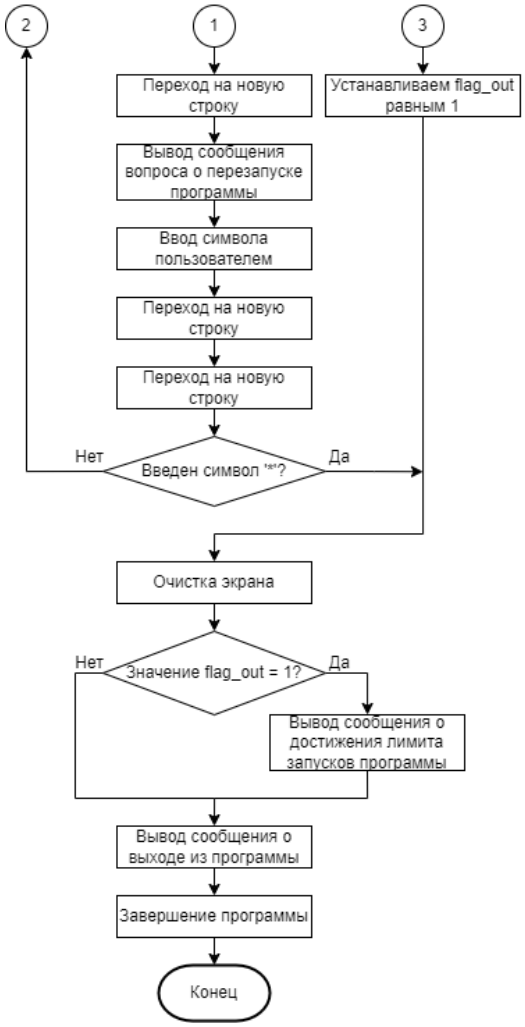
3. Описание ошибок, возникших при отладке № 5

№ п/п	Проявление ошибки	Причина ошибки	Способ устранения
1.	При значении регистра AL, равного ‘$’, не выполняется команда условного перехода к метке flag1: CMP AL, '$' JNE flag1	JNE – команда условного перехода, если не равно, JE – если равно.	Заменим в коде команду условного перехода JNE на команду JE.
2.	Внешний цикл ввода строк cycle_main не прекращает свою работу.	Не сохранялся регистр CX при начале работы цикла. Регистр CX	В начале работы цикла cycle_main добавим строку PUSH CX, а при

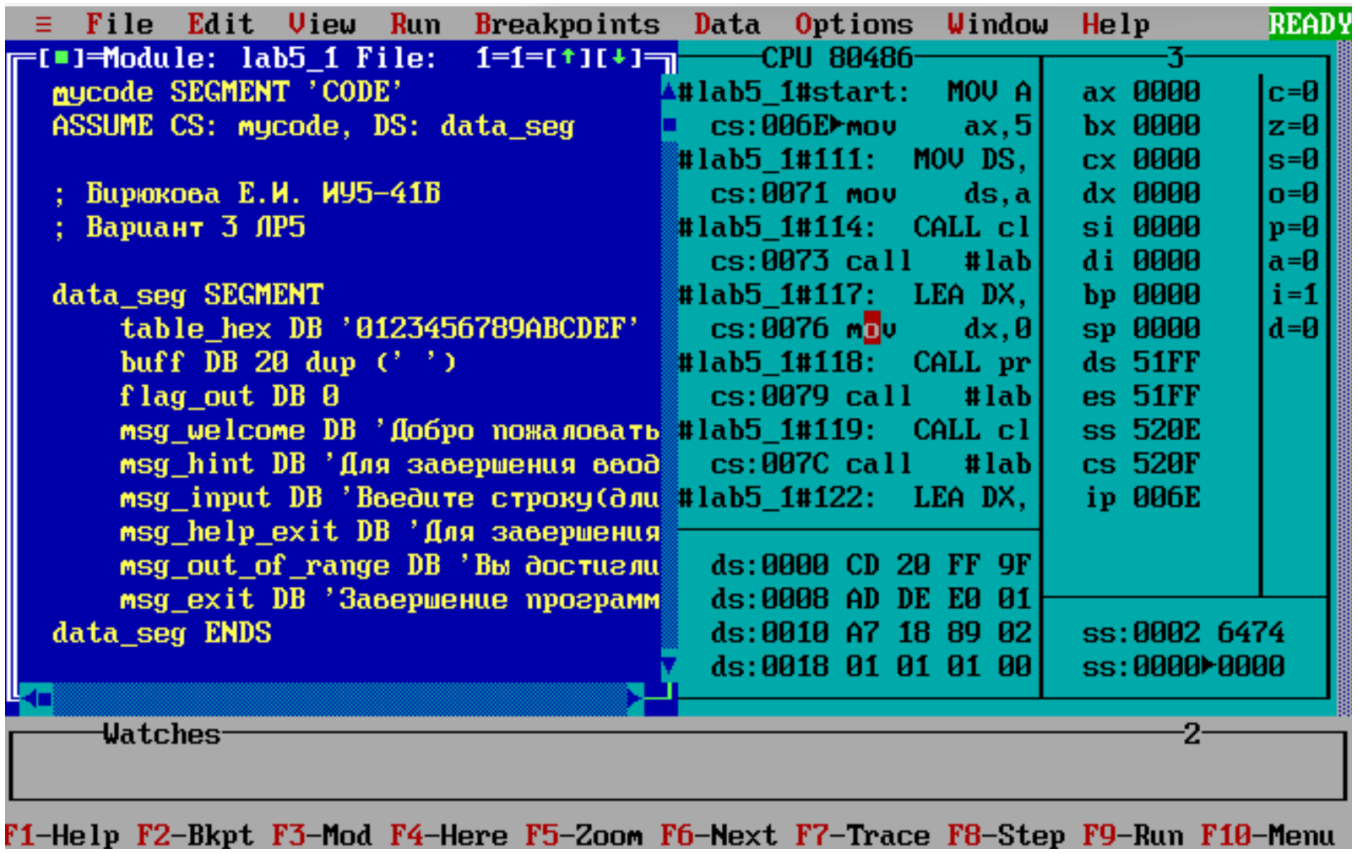
		изменялся при работе вложенных циклов cycle_input и cycle_output.	окончании цикла – POP CX.

4. Блок-схема программы





5. Скриншот программы в TD.exe



6. Текст программы на языке Ассемблера

Листинг программы:


```
 1 0000                 mycode SEGMENT 'CODE'
 2                  ASSUME CS: mycode, DS: data_seg
 3
 4                  ; Бирюкова Е. И. ИУ5-41Б
 5                  ; Вариант 3 ЛР5
 6
 7 0000                 data_seg SEGMENT
 8 0000  30 31 32 33 34 35 36+  table_hex DB '0123456789ABCDEF'
 9       37 38 39 41 42 43 44+
10       45 46
11 0010  14*(20)           buff DB 20 dup (' ')
12 0024  00              flag_out DB 0
13 0025  84 AE A1 E0 AE 20 AF+  msg_welcome DB 'Добро пожаловать!$'
14       AE A6 A0 AB AE A2 A0+
15       E2 EC 21 24
16 0037  84 AB EF 20 A7 A0 A2+  msg_hint DB 'Для завершения ввода строки введите символ: $'
17       A5 E0 E8 A5 AD A8 EF+
18       20 A2 A2 AE A4 A0 20+
19       E1 E2 E0 AE AA A8 20+
20       A2 A2 A5 A4 A8 E2 A5+
21       20 E1 A8 AC A2 AE AB+
22       3A 20 24
23 0064  82 A2 A5 A4 A8 E2 A5+  msg_input DB 'Введите строку(длиной до 20 символов!): $'
24       20 E1 E2 E0 AE AA E3+
25       28 A4 AB A8 AD AE A9+
26       20 A4 AE 20 32 30 20+
27       E1 A8 AC A2 AE AB AE+
28       A2 21 29 3A 20 24
29 008D  84 AB EF 20 A7 A0 A2+  msg_help_exit DB 'Для завершения программы введите "*": $'
30       A5 E0 E8 A5 AD A8 EF+
31       20 AF E0 AE A3 E0 A0+
32       AC AC EB 20 A2 A2 A5+
33       A4 A8 E2 A5 20 22 2A+
34       22 3A 20 24
35 00B4  82 EB 20 A4 AE E1 E2+  msg_out_of_range DB 'Вы достигли лимита количества запусков
программы(10)!$'
36       A8 A3 AB A8 20 AB A8+
37       AC A8 E2 A0 20 AA AE+
38       AB A8 E7 A5 E1 E2 A2+
39       A0 20 A7 A0 AF E3 E1+
40       AA AE A2 20 AF E0 AE+
41       A3 E0 A0 AC AC EB 28+
42       31 30 29 21 24
43 00EA  87 A0 A2 A5 E0 E8 A5+  msg_exit DB 'Завершение программы. Всего доброго!$'
44       AD A8 A5 20 AF E0 AE+
45       A3 E0 A0 AC AC EB 2E+
46       20 82 E1 A5 A3 AE 20+
47       A4 AE A1 E0 AE A3 AE+
48       21 24
49 010F                data_seg ENDS
50
51                 ;Вывод одного символа
52 0000                putch PROC
53 0000  B4 02           MOV AH, 02h
```

```
 54 0002  CD 21         INT 21h
 55 0004  C3           RET
 56 0005              putch ENDP
 57
```



```
 58              ;Вывод строки
 59 0005              print_str proc
 60 0005  B4 09        MOV AH, 09h
 61 0007  CD 21         INT 21h
 62 0009  C3           RET
 63 000A              print_str endp
 64
 65              ;Ввод символа
 66 000A              getch PROC
 67 000A  B4 01        MOV AH, 01H
 68 000C  CD 21         INT 21H
 69 000E  C3           RET
 70 000F              getch ENDP
 71
 72              ;Перевод строки и возврат каретки
 73 000F              clrf PROC
 74 000F  B2 0A        MOV DL, 0AH
 75 0011  E8 FFEC          CALL putch
 76 0014  B2 0D        MOV DL, 0DH
 77 0016  E8 FFE7          CALL putch
 78 0019  C3           RET
 79 001A              clrf ENDP
 80
 81              ;Очитска экрана при завершении работы
 82 001A              clear_screen PROC
 83 001A  B4 00        MOV AH, 00H
 84 001C  B0 03        MOV AL,3
 85 001E  CD 10         INT 10H
 86 0020  B9 0019          MOV CX,  25
 87 0023              labclr:
 88 0023  B4 02           MOV  AH , 02H
 89 0025  B2 0A           MOV  DL , 10
 90 0027  CD 21           INT 021H
 91 0029  E2 F8        LOOP labclr
 92 002B  C3           RET
 93 002C              clear_screen ENDP
 94
 95              ;Вывод символа в шестнадцатеричной кодировке
 96 002C              HEX PROC
 97 002C  52           PUSH DX
 98 002D  8A C2         MOV AL, DL
 99 002F  D0 E8 D0 E8 D0 E8 D0+   SHR AL, 4
100       E8
101 0037  BB 0000r          LEA BX, table_hex
102 003A  D7           XLAT
103 003B  8A D0         MOV DL, AL
104 003D  E8 FFC0          CALL putch
105 0040  5A           POP DX
106 0041  8A C2         MOV AL, DL
107 0043  24 0F        AND AL, 0FH
108 0045  D7           XLAT
109 0046  8A D0         MOV DL, AL
```

```
 110 0048  E8 FFB5            CALL putch
 111                ;MOV DL, 'h'
 112                ;CALL putch
 113 004B  C3              RET
 114 004C                HEX ENDP
```



```
 115
 116                ;Выход из программы
 117 004C                exit_prog PROC
 118                ;Очистка экрана
 119 004C  E8 FFCB           CALL clear_screen
 120
 121                ;Сообщение, что пользователь достиг лимита ввода строк
 122 004F  A0 0024r          MOV AL, flag_out
 123 0052  3C 01          CMP AL, 1
 124 0054  75 09          JNE flag4
 125
 126 0056  BA 00B4r           LEA DX, msg_out_of_range
 127 0059  E8 FFA9           CALL print_str
 128 005C  E8 FFB0           CALL clrf
 129
 130 005F                flag4:
 131                ;Сообщение о выходе из программы
 132 005F  BA 00EAr           LEA DX, msg_exit
 133 0062  E8 FFA0           CALL print_str
 134 0065  E8 FFA7           CALL clrf
 135
 136 0068  B0 00          mov al, 00
 137 006A  B4 4C           mov ah, 4ch
 138 006C  CD 21          int 021h
 139 006E                exit_prog ENDP
 140
 141 006E                START:
 142                ;Подключение сегментов
 143 006E  B8 0000s          MOV AX, data_seg
 144 0071  8E D8          MOV DS, AX
 145
 146                ;Очистка экрана
 147 0073  E8 FFA4           CALL clear_screen
 148
 149                ;Приветствие пользователя
 150 0076  BA 0025r           LEA DX, msg_welcome
 151 0079  E8 FF89           CALL print_str
 152 007C  E8 FF90           CALL clrf
 153
 154                ;Подсказка для пользователя по символу $
 155 007F  BA 0037r           LEA DX, msg_hint
 156 0082  E8 FF80           CALL print_str
 157 0085  B2 24          MOV DL, '$'
 158 0087  E8 FF76           CALL putch
 159 008A  E8 FF82           CALL clrf
 160
 161                ;Внешний цикл, ввод строк
 162 008D  B9 000A           MOV CX, 10
 163 0090                cycle_main:
 164
```

```
165 0090  51              PUSH CX
166
167                 ;Приглашение к вводу строк
168 0091  BA 0064r           LEA DX, msg_input
169 0094  E8 FF6E            CALL print_str
170 0097  E8 FF75            CALL clrf
171
```



```
172                 ;Ввод и буферизация строки символов
173 009A  1E              PUSH DS
174 009B  07              POP ES
175 009C  BF 0010r           LEA DI , buff
176
177 009F  B9 0014            MOV CX, 20
178 00A2                cycle_input:
179 00A2  E8 FF65            CALL getch
180 00A5  AA              STOSB
181
182 00A6  3C 24           CMP AL, '$'
183 00A8  74 02           JE flag1
184 00AA  E2 F6           LOOP cycle_input
185
186                 ;Вывод промежуточного знака =
187 00AC                flag1:
188 00AC  B2 3D           MOV DL, '='
189 00AE  E8 FF4F            CALL putch
190
191                 ;Вывод и перевод символа в шестнадцатеричную систему
192 00B1  1E              PUSH DS
193 00B2  07              POP ES
194 00B3  BE 0010r           LEA SI , buff
195
196 00B6  B9 0014            MOV CX, 20
197 00B9                cycle_output:
198 00B9  AC              LODSB
199
200 00BA  3C 24           CMP AL, '$'
201 00BC  74 0C           JE flag2
202
203 00BE  8A D0           MOV DL, AL
204 00C0  E8 FF69            CALL HEX
205 00C3  B2 20           MOV DL, ' '
206 00C5  E8 FF38            CALL putch
207 00C8  E2 EF           LOOP cycle_output
208
209 00CA                flag2:
210 00CA  E8 FF42            CALL clrf
211
212                 ;Спрашиваем, надо ли повторно запустить программу
213 00CD  BA 008Dr           LEA DX, msg_help_exit
214 00D0  E8 FF32            CALL print_str
215
216 00D3  E8 FF34            CALL getch
217 00D6  3C 2A           CMP AL, '*'
218
219 00D8  E8 FF34            CALL clrf
```

```
   220 00DB  E8 FF31              CALL clrf
   221
   222 00DE  74 08            JE flag3
   223
   224 00E0  59               POP CX
   225 00E1  E2 AD            LOOP cycle_main
   226
   227                ;Устанавливаем флаг, что пользователь достиг лимита
   228 00E3  C6 06 0024r 01         MOV flag_out, 1
```



```
   229
   230 00E8               flag3:
   231 00E8  E8 FF61          CALL exit_prog
   232
   233 00EB               mycode ENDS
   234                END START
```



```
Symbol Name     Type   Value              Cref (defined at #)

??DATE          Text   "04/07/24"
??FILENAME      Text   "lab5_1  "
??TIME          Text   "16:05:38"
??VERSION       Number 030A
@CPU            Text   0101H
@CURSEG         Text   MYCODE            #1  #7  #49
@FILENAME       Text   LAB5_1
@WORDSIZE       Text   2              #1  #7  #49
BUFF            Byte   DATA_SEG:0010          #11  175  194
CLEAR_SCREEN        Near   MYCODE:001A        #82  119  147
CLRF            Near   MYCODE:000F        #73  128  134  152  159  170  210  219  220
CYCLE_INPUT     Near   MYCODE:00A2        #178  184
CYCLE_MAIN      Near   MYCODE:0090        #163  225
CYCLE_OUTPUT        Near   MYCODE:00B9        #197  207
EXIT_PROG       Near   MYCODE:004C        #117  231
FLAG1           Near   MYCODE:00AC        183  #187
FLAG2           Near   MYCODE:00CA        201  #209
FLAG3           Near   MYCODE:00E8        222  #230
FLAG4           Near   MYCODE:005F        124  #130
FLAG_OUT        Byte   DATA_SEG:0024          #12  122  228
GETCH           Near   MYCODE:000A        #66  179  216
HEX         Near   MYCODE:002C        #96  204
LABCLR          Near   MYCODE:0023        #87  91
MSG_EXIT        Byte   DATA_SEG:00EA          #43  132
MSG_HELP_EXIT       Byte   DATA_SEG:008D          #29  213
MSG_HINT        Byte   DATA_SEG:0037          #16  155
MSG_INPUT       Byte   DATA_SEG:0064          #23  168
MSG_OUT_OF_RANGE    Byte   DATA_SEG:00B4          #35  126
MSG_WELCOME     Byte   DATA_SEG:0025          #13  150
PRINT_STR       Near   MYCODE:0005        #59  127  133  151  156  169  214
PUTCH           Near   MYCODE:0000        #52  75  77  104  110  158  189  206
START           Near   MYCODE:006E        #141  234
```

```
TABLE_HEX       Byte   DATA_SEG:0000          #8  101

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

DATA_SEG        16  010F Para   none          2  #7  143
MYCODE          16  00EB Para   none    CODE      #1  2
```

Текст программы:

```
mycode SEGMENT 'CODE'
ASSUME CS: mycode, DS: data_seg

; Бирюкова Е.И ИУ5-41Б
; Вариант 3 ЛР5

data_seg SEGMENT
   table_hex DB '0123456789ABCDEF'
   buff DB 20 dup (' ')
   flag_out DB 0
   msg_welcome DB 'Добро пожаловать!$'
   msg_hint DB 'Для завершения ввода строки введите символ: $'
   msg_input DB 'Введите строку(длиной до 20 символов!): $'
   msg_help_exit DB 'Для завершения программы введите “*”: $'
   msg_out_of_range DB 'Вы достигли лимита количества запусков программы(10)!$'
   msg_exit DB 'Завершение программы. Всего доброго!$'
data_seg ENDS

; Вывод одного символа
putch PROC
   MOV AH, 02h
   INT 21h
   RET
putch ENDP

;Вывод строки
print_str proc
   MOV AH, 09h
   INT 21h
   RET
print_str endp

;Ввод символа
getch PROC
   MOV AH, 01H
   INT 21H
   RET
getch ENDP

;Перевод строки и возврат каретки
clrf PROC
   MOV DL, 0AH
   CALL putch
   MOV DL, 0DH
   CALL putch
   RET
clrf ENDP
```

```
;Очистка экрана при завершении работы
clear_screen PROC
   MOV AH, 00H
   MOV AL,3
   INT 10H
   MOV CX,  25
   labclr:
      MOV  AH , 02H
      MOV  DL , 10
      INT 021H
   LOOP labclr
   RET
clear_screen ENDP

;Вывод символа в шестнадцатеричной кодировке
HEX PROC
    PUSH DX
    MOV AL, DL
    SHR AL, 4
    LEA BX, table_hex
    XLAT
    MOV DL, AL
    CALL putch
    POP DX
    MOV AL, DL
    AND AL, 0FH
    XLAT
    MOV DL, AL
    CALL putch
    ;MOV DL, 'h'
    ;CALL putch
    RET
HEX ENDP

;Выход из программы
exit_prog PROC
   ;Очистка экрана
   CALL clear_screen

   ;Сообщение, что пользователь достиг лимита ввода строк
   MOV AL, flag_out
   CMP AL, 1
   JNE flag4

   LEA DX, msg_out_of_range
   CALL print_str
   CALL clrf

   flag4:
   ;Сообщение о выходе из программы
   LEA DX, msg_exit
   CALL print_str
   CALL clrf
```

```
    mov al, 00
    mov ah, 4ch
    int 021h
exit_prog ENDP

START:
    ;Подключение сегментов
    MOV AX, data_seg
    MOV DS, AX

    ;Очистка экрана
    CALL clear_screen

    ;Приветствие пользователя
    LEA DX, msg_welcome
    CALL print_str
    CALL clrf

    ;Подсказка для пользователя по символу $
    LEA DX, msg_hint
    CALL print_str
    MOV DL, '$'
    CALL putch
    CALL clrf

    ;Внешний цикл ввода строк
    MOV CX, 10
    cycle_main:

        PUSH CX

        ;Приглашение к вводу строки
        LEA DX, msg_input
        CALL print_str
        CALL clrf

        ;Ввод и буферизация строки символов
        PUSH DS
        POP ES
        LEA DI , buff

        MOV CX, 20
        cycle_input:
            CALL getch
            STOSB

            CMP AL, '$'
            JE flag1
        LOOP cycle_input

        ;Вывод промежуточного знака =
        flag1:
        MOV DL, '='
        CALL putch
```

```
    ;Вывод и перевод символа в шестнадцатеричную систему
    PUSH DS
    POP ES
    LEA SI , buff

    MOV CX, 20
    cycle_output:
      LODSB

      CMP AL, '$'
      JE flag2

      MOV DL, AL
      CALL HEX
      MOV DL, ' '
      CALL putch
    LOOP cycle_output

    flag2:
    CALL clrf

    ;Спрашиваем, надо ли повторно запустить программу
    LEA DX, msg_help_exit
    CALL print_str

    CALL getch
    CMP AL, '*'

    CALL clrf
    CALL clrf

    JE flag3

    POP CX
  LOOP cycle_main

  ;Устанавливаем флаг, что пользователь достиг лимита
  MOV flag_out, 1

  flag3:
  CALL exit_prog

mycode ENDS
END START
```

7. Результаты работы программы

Начало работы программы и очистка экрана:

```
Добро пожаловать!
Для завершения ввода строки введите символ: $
Введите строку(длиной до 20 символов!):
```

Ввод строки и запрос ввода новой:

```
Добро пожаловать!
Для завершения ввода строки введите символ: $
Введите строку(длиной до 20 символов!):
АБВГДЕ$=80 81 82 83 84 85
Для завершения программы введите "*": _
```

Повторный ввод строки:

```
Добро пожаловать!
Для завершения ввода строки введите символ: $
Введите строку(длиной до 20 символов!):
АБВГДЕ$=80 81 82 83 84 85
Для завершения программы введите "*": o

Введите строку(длиной до 20 символов!):
АБВГД$=80 81 82 83 84
Для завершения программы введите "*":
```

Завершение программы по символу ‘*’ и очистка экрана:

```
Завершение программы. Всего доброго!

V:\TASM\TASM3>
```

Принудительно завершение программы и очистка экрана:

```
Вы достигли лимита количества запусков программы(10)!
Завершение программы. Всего доброго!

V:\TASM\TASM3>_
```

8. Выводы по ЛР № 5

В результате выполнения данной лабораторной работы была разработана и отлажена программа на языке Ассемблер, которая позволяет осуществлять ввод строки символов с клавиатуры, буферизировать её, а затем последовательно выводить на экран в

шестнадцатеричном представлении. Для корректной работы программы было предусмотрено запоминание строки символов в байтовом массиве.

Программа организована с помощью вложенных циклов: цикла ввода символов (с помощью команд условного перехода), цикла вывода шестнадцатеричного представления и общего цикла ввода строк (с использованием команды LOOP). Ввод строки может быть завершен по вводу специального символа '*'. Была организована процедура HEX для перевода символов в шестнадцатеричное представление и процедура CLRSCR для очистки экрана.

В ходе выполнения данной лабораторной работы было показано, как можно эффективно использовать вложенные циклы, условные переходы, процедуры и другие конструкции языка Ассемблер для решения задачи ввода и вывода строк символов на экран.