Защищено:
Большаков С.А.

"__"_____2020 г.

Демонстрация ЛР:
Большаков С.А.

"__"_____2020 г.

Отчет по лабораторной работе №5 по курсу Системное программирование

"Ввод и вывод в машинном формате"

(есть ли дополнительные требования - НЕТ)

8
(количество листов)
Вариант № **1**

ИСПОЛНИТЕЛЬ:

студент группы **ИУ5Ц-62** _____
(подпись)

Гусев С.Р.

"__"_____2020 г.

Москва, МГТУ - 2020

СОДЕРЖАНИЕ

1. Цель выполнения лабораторной работы № 5

Разработать и отладить программу на языке Ассемблер для ввода строки символов с клавиатуры (последовательности символов) и последовательного вывода их в шестнадцатеричном представлении на экран (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

Студенты разрабатывают работоспособную программу на языке Ассемблер по заданию ЛР, выполняют следующие действия (порядок выполнения работы):

• Знакомятся и осмысливают задание на ЛР.

• Разрабатывают алгоритм реализации задачи (блок-схема программы можно оформить в MS VISIO, MS WORD или на листе бумаги).

• Выполняют написание текста программы на языке Ассемблер и вводят его в отдельном текстовом редакторе (можно использовать текстовый редактор ASM_ED.EXE – есть на сайте) или в интегрированной оболочке (например, в QC).

• Выполняют отладку программы в отладчике (TD.EXE), демонстрируют преподавателю умение работать в отладчике, выполняя различные действия (выполнение по шагам, просмотр данных и т.д.).

• Формируют исполнимый модуль программы заданного типа (COM или EXE см. задание).

• Демонстрируют преподавателю работоспособную программу.

• По требованию преподавателя, если нужно, вносят изменения в программу и демонстрируют знание действий необходимых для создания исполнимого модуля (это предварительная сдача ЛР).

• Оформляют отчет по данной лабораторной работе в соответствии с требованиями приведенными ниже и на основе шаблона отчетов по ЛР.

• На основе отчета по ЛР (распечатанного) выполняют защиту ЛР у преподавателя (ответы на контрольные вопросы), после чего в журнале отмечается: срок сдачи ЛР, срок защиты ЛР, оценка за защиту данной ЛР и выполнение дополнительных требований к ЛР. На защите задаются вопросы, перечисленные в разделе “Контрольные вопросы по каждой ЛР и общие вопросы”, а также вопросы по листингу программы (отметьте себе, не по тексту программы, по листингу).

Работа считается выполненной полностью и в срок, если студент полностью сдал и защитил отчет ЛР в срок. Если студент сделал работу с дополнительными требованиями, то это обязательно отмечается в журнале ЛР и учитывается в оценке при подведении итогов семестра по данной дисциплине и на экзамене. Если студент выполнил все ЛР с дополнительными требованиями и получил отметки не ниже “хорошо”, то на зачете он освобождается от решения задачи (задачи на зачете заключаются в написании процедуры на языке Ассемблер или командного файла) и может претендовать на получение автоматической оценки по курсовой работе - ОТЛИЧНО, при своевременной ее сдаче.

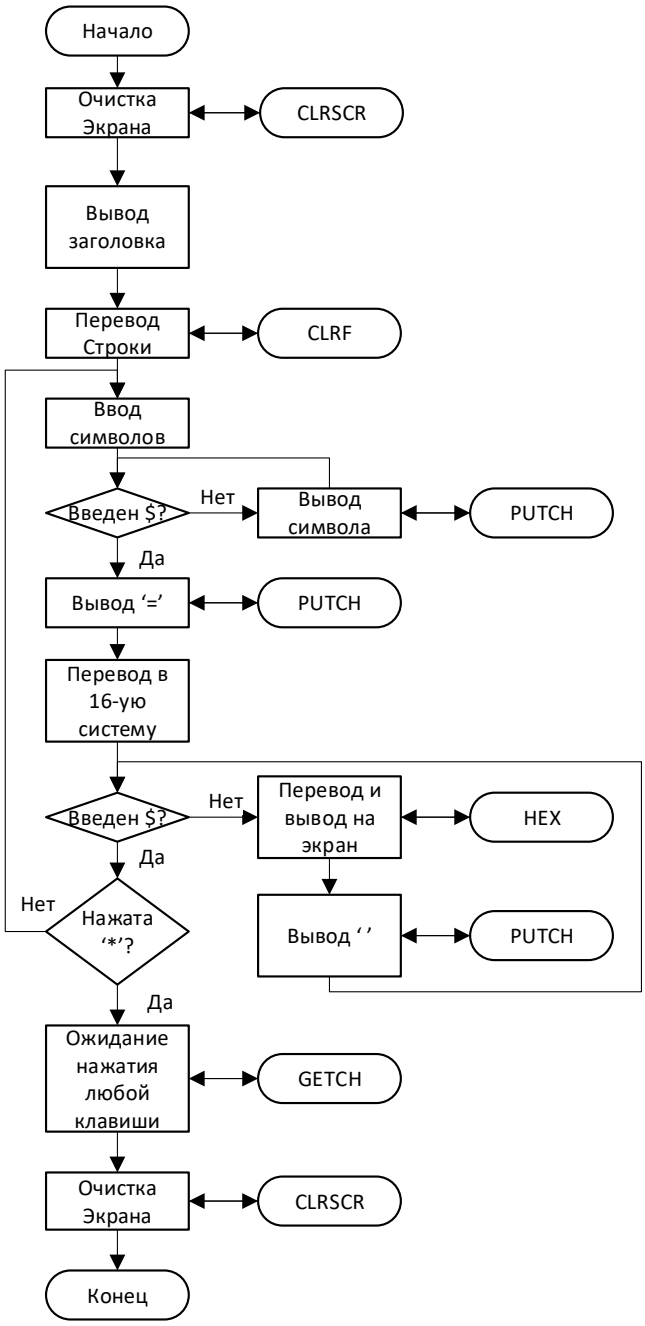
Если преподаватель обнаруживает (поверьте, сделать очень просто), что программа и отчет по ней сделаны несамостоятельно (проще - списаны), то он отмечает данный факт в журнале, а на зачете в этом случае задаются дополнительные вопросы по лабораторным работам, методическому пособию и материалам лекций. Кроме того, студент в этом случае не вправе рассчитывать на оценку по курсовой работе выше чем – удовлетворительно.

Для выполнения цикла лабораторных работ по курсу полезно познакомиться с указанными выше разделами методических указаний к ЛР, подготовленных преподавателем (отдельный документ – есть на сайте – оранжевая кнопка).

3. Описание ошибок, возникших при отладке № 5

№ п/п	Проявление ошибки	Причина ошибки	Способ устранения
1.	Ошибка компиляции	Неправильное использование оператора ‘je’	Использовать оператор правильно
2.	Зависание программы, бесконечный вывод в консоль	Неправильно указано количество итераций цикла	Занести верное количество итераций цикла в регистр cx

4. Блок-схема программы



5. Текст программы на языке Ассемблера


```
 1    0000                        MYCODE SEGMENT 'CODE'
 2                                        ASSUME CS:MYCODE, DS:MYCODE
 3
 4    0000  0D 0A 45 6E 74 65 72+ mes0 DB 13,10,'Enter string: $'
 5          20 73 74 72 69 6E   67+
 6          3A 20 24
 7    0011  0D 0A 50 72 65 73 73+ mes1 DB 13,10,'Press any key to restart...      Press *   to exit... $'
 8          20 61 6E 79 20 6B   65+
 9          79 20 74 6F 20 72   65+
10          73 74 61 72 74 2E   2E+
11          2E 20 50 72 65 73   73+
12          20 2A 20 74 6F 20   65+
13          78 69 74 2E 2E 2E   20+
14          24
15    0043  30 31 32 33 34 35   36+ HEX_STRING      DB '0123456789ABCDEF'
16          37 38 39 41 42 43   44+
17          45 46
18    0053  0100*(00)             mystr db 256 dup(0) ;mystr      db' $'
19
20    0153                        CLRF PROC  ;perevod stroki
21    0153  50                    PUSH AX
22    0154  B4 02                       MOV AH, 02H
23    0156  B2 0D                       MOV DL, 0dh
24    0158  CD 21                       INT 21H
25    015A  B4 02                       MOV AH, 02H
26    015C  B2 0A                       MOV DL, 0ah
27    015E  CD 21                       INT 21H
28    0160  58                    POP AX
29    0161  C3                        RET
30    0162                        CLRF ENDP
31
32    0162                        GETCH PROC
33    0162  B4 01                       MOV AH, 01H
34    0164  CD 21                       INT 21H
35    0166  C3                        RET
36    0167                        GETCH ENDP
37
38    0167                        PUTCH PROC
39    0167  B4 02                       MOV AH, 02H
40    0169  CD 21                       INT 21H
41    016B  C3                        RET
42    016C                        PUTCH ENDP
43
44    016C                        CLRSCR PROC
45    016C  B8 0003                     MOV AX, 03H
46    016F  CD 10                       INT 10H
47    0171  C3                        RET
48    0172                        CLRSCR ENDP
49
50    0172                        HEX PROC; procedura perevoda v 16 ss
51    0172  53                   PUSH BX
52    0173  BB 0043r                    MOV BX, offset HEX_STRING
53    0176  8B C2                       MOV AX,DX
54    0178  50                    PUSH AX
55    0179  D0 E8 D0 E8 D0 E8          D0+          SHR AL,4
56          E8
57    0181  D7                          XLAT
```

```
 58  0182  8A D0                    MOV DL,AL
 59  0184  E8 FFE0                  CALL PUTCH
 60  0187  58                POP AX
 61  0188  24 0F                    AND AL,00001111b
 62  018A  D7                       XLAT
 63  018B  8A D0                    MOV DL,AL
 64  018D  E8 FFD7                  CALL PUTCH
 65  0190  5B                       POP BX
 66  0191  C3                       RET
 67  0192                    HEX ENDP;
 68
 69  0192                    START:
 70                          ; ђ—ђ°ђi‘Ъ‘ґђ·ђєђ°   ‘ҐђµђiђjђµђS‘,ђSђsђiђs ‘Ъђµђiђë‘Ґ‘,‘Ъђ° ђrђ°ђSђS‘‹‘… ds
 71  0192  0E                       PUSH CS
 72  0193  1F                       POP  DS
 73  0194  E8 FFD5                  CALL CLRSCR
 74  0197  BA 0000r                 MOV DX, OFFSET mes0
 75  019A  B4 09                    MOV AH, 09H
 76  019C  CD 21                    INT 21H
 77
 78  019E                           Loop0:
 79  019E  E8 FFCB                  CALL CLRSCR
 80  01A1  E8 FFAF                  CALL CLRF
 81  01A4  BB 0053r                 LEA BX,mystr
 82  01A7  BB 0000                  MOV BX,0
 83  01AA                           Loop1:
 84  01AA                                LoopCh:
 85  01AA  E8 FFB5                       CALL GETCH
 86  01AD  3C 21                              CMP AL,21h
 87  01AF  72 F9                              JB   LoopCh
 88  01B1  88 07                         MOV [BX],AL
 89  01B3  3C 24                         CMP AL, '$'
 90  01B5  75 03                         JNE ENIF
 91  01B7  EB 03 90                      JMP ENDEF
 92  01BA                                     ENIF:
 93  01BA  8A D0                                   MOV DL,AL
 94  01BC                                     ENDEF:
 95  01BC  43                                      INC BX
 96  01BD  47                                      INC DI
 97  01BE  3C 24                                   CMP AL,'$'
 98  01C0  75 E8                                   JNE Loop1
 99  01C2  B2 3D                                   MOV DL, '='
100  01C4  E8 FFA0                                 CALL PUTCH
101  01C7  BB 0000                                 MOV BX, 0
102  01CA                           Loop2:
103  01CA  8A 17                         MOV DL,[BX]
104  01CC  52                            PUSH DX
105  01CD  80 FA 24                      CMP DL, '$'
106  01D0  75 03                         JNE Hexing
107  01D2  EB 27 90                      JMP G1
108  01D5                                     Hexing:
109  01D5  E8 FF9A                                 CALL HEX
110  01D8  43                                      INC BX
111                                                ;MOV DL, '   '
112                                                ;CALL PUTCH
113  01D9  B2 48                                   MOV DL, 'H'
114  01DB  E8 FF89                                 CALL PUTCH
```

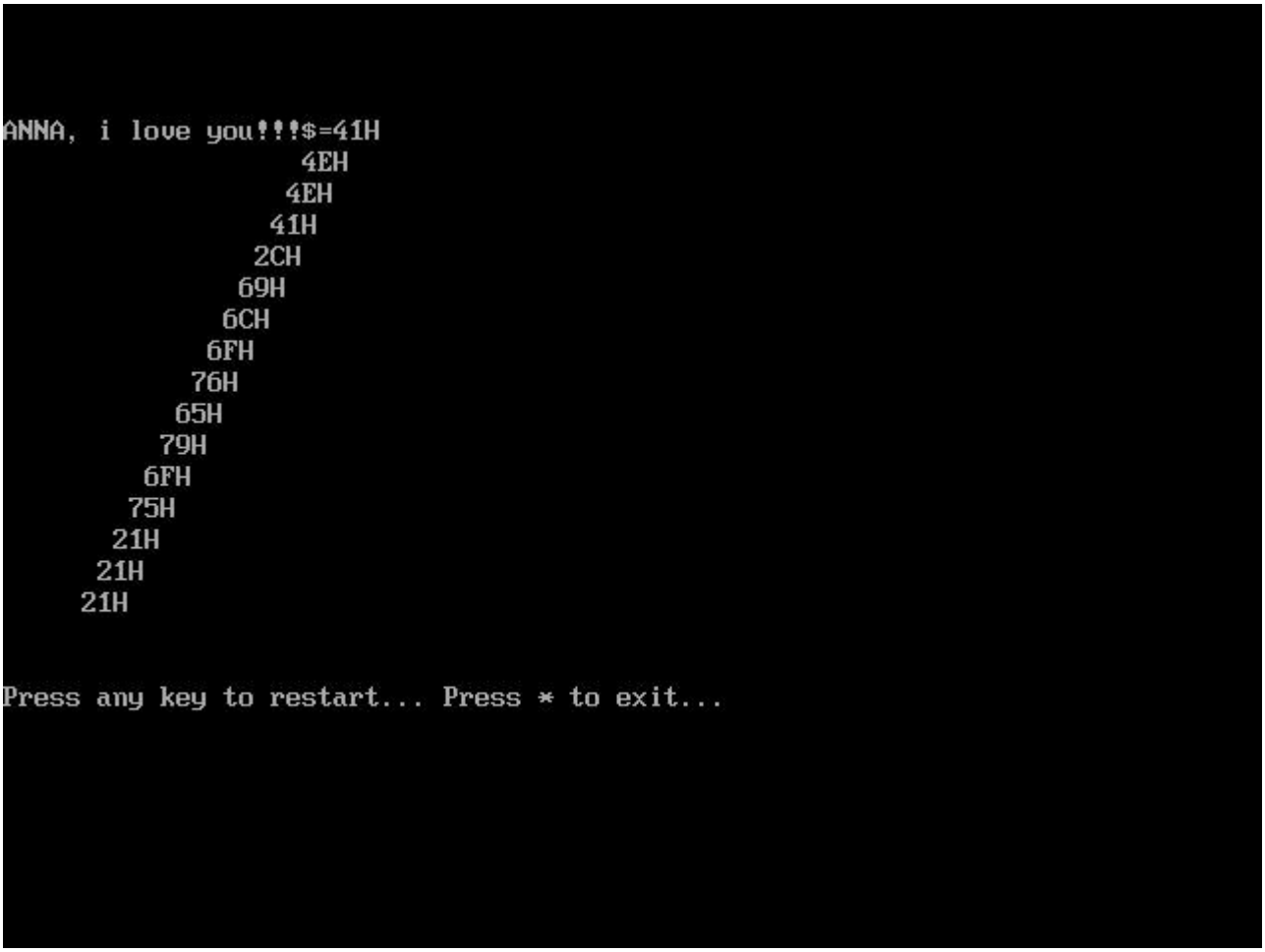
```
115  01DE  53                                    PUSH BX
116  01DF  50                                    PUSH AX
117  01E0  51                                    PUSH CX
118  01E1  B4 02                                 MOV AH, 02H
119  01E3  B2 0A                                 MOV DL, 10
120  01E5  CD 21                                 INT 21H
121  01E7  BB 0000                               MOV BX, 0
122  01EA  03 DF                                 ADD BX, DI
123  01EC  83 EF 01                              SUB DI, 1
124  01EF  8B CB                                 MOV CX, BX
125  01F1                                        LOOP5:
126  01F1  B2 20                                         MOV DL, ' '
127  01F3  E8 FF71                                       CALL PUTCH
128  01F6  E2 F9                                 LOOP LOOP5
129
130  01F8  59                                    POP CX
131  01F9  58                                    POP AX
132  01FA  5B                                    POP BX
133
134  01FB                                  G1:
135                                              ;POP DX
136  01FB  80 FA 24                              CMP DL, '$'
137  01FE  75 CA                                 JNE Loop2
138
139  0200  E8 FF50                       CALL CLRF
140                                      ; ђhђ¶ђёђrђ°ђSђёђµ  ђ·ђ°ђIђµ‘Ђ‘€ђµђSђё‘Ц ђї‘Ђђsђi‘Ђђ°ђjђj‘‹
141  0203  BA 0011r                      MOV DX, OFFSET mes1
142  0206  B4 09                         MOV AH, 09H
143  0208  CD 21                         INT 21H
144  020A  E8 FF55                       CALL GETCH
145  020D  3C 2A                         CMP AL, '*'
146  020F  74 02                                 JE  FIN
147  0211  E2 8B                         LOOP Loop0
148
149  0213                                FIN:
150                                      ; ђ’‘‹…ђsђr ђёђ· ђї‘Ђђsђi‘Ђђ°ђjђj‘‹
151  0213  E8 FF56                               CALL CLRSCR
152  0216  B0 00                                 MOV AL, 0
153  0218  B4 4C                                 MOV AH, 4CH
154  021A  CD 21                                 INT 21H
155  021C                                MYCODE ENDS
156                                      END START
```

Symbol Table

Symbol Name	Type	Value	Cref (defined at #)
??DATE	Text	"05/02/20"	
??FILENAME	Text	"LR5 "	
??TIME	Text	"18:43:48"	
??VERSION	Number	030A	
@CPU	Text	0101H	
@CURSEG	Text	MYCODE	#1
@FILENAME	Text	LR5	
@WORDSIZE	Text	2	#1
CLRF	Near	MYCODE:0153	#20 80 139
CLRSCR	Near	MYCODE:016C	#44 73 79 151
ENDEF	Near	MYCODE:01BC	91 #94
ENIF	Near	MYCODE:01BA	90 #92
FIN	Near	MYCODE:0213	146 #149
G1	Near	MYCODE:01FB	107 #134
GETCH	Near	MYCODE:0162	#32 85 144
HEX	Near	MYCODE:0172	#50 109
HEXING	Near	MYCODE:01D5	106 #108
HEX_STRING	Byte	MYCODE:0043	#15 52
LOOP0	Near	MYCODE:019E	#78 147
LOOP1	Near	MYCODE:01AA	#83 98
LOOP2	Near	MYCODE:01CA	#102 137
LOOP5	Near	MYCODE:01F1	#125 128
LOOPCH	Near	MYCODE:01AA	#84 87
MES0	Byte	MYCODE:0000	#4 74
MES1	Byte	MYCODE:0011	#7 141
MYSTR	Byte	MYCODE:0053	#18 81
PUTCH	Near	MYCODE:0167	#38 59 64 100 114 127
START	Near	MYCODE:0192	#69 156

Groups & Segments	Bit Size	Align	Combine	Class	Cref (defined at #)
MYCODE	16	021C	Para	none	CODE #1 2 2

6. Результаты работы программы



7. Выводы по ЛР № 5

В ходе этой лабораторной работы я научился работать с вводом символов и строк в ассемблере, а также получать коды символов с помощью заданной таблицы и оператора ‘xlat’.