

Защищено:
Большаков С.А.

"__" _____ 2020 г.

Демонстрация ЛР:
Большаков С.А.

"__" _____ 2020 г.

Отчет по лабораторной работе №4 по курсу Системное программирование

"Циклы и перевод символов"

(есть ли дополнительные требования - НЕТ)

10
(количество листов)
Вариант № 1

ИСПОЛНИТЕЛЬ:

студент группы **ИУ5Ц-62**

Гусев С. Р.

(подпись)

"__" _____ 2020 г.

СОДЕРЖАНИЕ

1. Цель выполнения лабораторной работы № 4.....	3
2. Порядок и условия проведения работы № 4	3
3. Описание ошибок, возникших при отладке № 4.....	4
4. Блок-схема программы	4
5. Текст программы на языке Ассемблера	5
6. Результаты работы программы.....	10
7. Выводы по ЛР № 4	10

1. Цель выполнения лабораторной работы № 4

Комплекс лабораторных работ (3-9 ЛР) по языку Ассемблер выполняется студентами для освоения языка программирования, получения навыков разработки и отладки программ на нем, изучения и использования компонентов системы программирования Ассемблер (компилятора, редактора связей, отладчика) и получения навыков оформления документации по программным разработкам, реализуемым на языке Ассемблера.

Лабораторные работы взаимосвязаны друг с другом, поэтому их необходимо выполнять последовательно, начиная с 3-й ЛР СП (первые две ЛР курса посвящены другим темам). Это позволяет значительно упростить задачу выполнения всего цикла лабораторных работ. При этом в новую работу могут быть успешно включены проработки и отлаженные фрагменты из предыдущей работы (процедуры и циклы). Кроме того, самостоятельное выполнение цикла лабораторных работ позволит студентам успешно справиться с заданием на курсовую работу, которая выполняется также в данном семестре (4-й семестр). В курсовой работе студенты на языке Ассемблера разрабатывают резидентную программу.

2. Порядок и условия проведения работы № 4

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 самостоятельно (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.

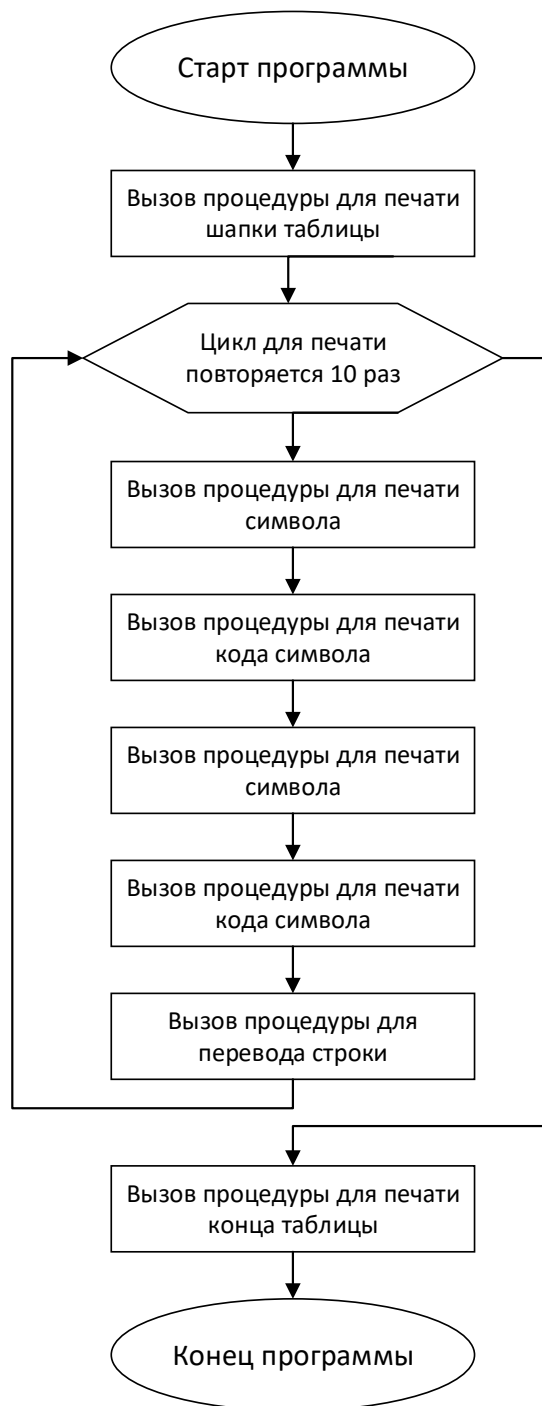
Для выполнения цикла лабораторных работ по курсу полезно познакомиться с указанными выше разделами методических указаний к ЛР, подготовленных преподавателем (отдельный документ – есть на сайте – оранжевая кнопка).

3. Описание ошибок, возникших при отладке № 4

№ 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

(на следующей странице)



5. Текст программы на языке Ассемблера

Turbo Assembler Version 3.1
V:\LR\LR4\LR4.asm

04/30/20 17:56:30

Page 1

```

1          ;Гусев Сергей ИУ5Ц-626
2
3  0000          MYCODE segment 'CODE'
4                  assume cs:MYCODE, ds:MYCODE
5
6  0000  30 31 32 33 34 35  36+    HEX_STRING      DB '0123456789ABCDEF' ;    выборка
кодировки
7      37 38 39 41 42 43    44+
```

```

8      45 46
9      0010 A2 A2 A5 A4 A8 E2      A5+      startStr db 'введите символ для начала работы программы'
10     20 E1 A8 AC A2 AE AB+
11     20 A4 AB EF 20 AD A0+
12     E7 A0 AB A0 20 E0 A0+
13     A1 AE E2 EB 20 AF E0+
14     AE A3 E0 A0 AC AC EB
15     003A 84 AB EF 20 A2 EB      E5+      continueStr db 'Для выхода нажмите "q"$'
16     AE A4 A0 20 AD A0 A6+
17     AC A8 E2 A5 20 22 71+
18     22 24
19
20     0051      start:
21                ;Загрузка сегментного регистра данных DS
22     0051 0E      push CS
23     0052 1F      pop DS
24     0053 BB 0000r      mov bx, offset HEX_STRING
25
26     0056      main:
27                ;Дополнительное требование: очистка экрана
28     я
29     0056 E8 0069      call clrscr;
30
31                ;Вывод строки-подсказки о том, что надо ввести букву
32     0059 BA 0010r      mov dx, offset startStr
33     005C E8 0049      call putst
34     005F E8 0050      call clrf
35
36                ; Запрос на ввод символа
37     0062 E8 0058      call getch
38     0065 50      push ax
39
40                ; Циклический вывод букв на экран
41     0066 B9 000D      mov cx, 13
42     0069      cycle:
43
44                ; Ввод буквы
45     0069 58      pop ax
46     006A 50      push ax
47     006B 8A D0      mov dl, al
48     006D 50      push ax
49     006E E8 003C      call putch
50
51                ; Type
52     0071 BA 0020      mov dx, 32
53     0074 E8 0036      call putch
54     0077 BA 00CD      mov dx, 205
55     007A E8 0030      call putch
56     007D BA 0020      mov dx, 32
57     0080 E8 002A      call putch

```

```
58
59                                ; Вывод hex
60 0083 58                        pop ax
61 0084 E8 0042                  call hex
62 0087 58                        pop ax
63
64                                ; Increment буквы
65 0088 FE C0                     inc al
66 008A 50                        push ax
67
68 008B E2 DC                     loop cycle
69
70
71                                ; Запрос на продолжение программы
72 008D BA 003Ar                  mov dx, offset continueStr
73 0090 E8 0015                  call putst
74 0093 E8 001C                  call clrf
75 0096 E8 0024                  call getch
76 0099 3C 71                    cmp al, 'q'
77 009B 74 02                    je exit
78 009D EB B7                    jmp main
79
80 009F                            exit:
81                                ; Очистка экрана
82 009F E8 0020                  call clrscr;
83
84                                ; Выход из программы
85 00A2 B0 00                    mov al, 0
86 00A4 B4 4C                    mov ah, 4ch
87 00A6 CD 21                    int 021h
88
89                                ; Процедура - вывод строки на экран
90 00A8                            putst proc
91 00A8 B4 09                    mov ah, 09h
92 00AA CD 21                    int 021h
93 00AC C3                        ret
94 00AD                            putst endp
95
96                                ;Процедура - вывод символа
97 00AD                            putch proc
98 00AD B4 02                    mov ah, 02h
99 00AF CD 21                    int 021h
100 00B1 C3                        ret
101 00B2                            putch endp
102
103                                ; Процедура - перевод строки
104 00B2                            clrf proc
105 00B2 B2 0A                    mov dl, 10
106 00B4 E8 FFF6                  call putch
107 00B7 B2 0D                    mov dl, 13
108 00B9 E8 FFF1                  call putch
109 00BC C3                        ret
110 00BD                            clrf endp
111
112                                ; Процедура - ввод символа
113 00BD                            getch proc
114 00BD B4 08                    mov ah, 08h
```

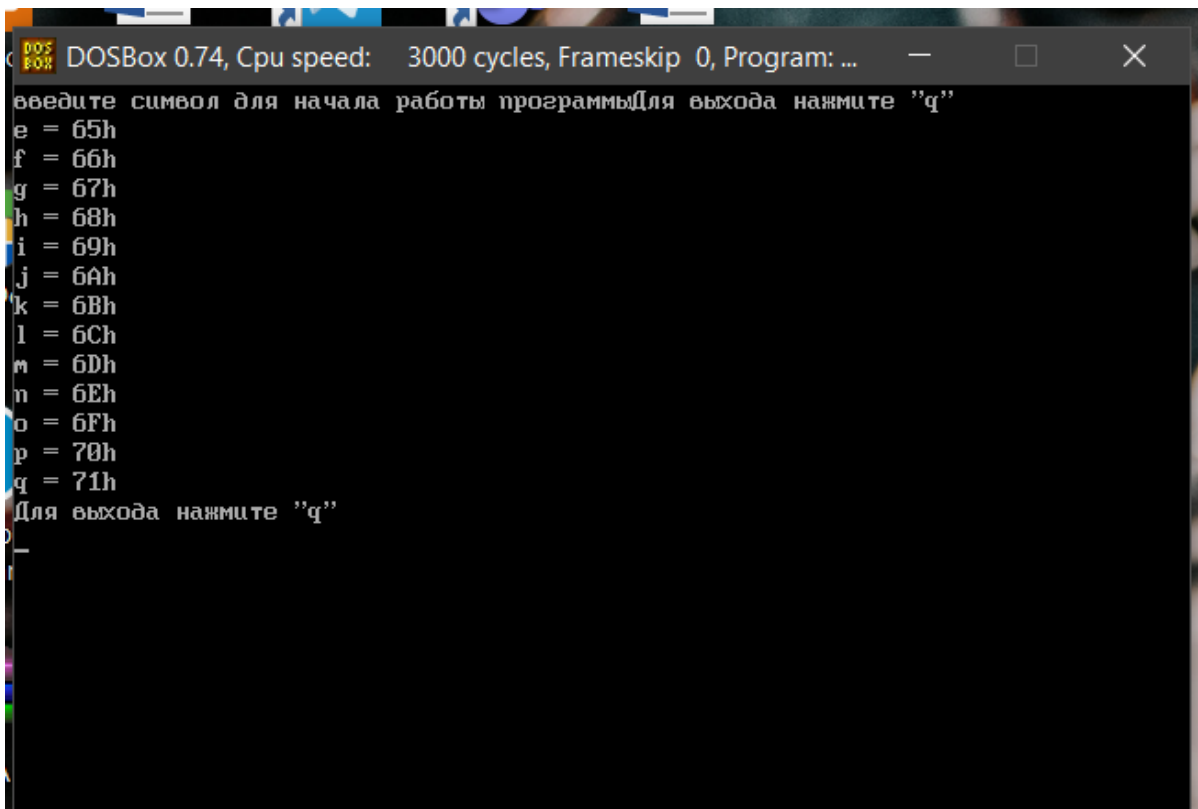
```

115 00BF CD 21                int 021h
116 00C1 C3                  ret
117 00C2                      getch endp
118
119                          ; Процедура - очистка экрана
120 00C2                      clrscr proc
121 00C2 B4 00                mov ah, 00h
122 00C4 B0 02                mov al, 02
123 00C6 CD 10                int 10h
124 00C8 C3                  ret
125 00C9                      clrscr endp
126
127                          ; Перевод в 16
128 00C9                      hex proc
129 00C9 50                  push ax
130 00CA D0 E8 D0 E8 D0 E8    D0+    shr al, 4
131    E8
132 00D2 D7                  xlat
133 00D3 8A D0                mov dl, al
134 00D5 E8 FFD5              call putch
135 00D8 58                  pop ax
136 00D9 24 0F                and al, 00001111b
137 00DB D7                  xlat
138 00DC 8A D0                mov dl, al
139 00DE E8 FFCC              call putch
140 00E1 BA 0068              mov dx, 104
141 00E4 E8 FFC6              call putch
142 00E7 E8 FFC8              call clrf
143 00EA C3                  ret
144 00EB                      hex endp
145                          ;Конец сегмента
146 00EB                      MYCODE ends
147                          end start

```


Symbol Name	Type	Value	Cref	(defined at #)
??DATE	Text	"04/30/20"		
??FILENAME	Text	"LR4"		
??TIME	Text	"17:56:30"		
??VERSION	Number	030A		
@CPU	Text	0101H		
@CURSEG	Text	MYCODE	#3	
@FILENAME	Text	LR4		
@WORDSIZE	Text	2	#3	
CLRF	Near	MYCODE:00B2	34 74 #104	142
CLRSCR	Near	MYCODE:00C2	29 82 #120	
CONTINUESTR	Byte	MYCODE:003A	#15	72
CYCLE	Near	MYCODE:0069	#42	68
EXIT	Near	MYCODE:009F	77 #80	
GETCH	Near	MYCODE:00BD	37 75 #113	
HEX	Near	MYCODE:00C9	61 #128	
HEX_STRING	Byte	MYCODE:0000	#6	24
MAIN	Near	MYCODE:0056	#26	78
PUTCH	Near	MYCODE:00AD	49 53 55 57 #97	106 108 134 139 141
PUTST	Near	MYCODE:00A8	33 73 #90	
START	Near	MYCODE:0051	#20	147
STARTSTR	Byte	MYCODE:0010	#9	32
Groups & Segments	Bit	Size Align	Combine	Class Cref (defined at #)
MYCODE	16	00EB	Para none	CODE #3 4 4

6. Результаты работы программы



The screenshot shows a DOSBox window with the title bar "DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: ...". The window contains the following text:

```
Введите символ для начала работы программыДля выхода нажмите "q"  
e = 65h  
f = 66h  
g = 67h  
h = 68h  
i = 69h  
j = 6Ah  
k = 6Bh  
l = 6Ch  
m = 6Dh  
n = 6Eh  
o = 6Fh  
p = 70h  
q = 71h  
Для выхода нажмите "q"
```

7. Выводы по ЛР № 4

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