

# **Лабораторная работа №7**

**Команды безусловного и условного переходов в Nasm.  
Программирование ветвлений**

Налобин Михаил Дмитриевич

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Ход работы</b>	<b>6</b>
<b>3</b>	<b>Выводы</b>	<b>21</b>

## Список иллюстраций

2.1	Создание каталога lab07 . . . . .	6
2.2	Код программы lab7-1.asm . . . . .	7
2.3	Запуск программы lab7-1 . . . . .	8
2.4	Измененный код программы lab7-1.asm . . . . .	9
2.5	Запуск измененной программы lab7-1 . . . . .	10
2.6	Повторно измененный код программы lab7-1.asm . . . . .	11
2.7	Запуск повторно измененной программы lab7-1 . . . . .	12
2.8	Создание файла lab7-2.asm . . . . .	12
2.9	Код программы lab7-2.asm . . . . .	13
2.10	Запуск программы lab7-2 . . . . .	14
2.11	Создание файла листинга lab7-2.lst . . . . .	14
2.12	Строки 24-26 в листинге lab7-2.lst . . . . .	14
2.13	Удаление одного из операндов . . . . .	15
2.14	Попытка создания листинга . . . . .	15
2.15	Полученный листинг . . . . .	16
2.16	Создание файла lab7-3.asm . . . . .	16
2.17	Код программы lab7-3.asm . . . . .	17
2.18	Запуск программы lab7-3 . . . . .	18
2.19	Создание файла lab7-4.asm . . . . .	18
2.20	Код программы lab7-4.asm . . . . .	19
2.21	Запуск программы lab7-4 . . . . .	20

## Список таблиц

# 1 Цель работы

1. Изучить команды условного и безусловного переходов
2. Освоить написание программ с использованием переходов
3. Познакомиться с назначением и структурой файла листинга

## 2 Ход работы

Создали каталог lab07 для файлов лабораторной работы №7 и в нем файл lab7-1.asm (рис. 2.1).

```
[mdnalobin@mdnalobin ~]$ mkdir -p ~/work/arch-pc/lab07  
[mdnalobin@mdnalobin ~]$ cd ~/work/arch-pc/lab07  
[mdnalobin@mdnalobin lab07]$ touch lab7-1.asm
```

Рис. 2.1: Создание каталога lab07

Открыли с помощью gedit файл lab7-1.asm и переписали в него пример Листинга 7.1. (рис. 2.2).

```

#include          'in_out.asm'

SECTION .data
msg1:    DB  'Сообщение №1',0
msg2:    DB  'Сообщение №2',0
msg3:    DB  'Сообщение №3',0

SECTION .text
GLOBAL _start
_start:

    jmp _label2

_label1:
    mov eax, msg1
    call printf

_label2:
    mov eax, msg2
    call printf

_label3:
    mov eax, msg3
    call printf

_end:
    call quit

```

Рис. 2.2: Код программы lab7-1.asm

Далее создали исполняемый файл и запустили его (рис. 2.3).

```
[mdnalobin@mdnalobin lab07]$ nasm -f elf lab7-1.asm  
[mdnalobin@mdnalobin lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o  
[mdnalobin@mdnalobin lab07]$ ./lab7-1  
Сообщение №2  
Сообщение №3
```

Рис. 2.3: Запуск программы lab7-1

После чего снова открыли файл lab7-1.asm и изменили текст в соответствии с Листингом 7.2. (рис. 2.4).



```
%include          'in_out.asm'

SECTION .data
msg1:  DB  'Сообщение №1',0
msg2:  DB  'Сообщение №2',0
msg3:  DB  'Сообщение №3',0

SECTION .text
GLOBAL _start
_start:

    jmp _label2

_label1:
    mov eax, msg1
    call sprintfLF
    jmp _end

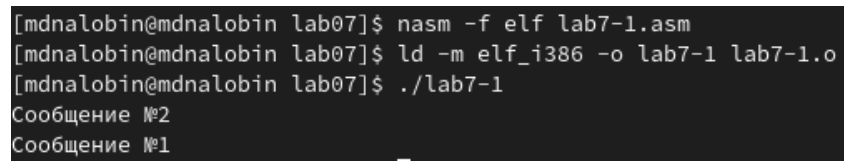
_label2:
    mov eax, msg2
    call sprintfLF
    jmp _label1

_label3:
    mov eax, msg3
    call sprintfLF

_end:
    call quit
```

Рис. 2.4: Измененный код программы lab7-1.asm

Так же создали исполняемый файл и запустили его (рис. 2.5).



```
[mdnlobin@mdnlobin lab07]$ nasm -f elf lab7-1.asm
[mdnlobin@mdnlobin lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[mdnlobin@mdnlobin lab07]$ ./lab7-1
Сообщение №2
Сообщение №1
```

Рис. 2.5: Запуск измененной программы lab7-1

Затем опять изменили код программы lab7-1.asm, чтобы программа выводила сообщения в обратном порядке, создали исполняемый файл и запустили его (рис. 2.6 и рис. 2.7).

```

#include                'in_out.asm'

SECTION .data
msg1:    DB  'Сообщение №1',0
msg2:    DB  'Сообщение №2',0
msg3:    DB  'Сообщение №3',0

SECTION .text
GLOBAL _start
_start:

    jmp _label3

_label1:
    mov eax, msg1
    call sprintfLF
    jmp _end

_label2:
    mov eax, msg2
    call sprintfLF
    jmp _label1

_label3:
    mov eax, msg3
    call sprintfLF
    jmp _label2

_end:
    call quit

```

Рис. 2.6: Повторно измененный код программы lab7-1.asm

```
[mdnalogin@mdnalogin lab07]$ nasm -f elf lab7-1.asm
[mdnalogin@mdnalogin lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[mdnalogin@mdnalogin lab07]$ ./lab7-1
Сообщение №3
Сообщение №2
Сообщение №1
```

Рис. 2.7: Запуск повторно измененной программы lab7-1

Создали файл lab7-2.asm и, внимательно изучив текст Листинга 7.3., заполнили его (рис. 2.8 и рис. 2.9).

```
[mdnalogin@mdnalogin lab07]$ touch lab7-2.asm
```

Рис. 2.8: Создание файла lab7-2.asm

```

#include          'in_out.asm'

SECTION .data
    msg1:         DB 'Введите B:',0h
    msg2:         DB 'Наибольшее число: ',0h
    A:             DD '20'
    C:             DD '50'

SECTION .bss
    max:          resb 10
    B:            resb 10

SECTION .text
GLOBAL _start
_start:

    mov eax,msg1
    call sprint

    mov ecx,B
    mov edx,10
    call sread

    mov eax,B
    call atoi
    mov [B],eax

    mov ecx,[A]
    mov [max],ecx

    cmp ecx,[C]
    jg check_B
    mov ecx,[C]
    mov [max],ecx

    check_B:
    mov eax,max

```

Рис. 2.9: Код программы lab7-2.asm

Далее создали исполняемый файл и запустили его, проверив на правильность работы (рис. 2.10).

```
[mdnalobin@mdnalobin lab07]$ nasm -f elf lab7-2.asm
[mdnalobin@mdnalobin lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[mdnalobin@mdnalobin lab07]$ ./lab7-2
Введите B:60
Наибольшее число: 60
[mdnalobin@mdnalobin lab07]$ ./lab7-2
Введите B:20
Наибольшее число: 50
[mdnalobin@mdnalobin lab07]$ ./lab7-2
Введите B:51
Наибольшее число: 51
```

Рис. 2.10: Запуск программы lab7-2

Создали файл листинга lab7-2.lst и открыли его с помощью редактора mcedit. Для объяснения возьмем строки 24-26. Первые числа являются номерами строк, затем идет адрес или же смещение от базового адреса структуры до интересующего вас поля, далее идет машинный код, показывающий итог ассемблирования исходной строки, и завершает листинг соответственно исходный код, где сначала введенное значение B перемещается в регистр eax для выполнения следующего хода, вызывается подпрограмма для перевода символа в число и число записывается обратно в B (рис. 2.11 и рис. 2.12).

```
[mdnalobin@mdnalobin lab07]$ nasm -f elf -l lab7-2.lst lab7-2.asm
[mdnalobin@mdnalobin lab07]$ mcedit lab7-2.lst
```

Рис. 2.11: Создание файла листинга lab7-2.lst

```
24 00000101 B8[0A000000] <----->mov eax,B
25 00000106 E891FFFFFF <----->call atoi
26 0000010B A3[0A000000] <----->mov [B],eax
```

Рис. 2.12: Строки 24-26 в листинге lab7-2.lst

Открыли файл с программой lab7-2.asm и из инструкции mov убрал один из операндов (рис. 2.13).

```
mov ecx,B
mov edx,10
call sread
```

```
mov eax|
call atoi
mov [B],eax

mov ecx,[A]
mov [max],ecx
```

Рис. 2.13: Удаление одного из операндов

Попробуем оттранслировать для получения файла листинга, в результате чего получаем ошибку в консоле и в самом листинге (рис. 2.14 и рис. 2.15).

```
[mdnalobin@mdnalobin lab07]$ gedit lab7-2.asm
[mdnalobin@mdnalobin lab07]$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:24: error: invalid combination of opcode and operands
```

Рис. 2.14: Попытка создания листинга

```

24                                     <-----> mov eax
24      *****                      error: invalid combination of opcode and operands
25 00000101 E896FFFFFF                <-----> call atoi
26 00000106 A3[0A000000]              <-----> mov [B],eax

```

Рис. 2.15: Полученный листинг

### ##Самостоятельная работа

Создали файл lab7-3.asm и написали в нем программу для нахождения наименьшего из 3 чисел (рис. 2.16 и рис. 2.17).

```
[mdn@alobin@mdn alobin lab07]$ touch lab7-3.asm
```

Рис. 2.16: Создание файла lab7-3.asm



```

#include          'in_out.asm'

SECTION .data
    msg1:  DB  'Введите A, B и C: ',0h
    msg2:  DB  'Наименьшее число: ',0h

SECTION .bss
    min:  resb 10
    A:    resb 10
    B:    resb 10
    C:    resb 10

SECTION .text
GLOBAL _start
_start:

    mov eax,msg1
    call sprintLF

    mov ecx,A
    mov edx,10
    call sread

    mov eax,A
    call atoi
    mov [A],eax

    mov ecx,B
    mov edx,10
    call sread

    mov eax,B
    call atoi
    mov [B],eax

```

Рис. 2.17: Код программы lab7-3.asm

После создали исполняемый файл lab7-3 и проверили на корректность, используя числа из 6 и 1 вариантов (рис. 2.18).

```
[mdnalobin@mdnalobin lab07]$ nasm -f elf lab7-3.asm
[mdnalobin@mdnalobin lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[mdnalobin@mdnalobin lab07]$ ./lab7-3
Введите A, B и C:
17
23
45
Наименьшее число: 17
[mdnalobin@mdnalobin lab07]$ ./lab7-3
Введите A, B и C:
79
83
41
Наименьшее число: 41
```

Рис. 2.18: Запуск программы lab7-3

Создали файл lab7-4.asm и написали в нем программу для вычисления значения заданной функции по 6 варианту с переменными x и a, введенных с клавиатуры (рис. 2.19 и рис. 2.20).

```
[mdnalobin@mdnalobin lab07]$ touch lab7-4.asm
```

Рис. 2.19: Создание файла lab7-4.asm

```

%include          'in_out.asm'

SECTION .data
    msg1:         DB 'Введите x и a:',0h
    msg2:         DB 'Результат: ',0h

SECTION .bss
    x: resb 10
    a: resb 10

SECTION .text
GLOBAL _start
_start:

    mov eax,msg1
    call sprintLF

    mov ecx,x
    mov edx,10
    call sread

    mov eax,x
    call atoi
    mov [x],eax

    mov ecx,a
    mov edx,10
    call sread

    mov eax,a
    call atoi
    mov [a],eax

    cmp eax,[x]
    je check_a
    jne check_b

```

Рис. 2.20: Код программы lab7-4.asm

После создали исполняемый файл lab7-4 и проверили, подставив предоставленные значения для x и a (рис. 2.21).

```
[mdnalobin@mdnalobin lab07]$ nasm -f elf lab7-4.asm
[mdnalobin@mdnalobin lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[mdnalobin@mdnalobin lab07]$ ./lab7-4
Введите x и a:
2
2
Результат: 4
[mdnalobin@mdnalobin lab07]$ ./lab7-4
Введите x и a:
2
1
Результат: 10
```

Рис. 2.21: Запуск программы lab7-4

## 3 Выводы

В ходе данной лабораторной работы поработали с командами условного и безусловного переходов, также приобрели навык написания программ с их использованием и изучили назначение и структуру файла листинга.

...