

Отчёта по лабораторной работе №7

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

Мошаров Денис Максимович

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Реализация переходов в NASM	6
3.2	Изучение структуры файлы листинга	11
3.3	Задание для самостоятельной работы	12
4	Выводы	17

Список иллюстраций

3.1	Создаем каталог с помощью команды <code>mkdir</code> и файл с помощью команды <code>touch</code>	6
3.2	Заполняем файл	7
3.3	Запускаем файл и смотрим на его работу	7
3.4	Изменяем файл	8
3.5	Запускаем файл и смотрим на его работу	8
3.6	Создаем файл командой <code>touch</code>	9
3.7	Заполняем файл	10
3.8	Смотрим на работу программ	10
3.9	Изучаем файл	11
3.10	Удаляем операндум из файла	12
3.11	Создаем файл командой <code>touch</code>	12
3.12	Пишем программу	13
3.13	Смотрим на рабботу программы(всё верно)	14
3.14	Создаем файл командой <code>touch</code>	14
3.15	Пишем программу	15
3.16	Проверяем работу программы	16
3.17	Проверяем работу программы	16

1 Цель работы

Освоить условного и безусловного перехода. Ознакомиться с назначением и структурой файла листинга.

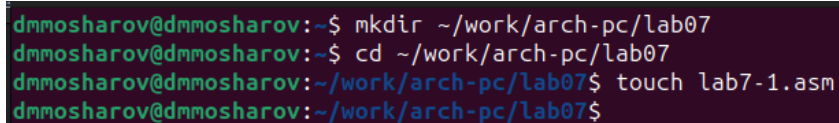
2 Задание

Написать программы для решения системы выражений.

3 Выполнение лабораторной работы

3.1 Реализация переходов в NASM

Создаем каталог для программ ЛБ7, и в нем создаем файл (рис. 3.2).



```
dmmosharov@dmmosharov:~$ mkdir ~/work/arch-pc/lab07
dmmosharov@dmmosharov:~$ cd ~/work/arch-pc/lab07
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ touch lab7-1.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$
```

Рис. 3.1: Создаем каталог с помощью команды `mkdir` и файл с помощью команды `touch`

Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 7.1 (рис. ??).

```

%include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения

```

Рис. 3.2: Заполняем файл

Создаем исполняемый файл и запускаем его (рис. 3.3).

```

dmmosharov@dmmosharov:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
dmmosharov@dmmosharov:~/work/arch-pc/lab07$

```

Рис. 3.3: Запускаем файл и смотрим на его работу

Снова открываем файл для редактирования и изменяем его в соответствии с листингом 7.2 (рис. 3.4).

```

#include 'in_out.asm' ; подключение внешнего файла
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 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения

```

Рис. 3.4: Изменяем файл

Создаем исполняемый файл и запускаем его (рис. 3.5).

```

dmmosharov@dmmosharov:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
dmmosharov@dmmosharov:~/work/arch-pc/lab07$

```

Рис. 3.5: Запускаем файл и смотрим на его работу

Создаем новый файл (рис. 3.6).

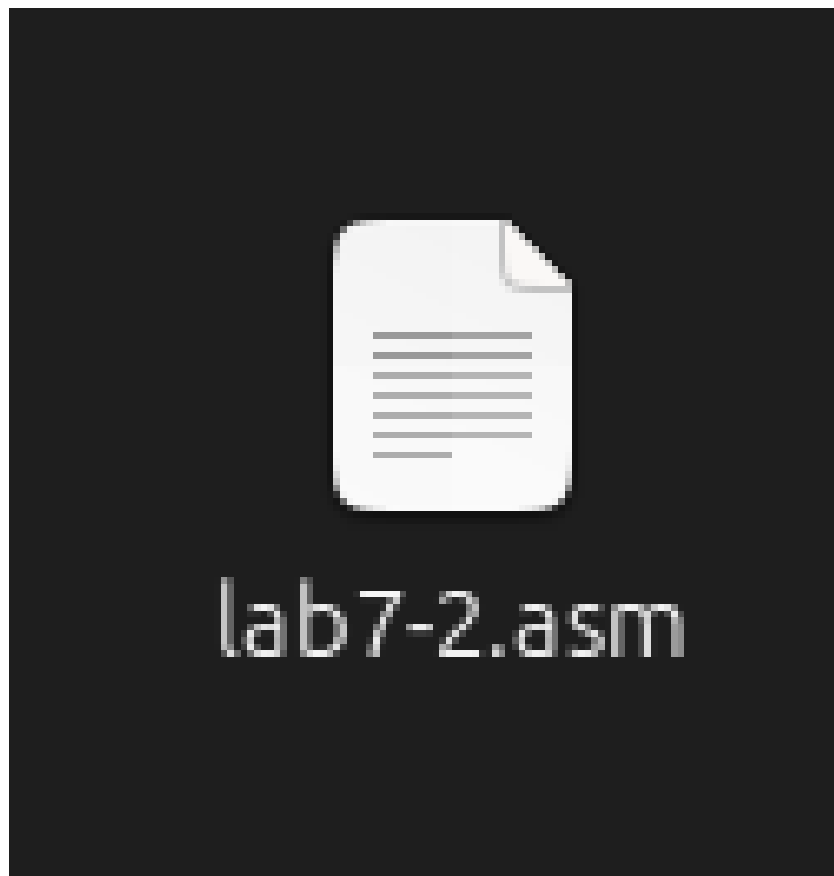


Рис. 3.6: Создаем файл командой touch

Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 7.3 (рис. 3.7).

```

%include 'in_out.asm' ; подкл
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 ; Вывод на экран строки
call sprintf ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения

```

Рис. 3.7: Заполняем файл

Создаем исполняемый файл и проверяем его работу, вводя разные значения В (рис. 3.8).

```

dmmosharov@dmmosharov:~/work/arch-pc/lab07$ touch lab7-2.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ./lab7-2
Введите В: 23
Наибольшее число: 50

```

Рис. 3.8: Смотрим на работу программ

3.2 Изучение структуры файлы листинга

Открываем файл листинга с помощью команды mscedit и изучаем его (рис. 3.9).

```
/home/dm-7-2.lst [----] 0 L: [ 1+ 0 1/225] *(0 /14458b) 0032 0x020 [*][X]
1      %include 'in_out.asm'
2      <1> ;----- slen -----
3      <1> ; Функция вычисления длины сообщения
4      <1> slen:.....
5      <1>     push    ebx.....
6      <1>     mov     ebx, eax.....
7      <1>     nextchar:.....
8      <1>     cmp     byte [eax], 0...
9      <1>     jz      finished.....
10     <1>     inc     eax.....
11     <1>     jmp     nextchar.....
12     <1>     finished:.....
13     <1>     sub     eax, ebx.....
14     <1>     pop     ebx.....
15     <1>     ret.....
16     <1>     ret.....
1Помощь 2Сох-ть 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюМС 10Выход
```

Рис. 3.9: Изучаем файл

Строка 33: 0000001D-адрес в сегменте кода, BB01000000-машинный код, mov ebx,1-присвоение переменной ebx значения 1.

Строка 34: 00000022-адрес в сегменте кода, B804000000-машинный код, mov eax,4-присвоение переменной eax значения 4.

Строка 35 00000027-адрес в сегменте кода, CD80-машинный код, int 80h-вызов ядра.

Открываем файл и удаляем один операндум (рис. 3.10).

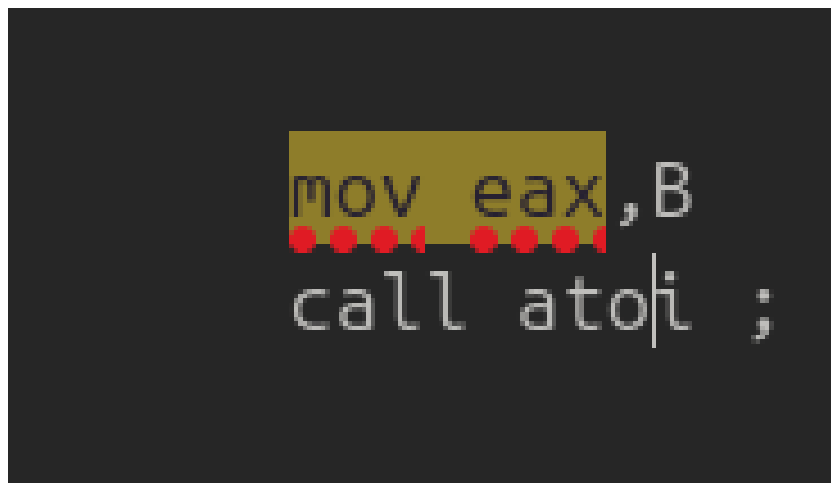


Рис. 3.10: Удаляем операндум из файла

Транслируем с получением файла листинга При трансляции файла, выдается ошибка, но создаются исполнительный файл lab7-2 и lab7-2.lst

3.3 Задание для самостоятельной работы

ВАРИАНТ-2

1. Напишите программу нахождения наименьшей из 3 целочисленных переменных a, b и c. Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу.

Создаем новый файл (рис. 3.11).

```
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ touch lab7-4.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$
```

Рис. 3.11: Создаем файл командой touch

Открываем его и пишем программу, которая выберет наименьшее число из трех (2 числа уже в программе, 3е вводится из консоли) (рис. 3.12).

```

#include 'in_out.asm'
section .data
msg1 DB 'Введите B: ', 0h
msg2 DB 'Наименьшее число: ', 0h
A dd '82'
C dd '61'
section .bss
min 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 [min],ecx
check_B:
mov eax,min
mov ecx,[min]
cmp ecx,[B]
il fin

```

Рис. 3.12: Пишем программу

Транслируем файл и смотрим на работу программы (рис. 3.13).

```
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ./lab7-3
Введите B: 59
Наименьшее число: 59
```

Рис. 3.13: Смотрим на работу программы(всё верно)

2. Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений x и a из 7.6.

Создаем новый файл (рис. 3.14).

```
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ touch lab7-4.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$
```

Рис. 3.14: Создаем файл командой touch

Открываем его и пишем программу, которая решит систему уравнений, при данных, введенных в консоль (рис. 3.15).

```

#include 'in_out.asm'
section .data
msg1 DB 'Введите X: ',0h
msg2 DB 'Введите A: ',0h
ans: DB 'F(x)=',0h
section .bss
x: RESB 80
a: RESB 80
res: RESB 80
section .text
global _start
_start:
mov eax,msg1
call sprint
mov ecx,x
mov edx,80
call sread
mov eax,x
call atoi
mov [x],eax
mov eax,msg2
call sprint
mov ecx,a
mov edx,80
call sread
mov eax,a
mov edx,80

```

Рис. 3.15: Пишем программу

Транслируем файл и проверяем его работу при $x=1$ и $a=1$ (рис. ??).

```
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ./lab7-4
Введите X: 5
Введите A: 7
F(x)=6
```

Рис. 3.16: Проверяем работу программы

Транслируем файл и проверяем его работу при $x=2$ и $a=2$ (рис. ??).

```
dmmosharov@dmmosharov:~/work/arch-pc/lab07$ ./lab7-4
Введите X: 6
Введите A: 4
F(x)=5
```

Рис. 3.17: Проверяем работу программы

4 Выводы

Мы познакомились с структурой файла листинга, изучили команды условного и безусловного перехода.