

# **Лабораторная работа No8**

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

Воробчук Лилия Андреевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Самостоятельная работа</b>	<b>16</b>
<b>4</b>	<b>Выводы</b>	<b>19</b>

# Список иллюстраций

2.1	Создание каталогов и файлов . . . . .	6
2.2	Ввод листинга 8.1 . . . . .	7
2.3	Запуск программы из файла . . . . .	8
2.4	Ввод изменений в программу из файла . . . . .	9
2.5	Запуск измененной программы из файла . . . . .	10
2.6	Повторный ввод изменений в программу из файла . . . . .	11
2.7	Запуск повторно измененной программы из файла . . . . .	12
2.8	Ввод листинга 8.3 . . . . .	13
2.9	Запуск программы из файла 'lab8-2.asm' . . . . .	14
2.10	Листинг программы из файла 'lab8-2.asm' . . . . .	14
2.11	изменение файла . . . . .	15
2.12	ошибка . . . . .	15
3.1	Создание файла . . . . .	16
3.2	Ввод программы в файл proper.asm . . . . .	17
3.3	Запуск программы из файла . . . . .	18
3.4	Запуск программы из файла boo.asm . . . . .	18

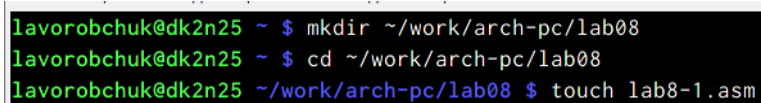
## **Список таблиц**

# 1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

## 2 Выполнение лабораторной работы

1. Создаю каталог 'lab08' с помощью команды `mkdir`, перехожу в него с помощью команды `cd` и создаем в нем файл 'lab8-1.asm' с помощью команды `touch` (рис. 2.1)



```
lavorobchuk@dk2n25 ~ $ mkdir ~/work/arch-pc/lab08
lavorobchuk@dk2n25 ~ $ cd ~/work/arch-pc/lab08
lavorobchuk@dk2n25 ~/work/arch-pc/lab08 $ touch lab8-1.asm
```

Рис. 2.1: Создание каталогов и файлов

2. Открываю файл 'lab8-1.asm' и ввожу листинг 8.1 (рис. 2.2)

```

1 %include 'in_out.asm'
2
3 SECTION .data
4 msg1: DB 'Сообщение No 1',0
5 msg2: DB 'Сообщение No 2',0
6 msg3: DB 'Сообщение No 3',0
7 SECTION .text
8 GLOBAL _start
9 _start:
10
11 jmp _label2
12 _label1:
13 mov eax, msg1
14 call sprintf
15 _label2:
16 mov eax, msg2
17 call sprintf
18 _label3:
19 mov eax, msg3
20 call sprintf
21 _end:
22
23 call quit

```

Рис. 2.2: Ввод листинга 8.1

Создаем исполняемый файл и запускаем его (рис. 2.3). Программа выводит правильный результат, значит, она написана корректно. (рис. 2.3)

```
lavorobchuk@dk2n25 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
lavorobchuk@dk2n25 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
lavorobchuk@dk2n25 ~/work/arch-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 3
```

Рис. 2.3: Запуск программы из файла

Снова открываю файл 'lab8-1.asm' и редактирую его так, чтобы она выводила сначала 'Сообщение No2', а потом 'Сообщение No1' (рис. 2.4)



```
1 %include 'in_out.asm'
2
3 SECTION .data
4 msg1: DB 'Сообщение No 1',0
5 msg2: DB 'Сообщение No 2',0
6 msg3: DB 'Сообщение No 3',0
7 SECTION .text
8 GLOBAL _start
9 _start:
10
11 jmp _label2
12 _label1:
13 mov eax, msg1
14 call sprintf
15 jmp _end
16 _label2:
17 mov eax, msg2
18 call sprintf
19 jmp _label1
20 _label3:
21 mov eax, msg3
22 call sprintf
23 _end:
24
25 call quit
```

Рис. 2.4: Ввод изменений в программу из файла

Создаю исполняемый файл и запускаю его. Программа выводит правильный результат, значит, она написана корректно (рис. 2.5)

```
lavorobchuk@dk2n25 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
lavorobchuk@dk2n25 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
lavorobchuk@dk2n25 ~/work/arch-pc/lab08 $ ./lab8-1
Сообщение No 2
Сообщение No 1
```

Рис. 2.5: Запуск измененной программы из файла

Далее редактирую файл 'lab8-1.asm' так, чтобы сообщения выводились в обратной последовательности: 'Сообщение No3', 'Сообщение No2', 'Сообщение No1' (рис. 2.6)

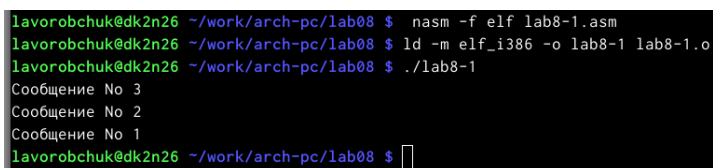
```

1 %include 'in_out.asm'
2
3 SECTION .data
4 msg1: DB 'Сообщение No 1',0
5 msg2: DB 'Сообщение No 2',0
6 msg3: DB 'Сообщение No 3',0
7 SECTION .text
8 GLOBAL _start
9 _start:
10
11 jmp _label3
12
13 _label1:
14 mov eax, msg1
15 call sprintf
16 jmp _end
17
18 _label2:
19 mov eax, msg2
20 call sprintf
21
22 jmp _label1
23
24 _label3:
25 mov eax, msg3
26 call sprintf
27 jmp _label2
28
29 _end:
30
31 call quit

```

Рис. 2.6: Повторный ввод изменений в программу из файла

Создаю исполняемый файл и запускаю его (рис. 2.7)

A terminal window with a black background and green text. The prompt is 'lavorobchuk@dk2n26 ~/work/arch-pc/lab08 \$'. The first command is 'nasm -f elf lab8-1.asm'. The second command is 'ld -m elf\_i386 -o lab8-1 lab8-1.o'. The third command is './lab8-1'. The output shows three messages: 'Сообщение No 3', 'Сообщение No 2', and 'Сообщение No 1'. The prompt returns to 'lavorobchuk@dk2n26 ~/work/arch-pc/lab08 \$' with a cursor.

```
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ nasm -f elf lab8-1.asm
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-1 lab8-1.o
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ./lab8-1
Сообщение No 3
Сообщение No 2
Сообщение No 1
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $
```

Рис. 2.7: Запуск повторно измененной программы из файла

3. Создаю файл 'lab8-2.asm' с помощью команды Создаю исполняемый файл и запускаю его. (рис. 2.8)

```

1 %include 'in_out.asm'
2 section .data
3 msg1 db 'Введите B: ',0h
4 msg2 db "Наибольшее число: ",0h
5 A dd '20'
6 C dd '50'
7 section .bss
8 max resb 10
9 B resb 10
10 section .text
11
12 global _start
13 _start:
14
15 mov eax,msg1
16 call sprint
17
18 mov ecx,B
19 mov edx,10
20 call sread
21
22 mov eax,B
23 call atoi
24 mov [B],eax
25 mov ecx,[A]
26 mov [max],ecx
27
28 cmp ecx,[C]
29 jg check_B
30 mov ecx,[C]
31 mov [max],ecx
32
33 check_B:
34 mov eax,max
35 call atoi
36 mov [max],eax
37
38 mov ecx,[max]
39 cmp ecx,[B]
40 jg fin
41 mov ecx,[B]
42
43 mov [max],ecx
44
45 fin:
46 mov eax, msg2
47 call sprint
48 mov eax [max]

```

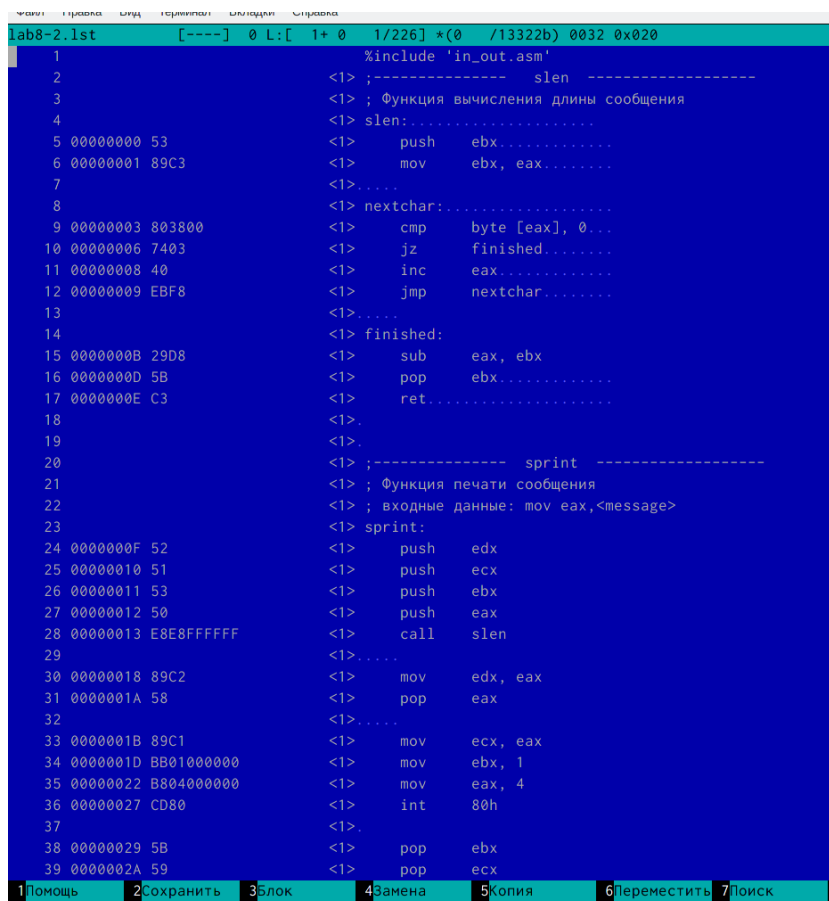
Рис. 2.8: Ввод листинга 8.3

Проверяю работу программы, вводя несколько чисел - программа выводит правильный результат, значит, она написана корректно (рис. 2.9)

```
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ nasm -f elf -l lab8-2.lst lab8-2.asm
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ./lab8-2
Введите B: 12
Наибольшее число: 50
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ nasm -f elf -l lab8-2.lst lab8-2.asm
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o lab8-2 lab8-2.o
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ./lab8-2
Введите B: 89
Наибольшее число: 89
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $
```

Рис. 2.9: Запуск программы из файла 'lab8-2.asm'

#### 4. Открываем файл листинга с помощью редактора mcedit (рис. 2.10)



```
lab8-2.lst  [----]  0 L: [ 1+ 0  1/226] *(0 /13322b) 0032 0x020
1          %include 'in_out.asm'
2          <I> ;----- slen -----
3          <I> ; Функция вычисления длины сообщения
4          <I> slen:
5          00000000 53          <I> push ebx
6          00000001 89C3       <I> mov ebx, eax
7          <I> .....
8          <I> nextchar:
9          00000003 803800     <I> cmp byte [eax], 0
10         00000006 7403       <I> jz finished
11         00000008 40        <I> inc eax
12         00000009 EBF8       <I> jmp nextchar
13         <I> .....
14         <I> finished:
15         0000000B 29D8       <I> sub eax, ebx
16         0000000D 5B        <I> pop ebx
17         0000000E C3        <I> ret
18         <I> .....
19         <I> .....
20         <I> ;----- sprintf -----
21         <I> ; Функция печати сообщения
22         <I> ; входные данные: mov eax, <message>
23         <I> sprintf:
24         0000000F 52        <I> push edx
25         00000010 51        <I> push ecx
26         00000011 53        <I> push ebx
27         00000012 50        <I> push eax
28         00000013 E8E8FFFF    <I> call slen
29         <I> .....
30         00000018 89C2       <I> mov edx, eax
31         0000001A 58        <I> pop eax
32         <I> .....
33         0000001B 89C1       <I> mov ecx, eax
34         0000001D BB010000     <I> mov ebx, 1
35         00000022 B8040000     <I> mov eax, 4
36         00000027 CD80       <I> int 80h
37         <I> .....
38         00000029 5B        <I> pop ebx
39         0000002A 59        <I> pop ecx
```

Рис. 2.10: Листинг программы из файла 'lab8-2.asm'

#### 5. Открываю файл 'lab8-2.asm' и убираю у команды 'cmp' второй операнд (рис. 2.11)

```
27  
28 cmp ecx  
29 jg check_B  
30 mov ecx,[C]  
31 mov [max],ecx  
32
```

Рис. 2.11: изменение файла

Выполняю трансляцию с получением файла листинга . Программа выводит ошибку, при этом файл листинга создается. Если открыть его, мы увидим, что в файле листинга также обозначена ошибка отсутствия одного операнда (рис. 2.12)

```
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ nasm -f elf -l lab8-2.lst lab8-2.asm  
lab8-2.asm:28: error: invalid combination of opcode and operands
```

Рис. 2.12: ошибка

### 3 Самостоятельная работа

1. Значения  $a$ ,  $b$  и  $c$  для первого задания, согласно таблице, 45, 67 и 15. Значит, программа должна выводить число 15 (рис. 3.1) Создаю файл 'proper.asm' и пишу в нем программу для вывода наименьшего числа (рис. 3.2)



```
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ touch proper.asm
```

Рис. 3.1: Создание файла



```

1 %include 'in_out.asm'
2 section .data
3 msg1 db 'Введите B: ',0h
4 msg2 db "Наименьшее число: ",0h
5 A dd '45'
6 C dd '15'
7
8 section .bss
9 min resb 10
10 B resb 10
11 section .text
12
13 global _start
14 _start:
15
16 mov eax,msg1
17 call sprint
18
19 mov ecx,B
20 mov edx,10
21 call sread
22
23 mov eax,B
24 call atoi
25 mov [B],eax
26
27 mov ecx,[A]
28 mov [min],ecx
29
30 cmp ecx,[C]
31 jb check_B
32 mov ecx,[C]
33 mov [min],ecx
34
35 check_B:
36 mov eax,min
37 call atoi
38 mov [min],eax
39
40 mov ecx,[min]
41 cmp ecx,[B]
42 jb fin
43 mov ecx,[B]
44 mov [min],ecx
45
46 fin:
47 mov eax, msg2

```

Рис. 3.2: Ввод программы в файл proper.asm

Создаю исполняемый файл и запускаю его. Программа выводит правильный результат, значит, она написана корректно. (рис. 3.3)

```
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ nasm -f elf proper.asm
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o proper proper.o
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ./proper
Введите B: 67
Наименьшее число: 15
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $
```

Рис. 3.3: Запуск программы из файла

2. Создаю файл 'boo.asm' и ввожу в него программу. Проверяю его работу (рис. 3.4)

```
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ nasm -f elf boo.asm
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o boo boo.o
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ./boo
Введите x:3
Введите a:0
Функция равна:7
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ nasm -f elf boo.asm
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ld -m elf_i386 -o boo boo.o
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $ ./boo
Введите x:3
Введите a:2
Функция равна:8
lavorobchuk@dk2n26 ~/work/arch-pc/lab08 $
```

Рис. 3.4: Запуск программы из файла boo.asm

## 4 Выводы

В ходе выполнения данной лабораторной работы я изучила команды условного и безусловного переходов, приобрела навыки написания программ с использованием переходов, ознакомилась с назначением и структурой файла листинга.