

Лабораторная работа №7

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

Тищенко Диана Борисовна

Содержание

1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Самостоятельная работа.	11
4	Вывод	15

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) Я создала каталог lab7 и внутри создала файл lab7-1.asm

```
> mkdir ~/work/arch-pc/lab07
> cd ~/work/arch-pc/lab07
> touch lab7-1.asm
> ls
lab7-1.asm
~/work/arch-pc/lab07 master !227 ?2 > |
```

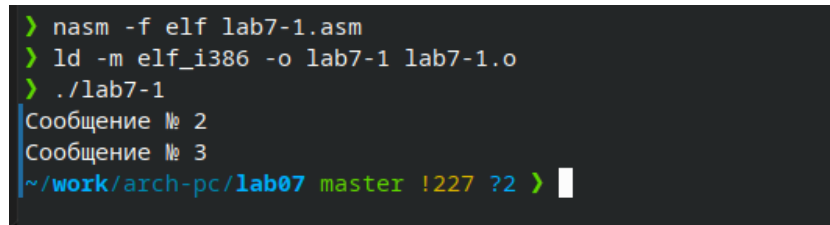
Рис. 2.1: Создание файла lab8-1.asm

- 2) Я ввела в файл текст программы и запустила его.

```
%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. 2.2: Текст в файле lab8-1.asm

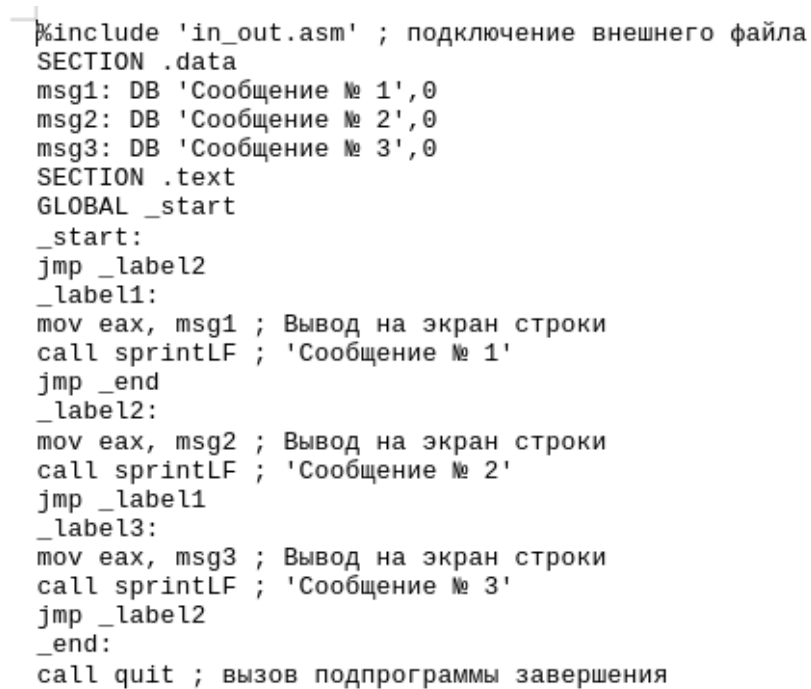
3) Я создала исполняемый файл и запустила его. Результат соответствовал нужному.



```
> nasm -f elf lab7-1.asm
> ld -m elf_i386 -o lab7-1 lab7-1.o
> ./lab7-1
Сообщение № 2
Сообщение № 3
~/work/arch-pc/lab07 master !227 ?2 > |
```

Рис. 2.3: Запуск программы lab8-1

4) Я изменила текст программы чтобы выводился нужный ответ и создала исполняемый файл.



```
%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'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.4: Изменение текста

```
> nasm -f elf lab7-1.asm
> ld -m elf_i386 -o lab7-1 lab7-1.o
> ./lab7-1
Сообщение № 2
Сообщение № 1
~/work/arch-pc/lab07 master !227 ?2 > █
```

Рис. 2.5: Проверка работы программы

5) Я изменила текст программы чтобы сначала выводило сообщение 3, затем 2, затем 1.

```
%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 _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. 2.6: Изменение текста

6) Запустила программу и проверила ее работу.

```
> nasm -f elf lab7-1.asm
> ld -m elf_i386 -o lab7-1 lab7-1.o
> ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
~/work/arch-pc/lab07 master !227 ?2 > |
```

Рис. 2.7: Запуск программы

7) Я создала файл lab7-2.asm и написала текст программы.

```

#include 'in_out.asm'
section .data
msg1 db 'Введите B: ',0h
msg2 db "Наименьшее число: ",0h
A dd '46'
C dd '74'
section .bss
min resb 10
B resb 10
section .text
global _start
_start:
; ----- Вывод сообщения 'Введите B: '
mov eax,msg1
call sprint
; ----- Ввод 'B'
mov ecx,B
mov edx,10
call sread
; ----- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'
; ----- Записываем 'A' в переменную 'min'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'min = A'
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'min = C'
; ----- Преобразование 'min(A,C)' из символа в число

```

Рис. 2.8: Текст программы для сравнения чисел

8) Я ввела два разных числа чтобы проверить как работает программа.


```
> nasm -f elf lab7-2.asm
> ld -m elf_i386 -o lab7-2 lab7-2.o
> ./lab7-2
Введите В: 78
Наименьшее число: 78
> ./lab7-2
Введите В: 14
Наименьшее число: 46
~/work/arch-pc/lab07 master !227 ?2 > █
```

Рис. 2.9: Программа для сравнения чисел

9) Я создала файл листинга lab7-2.lst и открыл его.

```
> nasm -f elf -l lab7-2.lst lab7-2.asm
> mcedit lab7-2.lst

~/work/a/lab07 master !227 ?2 > █
```

Рис. 2.10: Файл листинга lab8-2.lst

- 10) Проанализировав файл, я поняла как он работает и какие значения выводит.
- 11) Эта строка находится на 21 месте, ее адрес “00000101”, Машинный код - B80A000000, а mov eax,B - исходный текст программы, означающий что в регистр eax мы вносим значения переменной B.

```
21 00000101 B8[0A000000]          mov eax,B
00 00000105 5001555555          33 00000109
```

Рис. 2.11: Объяснения первой строки

- 2) Эта строка находится на 35 месте, ее адрес “00000135”, Машинный код - E862FFFFFF, а call atoi - исходный текст программы, означающий что символ лежащий в строке выше переводится в число.

```
35 00000135 E862FFFFFF      call atoi ;
36 0000013A A21000000000      mov [min],eax
```

Рис. 2.12: Объяснения второй строки

- 3) Эта строка находится на 47 месте, ее адрес “00000163”, Машинный код - A100000000, а `mov eax,max` - исходный текст программы, означающий что число хранившееся в переменной `max` записывается в регистр `eax`.

```
47 00000163 A1[00000000]      mov eax,[min]
```

Рис. 2.13: Объяснения третьей строки

- 11) В строке `mov eax,max` я убрала `max` и попробовал создать файл. Должно было выдать ошибку, так как для программы нужно два операнда. Но исходный файл автоматически исправлялся и прервать это было невозможно, из-за чего терминал ошибку не выдал.

```
> nasm -f elf -l lab7-2.lst lab7-2.asm
~/work/arch-pc/lab07 master !227 ?2 > |
```

Рис. 2.14: Создание файла без одного операнда

3 Самостоятельная работа.

- 1) Я написала программу для нахождения меньшего из трех чисел. Для большего удобства я сделал ввод чисел с клавиатуры. У меня девятый вариант, поэтому числа были :94,5,58. Программа вывела меньшее из этих чисел.

```
%include 'in_out.asm'
section .data
msg1 db 'Введите В: ',0h
msg2 db "Наименьшее число: ",0h
A dd '94'
C dd '58'
section .bss
min resb 10
B resb 10
section .text
global _start
```

Рис. 3.1: Текст программы

```
> nasm -f elf lab7-3.asm
> ld -m elf_i386 -o lab7-3 lab7-3.o
> ./lab7-3
Введите В: 5
Наименьшее число: 58
~/work/arch-pc/lab07 master !227 ?2 > |
```

Рис. 3.2: Результат работы программы

- 2) Я написала программу, чтобы она вычисляла выражение при введенных X и A . Так как у меня 3 вариант, то программа написана для 3 варианта.

```

%include 'in_out.asm'

SECTION .data
msg_x: DB 'Введите значение переменной x: ', 0
msg_a: DB 'Введите значение переменной a: ', 0
rem: DB 'Результат: ', 0

SECTION .bss
x: RESB 80
a: RESB 80

SECTION .text
GLOBAL _start

_start:
    mov eax, msg_x
    call sprint
    mov ecx, x
    mov edx, 80
    call sread
    mov eax, x
    call atoi
    mov edi, eax

    cmp edi, 3
    je calculate_3x

    mov eax, msg_a
    call sprint
    mov ecx, a
    mov edx, 80
    call sread
    mov eax, a
    call atoi
    mov esi, eax

    add esi, 1
    mov edi, esi
    jmp output_result

calculate_3x:
    mov eax, edi
    imul eax, 3

```

Рис. 3.3: Текст программы

```
> touch lab7-4.asm
> nasm -f elf lab7-4.asm
> ld -m elf_i386 -o lab7-4 lab7-4.o
> ./lab7-4
Введите значение переменной x: 3
Результат: 9
> ./lab7-4
Введите значение переменной x: 1
Введите значение переменной a: 4
Результат: 5
~/work/a/lab07 master !227 ?2 > |
```

Рис. 3.4: Проверка работы программы

4 Вывод

Я изучила команды условного и безусловного перехода. Приобрела навыки написания программ с переходами.