

Лабораторная работа №7

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

Байрамова Гюльсабах Акифовна

Содержание

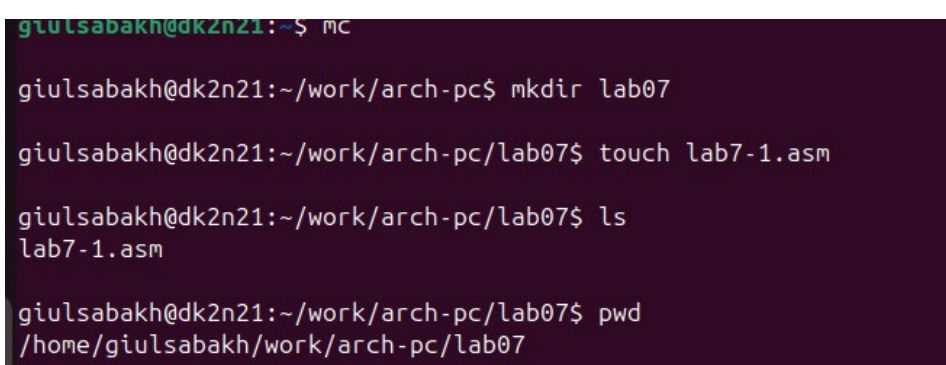
1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Самостоятельная работа.	11
4	Вывод	13

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 lab8 и внутри создала файл lab8-1.asm

A screenshot of a terminal window with a dark background and light-colored text. The terminal shows a series of commands and their outputs. The first line shows the user 'giulsabakh@dk2n21' at the prompt '~\$' with a green cursor. The second line shows the command 'mkdir lab07' and the prompt changes to '~/work/arch-pc\$'. The third line shows 'touch lab7-1.asm' and the prompt changes to '~/work/arch-pc/lab07\$'. The fourth line shows 'ls' and the output 'lab7-1.asm'. The fifth line shows 'pwd' and the output '/home/giulsabakh/work/arch-pc/lab07'.

```
giulsabakh@dk2n21:~$ mc
giulsabakh@dk2n21:~/work/arch-pc$ mkdir lab07
giulsabakh@dk2n21:~/work/arch-pc/lab07$ touch lab7-1.asm
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ls
lab7-1.asm
giulsabakh@dk2n21:~/work/arch-pc/lab07$ pwd
/home/giulsabakh/work/arch-pc/lab07
```

Рис. 2.1: Создание файла lab8-1.asm

2) Я ввела в файл текст программы и запустил его.

```
GNU nano 7.2 /home/giulsabakh/work/
%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 sprintLF ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; '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.

```
giulsabakh@dk2n21:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
```

Рис. 2.3: Запуск программы lab8-1

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

```
/home/giulsabakh/work/arch-pc/lab07/lab7-1.asm
%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 sprintLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.4: Изменение текста

```

giulsabakh@dk2n21:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm

giulsabakh@dk2n21:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o

giulsabakh@dk2n21:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1

```

Рис. 2.5: Проверка работы программы

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

```

/home/giulsabakh/work/arch-pc/lab07/lab7-1.asm
%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'
_end:
call quit ; вызов подпрограммы завершения

```

Рис. 2.6: Изменение текста

6) Запустила программу и проверила ее работу.

```
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
```

Рис. 2.7: Запуск программы

7) Я создала файл lab8-2.asm и написала текст программы.

```
/home/giulsabakh/work/arch-pc/lab07/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:
; ----- 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 'max'
mov ecx,[A] ; 'ecx = A'
mov [max],ecx ; 'max = A'
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
```

Рис. 2.8: Текст программы для сравнения чисел

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

```
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 65
Наибольшее число: 65

giulsabakh@dk2n21:~$ mc
giulsabakh@dk2n21:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 44
Наибольшее число: 50
```

Рис. 2.9: Программа для сравнения чисел

9) Я создала файл листинга lab8-2.lst и открыла его.

```
giulsabakh@dk2n21:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
giulsabakh@dk2n21:~/work/arch-pc/lab07$ mcedit lab7-2.lst
```

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 24 месте, ее адрес “00000101”, Машинный код - B8 [0A000000], а mov eax,B - исходный текст программы, означающий что в регистр eax мы вносим значения переменной B.

```
24 00000101 B8[0A000000]          mov eax,B
```

Рис. 2.11: Объяснения первой строки

2) Эта строка находится на 38 месте, ее адрес “00000134”, Машинный код - E863FFFFFF, а call atoi - исходный текст программы, означающий что символ лежащий в строке выше переводится в число.

```
37 00000121 B8[00000000]          mov eax,max
38 00000134 E863FFFFFF          call atoi
```

Рис. 2.12: Объяснения второй строки

- 3) Эта строка находится на 50 месте, ее адрес "00000162", Машинный код - A1[00000000], а `mov eax,[max]` - исходный текст программы, означающий что число хранившееся в переменной `max` записывается в регистр `eax`.



```
49 0000015B 5001E177  call sprintf
50 00000162 A1[00000000]  mov eax,[max]
51 00000167 50115555  jmp 0000011F
```

Рис. 2.13: Объяснения третьей строки


- 11) В строке `mov eax,max` я убрал `max` и попробовал создать файл. Выдало ошибку, так как для программы нужно два операнда.



```
giulsabakh@dk2n21:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
```

Рис. 2.14: Создание файла без одного операнда

- 12) В файле листинга показывает где именно ошибка и с чем она связана.



```
37                                     mov eax
37  ***** error: invalid combination of opcode and operands
```

Рис. 2.15: Файл листинга без одного операнда

3 Самостоятельная работа.

- 1) Я написала программу для нахождения меньшего из трех чисел. Для большего удобства я сделала ввод чисел с клавиатуры. У меня 11 вариантов, поэтому числа были :21, 28, 34.. Программа вывела меньшее из этих чисел.

```
/home/giulsabakh/work/arch-pc/lab07/lab7-4.asm
%include 'in_out.asm'
section .data
A1 db 'Введите число A: ',0h
B1 db 'Введите число B: ',0h
C1 DB 'Введите число C: ',0h
otv DB 'Наименьшее число: ',0h
SECTION .bss
min RESB 20
A RESB 20
B RESB 20
C RESB 20

section .text
GLOBAL _start
_start:

mov eax,A1
call sprint

mov ecx,A
mov edx,20
call sread

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

mov eax,B1
```

Рис. 3.1: Текст программы

2) Я написала программу, чтобы она вычисляла выражение при введенных X и A. Для большего удобства, выражение которое будет вычисляться я вывожу вначале работы программы. Так как у меня 11 вариант, то программа написана для 11 варианта.



```
GNU nano 7.2 /home/giulsabakh/w
#include 'in_out.asm'

section .data
prim1 DB '4a, x=0',0
prim2 DB '4a+x, x!=0',0
X1 DB 'Введите значение X:',0
A1 DB 'Введите значение a:',0
otv DB 'Ответ: ',0

SECTION .bss
X RESB 10
A RESB 10
F RESB 10
section .text
GLOBAL _start
_start:

mov eax,X1
call sprint

mov ecx,X
mov edx,20
call sread

mov eax,X
call atoi

[ Прочит
^G Справка      ^O Записать
^X Выход         ^R ЧитФайл
^W Поиск        ^\ Замена
^K Вырезать     ^U Вставить
```

Рис. 3.2: Текст программы

4 Вывод

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