

# Отчет по лабораторной работе №7

Дисциплина: архитектура компьютера

Козина Дарья Александровна

## Содержание

Цель работы.....	4
Задание.....	5
Выполнение лабораторной работы.....	6
Реализация переходов в NASM.....	6
Изучение структуры файла листинга.....	9
Выполнение заданий для самостоятельной работы.....	11
Вывод.....	17

## Цель работы

Изучить команды условного и безусловного переходов. Приобрести навыки написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

## Задание

1. Реализовать переходы в NASM;
2. Изучить структуры файла листинга;
3. Выполнение заданий для самостоятельной работы.

# Выполнение лабораторной работы

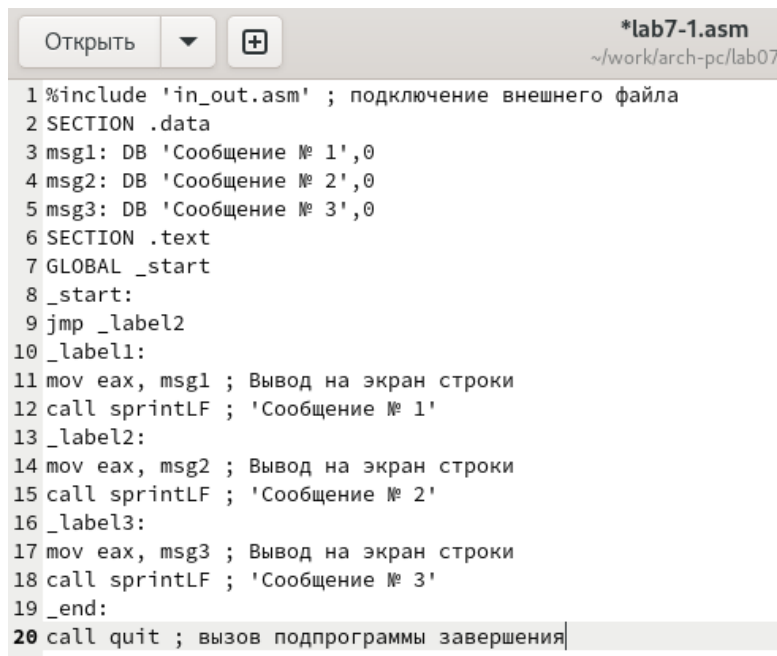
## Реализация переходов в NASM

Создадим каталог для программ лабораторной работы, перейдем в созданный каталог и создадим файл lab7-1.asm (рис. [-@fig:001]).

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

### Создание каталога и файла

С помощью редактора gedit введем в файл lab7-1.asm текст программы в соответствии с листингом 7.1 (рис. [-@fig:002]).



```
*lab7-1.asm
~/work/arch-pc/lab07

1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8 _start:
9 jmp _label2
10 _label1:
11 mov eax, msg1 ; Вывод на экран строки
12 call sprintf ; 'Сообщение № 1'
13 _label2:
14 mov eax, msg2 ; Вывод на экран строки
15 call sprintf ; 'Сообщение № 2'
16 _label3:
17 mov eax, msg3 ; Вывод на экран строки
18 call sprintf ; 'Сообщение № 3'
19 _end:
20 call quit ; вызов подпрограммы завершения
```

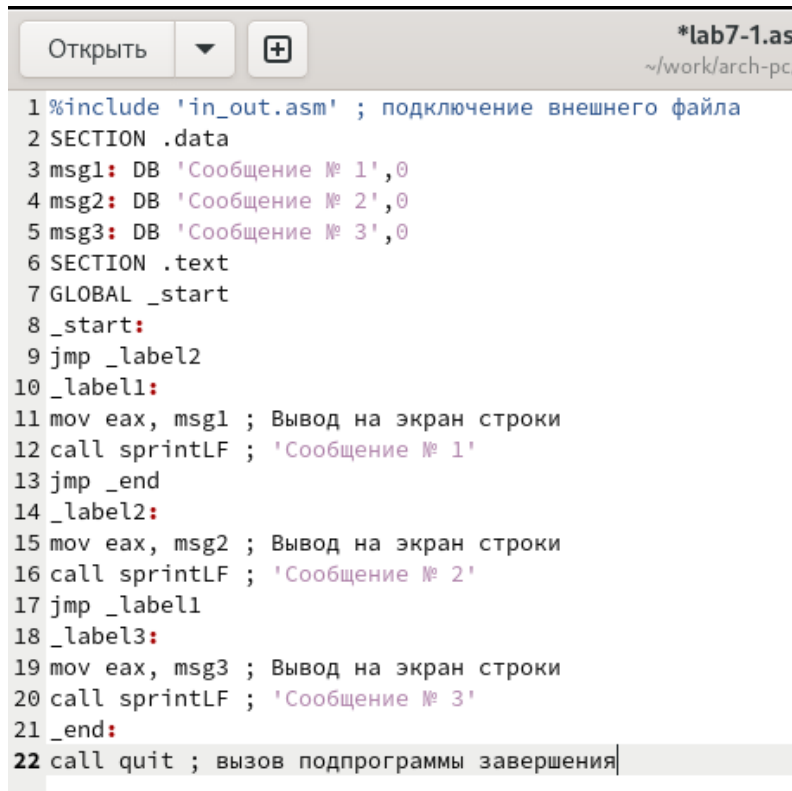
### Редактирование файла

Создадим исполняемый файл и запустим его (рис. [-@fig:003]).

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

### Создание и запуск файла

С помощью редактора gedit изменим текст программы так, чтобы сначала выводилось ‘Сообщение № 2’, потом ‘Сообщение № 1’ в соответствии с листингом 7.2 (рис. [-@fig:004]).



```
*lab7-1.asm
~/work/arch-pc

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

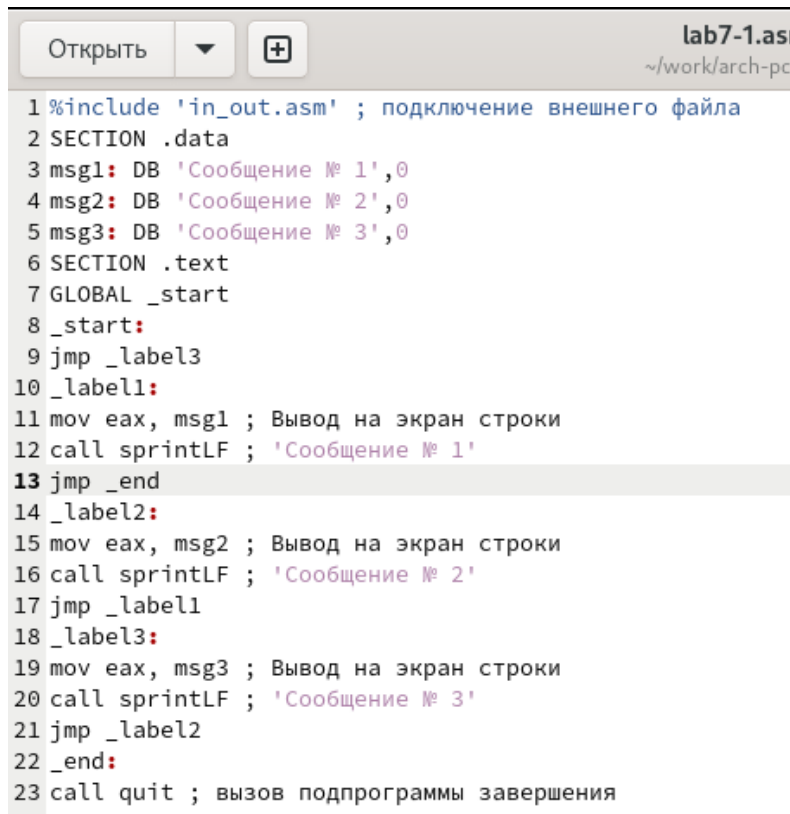
### Редактирование файла

Создадим исполняемый файл и запустим его (рис. [-@fig:005]).

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

### Создание и запуск файла

С помощью редактора gedit изменим текст программы, чтобы выводилось сначала ‘Сообщение № 3’, потом ‘Сообщение № 2’, а в конце ‘Сообщение № 1’ (рис. [-@fig:006]).



```
lab7-1.asm
~/work/arch-pc

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

### Редактирование файла

Создадим исполняемый файл и запустим его (рис. [-@fig:007]).

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

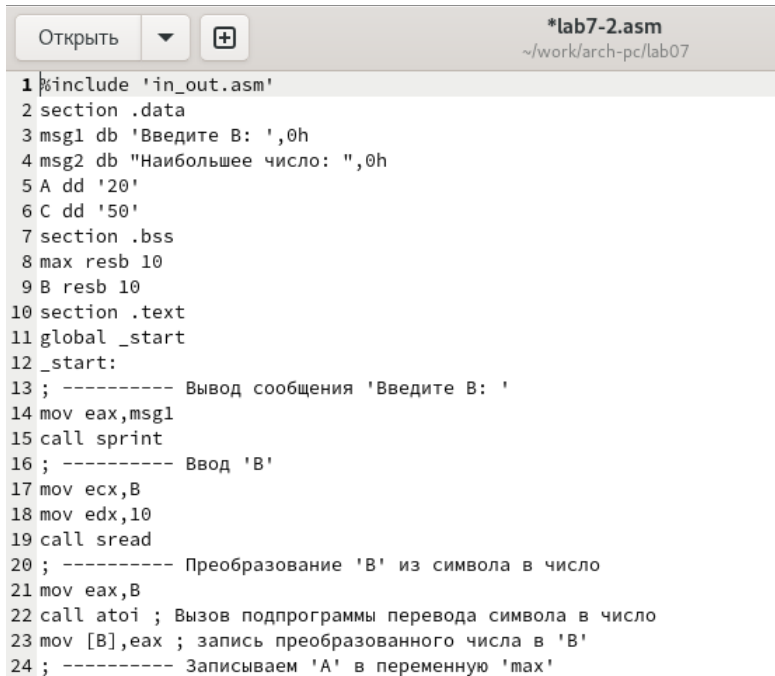
### Создание и запуск файла

Создадим файл lab7-2.asm в каталоге, созданном в начале лабораторной работы (рис. [-@fig:008]).

```
dakozina@vbox:~/work/arch-pc/lab07$ touch lab7-2.asm
```

### Создание файла

С помощью редактора gedit изменим текст программы в соответствии с листингом 7.3 (рис. [-@fig:009]).



```
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 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 mov eax,msg1
15 call sprint
16 ; ----- Ввод 'B'
17 mov ecx,B
18 mov edx,10
19 call sread
20 ; ----- Преобразование 'B' из символа в число
21 mov eax,B
22 call atoi ; Вызов подпрограммы перевода символа в число
23 mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
```

### Редактирование файла

Создадим исполняемый файл и запустим его(рис. [-@fig:010]).

```
dakozina@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
dakozina@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-2.o
dakozina@vbox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 65
Наибольшее число: 65
dakozina@vbox:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 44
Наибольшее число: 50
```

### Создание и запуск файла

## Изучение структуры файла листинга

С помощью команды `nasm` и ключа `-l` создадим файл листинга для программы из файла `lab7-2.asm` (рис. [-@fig:011]).

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

### Создание файла листинга

Откроем файл листинга `lab7-2.lst` в редакторе `mcedit`. Внимательно ознакомимся с его форматом и содержанием, объясним содержимое трех строк файла.

1. Эта строка находится на 21 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` (рис. [-@fig:012]).



```
21 00000101 B8[0A000000]          mov eax,B
```

#### Строка 21

- Эта строка находится на 35 месте, ее адрес “00000130”, Машинный код - E867FFFFFF, а call atoi - исходный текст программы, означающий что символ лежащий в строке выше переводится в число (рис. [-@fig:013]).

```
35 00000130 E867FFFFFF          call atoi ;
```

#### Строка 35

- Эта строка находится на 47 месте, ее адрес “00000163”, Машинный код - A1[00000000], а mov eax,[max] - исходный текст программы, означающий что число хранившееся в переменной max записывается в регистр eax (рис. [-@fig:014]).

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

#### Строка 47

С помощью редактора gedit откроем файл с программой lab7-2.asm и в строке ‘mov eax, max’ удалим ‘max’ (рис. [-@fig:015]).

```
34 mov eax
```

#### Удаление операнда

Выполним трансляцию с получением файла листинга (рис. [-@fig:016]).

```
dakozina@vbox:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:34: error: invalid combination of opcode and operands
```

#### Трансляция файла

В файле листинга показывает где именно ошибка и с чем она связана (рис. [-@fig:017]).

```
34 ***** error: invalid combination of opcode and operands
```

#### Ошибка в файле листинга

# Выполнение заданий для самостоятельной работы

1. Создадим файл lab7-3.asm (рис. [-@fig:017]).

```
dakozina@vbox:~/work/arch-pc/lab07$ touch lab7-3.asm
```

## Создание файла

В редакторе gedit напишем программу нахождения наименьшей из 3 целочисленных переменных (рис. [-@fig:017]).

```
1 %include 'in_out.asm'
2
3 SECTION .data
4 A1 DB 'Введите число A: ',0h
5 B1 DB 'Введите число B: ',0h
6 C1 DB 'Введите число C: ',0h
7 otv DB 'Наименьшее число: ',0h
8 SECTION .bss
9 min RESB 20
10 A RESB 20
11 B RESB 20
12 C RESB 20
13
14 SECTION .text
15 GLOBAL _start
16 _start:
17
18 mov eax,A1
19 call sprint
20
21 mov ecx,A
22 mov edx,20
23 call sread
24
25 mov eax, A
26 call atoi
27 mov [A],eax
28
29 xor eax,eax
30
31 mov eax,B1
```

## Редактирование файла

Создадим исполняемый файл и запустим его (рис. [-@fig:017]). Мой вариант 12, я ввожу значения a, b, c в соответствии с этим вариантом.

```
dakozina@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
dakozina@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
dakozina@vbox:~/work/arch-pc/lab07$ ./lab7-3
Введите число A: 99
Введите число B: 29
Введите число C: 26
Наименьшее число: 26
```

## Создание и запуск файла

## Листинг программы

```
%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

xor eax,eax

mov eax,B1
call sprint

mov ecx,B
mov edx,20
call sread

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

xor eax,eax

mov ecx, [A]
mov [min],ecx
mov ecx,[min]
```

```
cmp ecx,[B]
jl check_C
mov ecx,[B]
mov [min],ecx
```

check\_C:

```
mov eax,C1
call sprint
```

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

```
mov eax,C
call atoi
mov [C],eax
```

```
xor eax,eax
```

```
mov ecx,[min]
cmp ecx,[C]
jl fin
mov ecx,[C]
mov [min],ecx
```

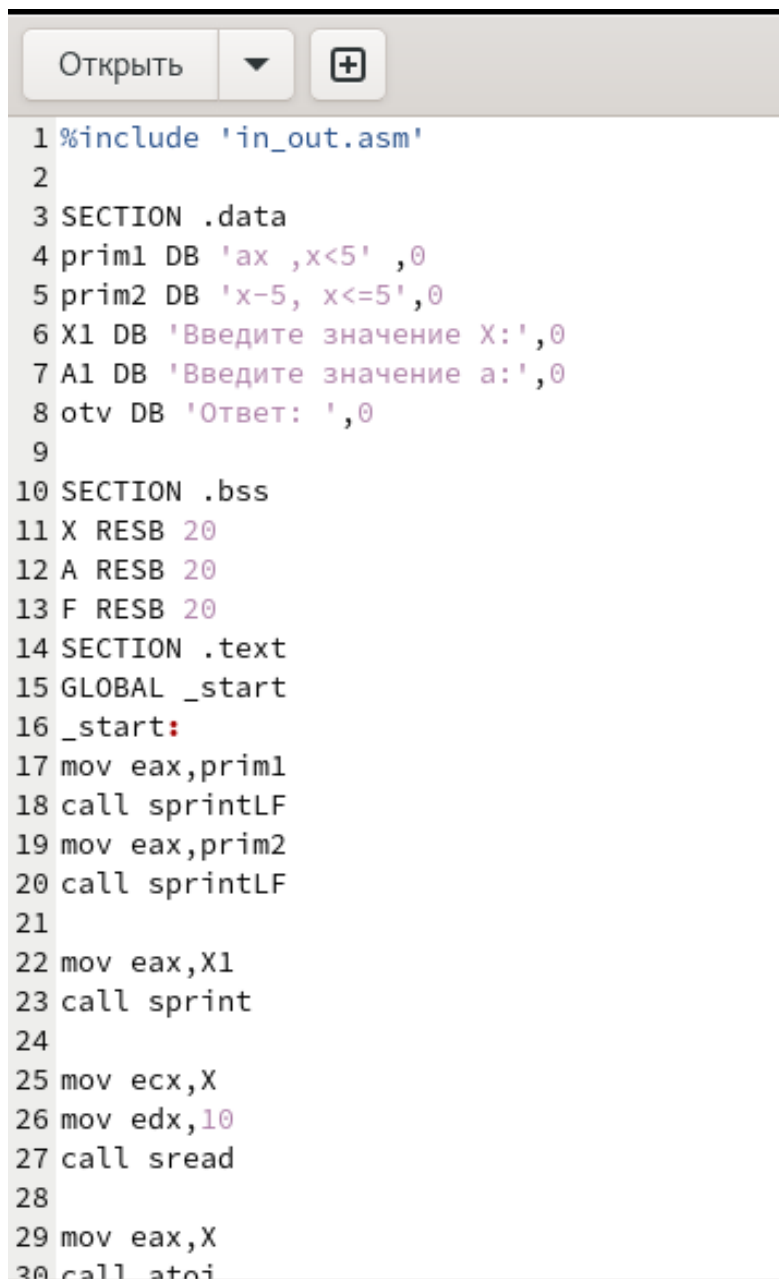
```
fin:
mov eax, otv
call sprint
mov eax,[min]
call iprintLF
call quit
```

2. Создадим файл lab7-4.asm (рис. [-@fig:017]).

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

### *Создание файла*

В редакторе gedit напишем программу для решения функции варианта 12 (рис. [-@fig:017]).



```
1 %include 'in_out.asm'
2
3 SECTION .data
4 prim1 DB 'ax ,x<5' ,0
5 prim2 DB 'x-5, x<=5',0
6 X1 DB 'Введите значение X:',0
7 A1 DB 'Введите значение a:',0
8 otv DB 'Ответ: ',0
9
10 SECTION .bss
11 X RESB 20
12 A RESB 20
13 F RESB 20
14 SECTION .text
15 GLOBAL _start
16 _start:
17 mov eax,prim1
18 call sprintf
19 mov eax,prim2
20 call sprintf
21
22 mov eax,X1
23 call sprintf
24
25 mov ecx,X
26 mov edx,10
27 call sread
28
29 mov eax,X
30 call atoi
```

*Редактирование файла*

Создадим исполняемый файл и запустим его (рис. [-@fig:017]).

```

dakoziina@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
dakoziina@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
dakoziina@vbox:~/work/arch-pc/lab07$ ./lab7-4
ax ,x<5
x-5, x<=5
Введите значение X:3
Введите значение a:7
Ответ: 21
dakoziina@vbox:~/work/arch-pc/lab07$ ./lab7-4
ax ,x<5
x-5, x<=5
Введите значение X:6
Введите значение a:4
Ответ: 1

```

*Создание и запуск файла*

## **Листинг программы**

```

#include 'in_out.asm'

SECTION .data
prim1 DB 'ax ,x<5' ,0
prim2 DB 'x-5, x<=5',0
X1 DB 'Введите значение X:',0
A1 DB 'Введите значение a:',0
otv DB 'Ответ: ',0

SECTION .bss
X RESB 20
A RESB 20
F RESB 20
SECTION .text
GLOBAL _start
_start:
mov eax,prim1
call sprintLF
mov eax,prim2
call sprintLF

mov eax,X1
call sprint

mov ecx,X
mov edx,10
call sread

mov eax,X
call atoi
mov [X],eax

mov eax,A1
call sprint

```

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

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

```
mov ecx,[X]
mov [F],ecx
mov eax, [X]
cmp ecx,5
jl check_or
mov edx,5
sub ecx,edx
mov [F],ecx
jmp fin
```

```
check_or:
mov eax,[A]
mov ebx,[X]
mul ebx
mov [F],eax
```

```
fin:
mov eax,otv
call sprint
mov eax,[F]
call iprintLF
call quit
```

## Вывод

В ходе лабораторной работы мы изучили команды условного и безусловного переходов. Приобрели навыки написания программ с использованием переходов. Познакомились с назначением и структурой файла листинга.