

# **Лабораторная работа №7**

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

Демин Даниил

# Содержание

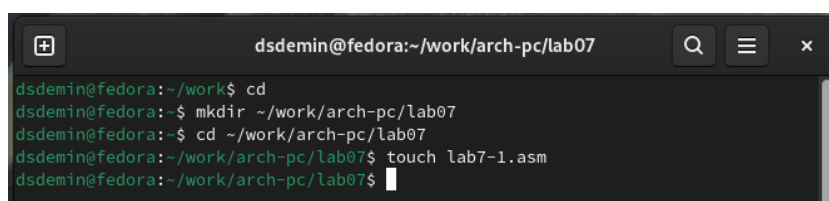
<b>1</b>	<b>Цель работы</b>	<b>3</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>4</b>
<b>3</b>	<b>Выполнение самостоятельной работы</b>	<b>15</b>
<b>4</b>	<b>Выводы</b>	<b>21</b>

# 1 Цель работы

Целью работы является приобретение практических навыков работы и изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. А так же знакомство с назначением и структурой файла листинга.

## 2 Выполнение лабораторной работы

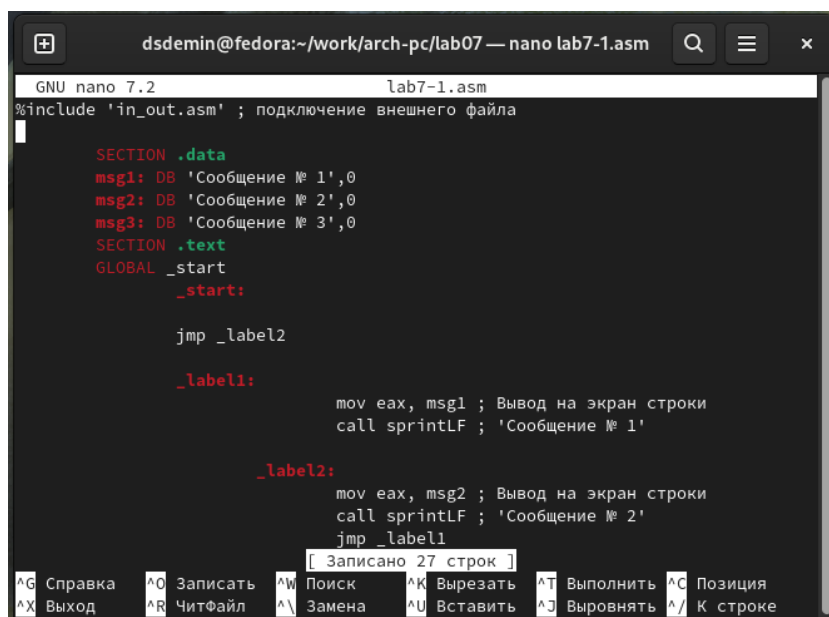
Создал и перешел в директорию для лабораторной работы и сделал файл lab7-1.asm (рис. 2.1).



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

Рис. 2.1: Директория lab07

Переписал код из листинга (рис. 2.2).



```
GNU nano 7.2 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'

_label2:
    mov eax, msg2 ; Вывод на экран строки
    call sprintLF ; 'Сообщение № 2'
    jmp _label1

[ Записано 27 строк ]
^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Позиция
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^J Выводить ^_ К строке
```

Рис. 2.2: Листинг кода

Листинг кода

```
%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
```

Создал и запустил исполняемый файл (рис. 2.3).

```

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

```

Рис. 2.3: Результат выполнения программы

Переписал код в файле с листинга (рис. 2.4).

```

_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: Обновленный код

Листинг кода

```
%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

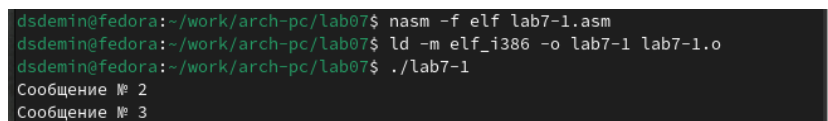
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'

_end:
call quit ; вызов подпрограммы завершения

```

Создал и запустил исполняемый файл. (рис. 2.5).



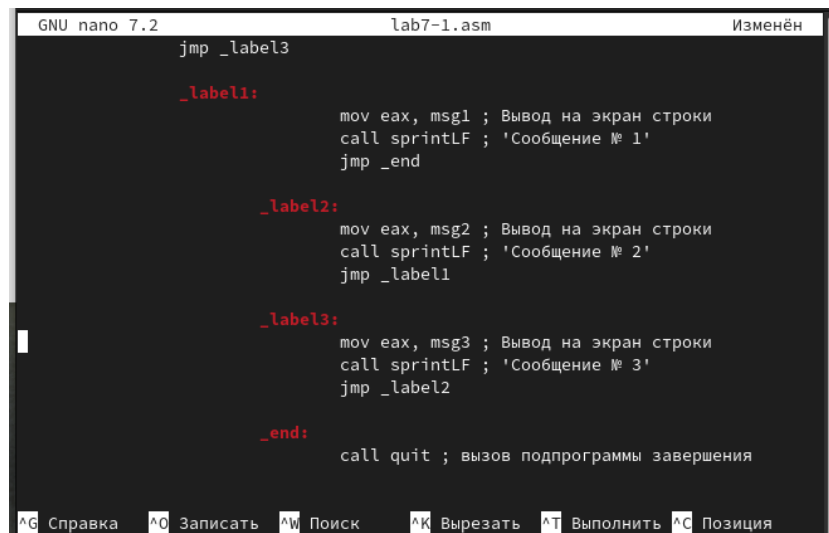
```

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

```

Рис. 2.5: Результат выполнения программы

Переписал код так, что бы он выводил сообщения в необходимом порядке (рис. 2.6).



```
GNU nano 7.2      lab7-1.asm      Изменён
jmp _label3

_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'
    jmp _label2

_end:
    call quit ; вызов подпрограммы завершения

^G Справка  ^O Записать  ^W Поиск    ^K Вырезать  ^T Выполнить ^C Позиция
```

Рис. 2.6: Листинг кода

Листинг кода 7.2.2:

`%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 sprintLF ; 'Сообщение № 1'`

`jmp _end`



```

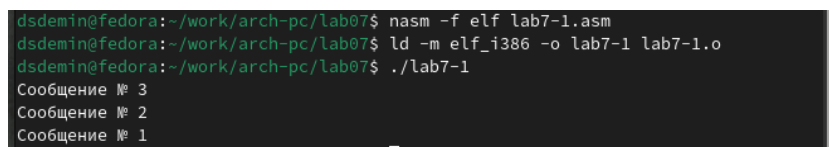
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'
jmp _label1

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'
jmp _label2

_end:
call quit ; вызов подпрограммы завершения

```

Создал и запустил исполняемый файл. (рис. 2.7).



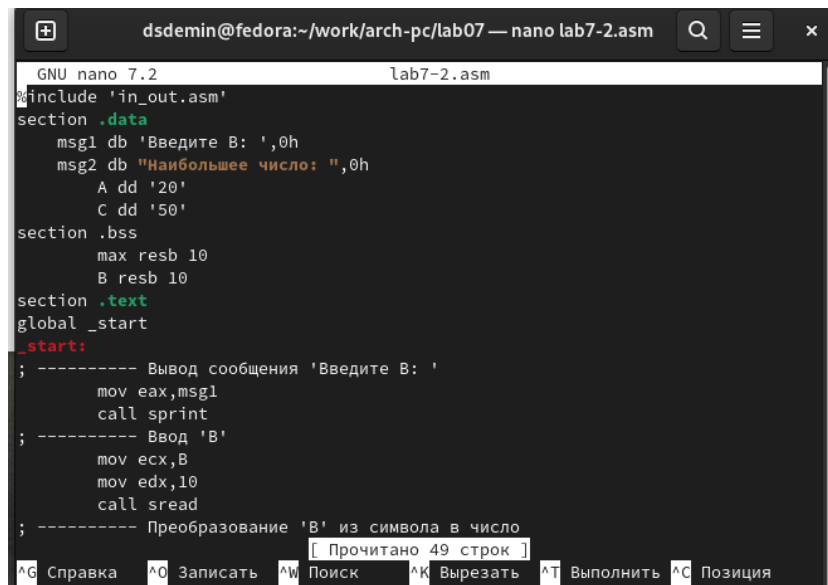
```

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

```

Рис. 2.7: Результат выполнения программы

Переписал в него код с листинга в новый файл (рис. 2.8).



```
GNU nano 7.2 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
[ Прочитано 49 строк ]
^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Позиция
```

Рис. 2.8: Листинг кода

#### Листинг кода

```
%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'
jg check_B ; если 'A>C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [max],ecx ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в число

check_B:
mov eax,max
call atoi ; Вызов подпрограммы перевода символа в число
mov [max],eax ; запись преобразованного числа в 'max'
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
jg fin ; если 'max(A,C)>B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'

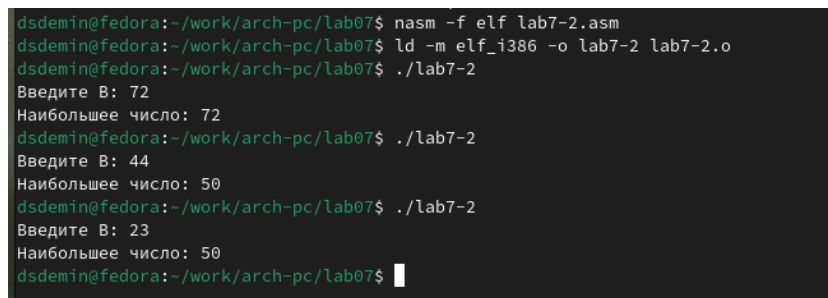
```

```

mov [max],ecx
; ----- Вывод результата
fin:
mov eax, msg2
call sprint ; Вывод сообщения 'Наибольшее число: '
mov eax,[max]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход

```

Создал и запустил исполняемый файл. Проверил несколькими числами (рис. 2.9).



```

dsdemin@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
dsdemin@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
dsdemin@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 72
Наибольшее число: 72
dsdemin@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 44
Наибольшее число: 50
dsdemin@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 23
Наибольшее число: 50
dsdemin@fedora:~/work/arch-pc/lab07$

```

Рис. 2.9: Результат выполнения программы

Создал листинг кода lab7-2.lst при помощи команды nasm. помимо основной программы помимо листинга основной программы выше находится код из in\_out.asm (рис. 2.10).

```

dsdemin@fedora:~/work/arch-pc/lab07 — mcedit lab7-2.lst
lab7-2.lst  [----]  0 L:[204+21 225/225] *(14497/14497b) <EOF>  [*] [X]
29 00000122 7F0C <----->jg check_B ; если 'A>C', то пере
30 00000124 8B0D[39000000] <----->mov ecx,[C] ; иначе 'ecx = C'
31 0000012A 890D[00000000] <----->mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' и
33 check_B:
34 00000130 B8[00000000] <----->mov eax,max
35 00000135 E862FFFFFF <----->call atoi ; Вызов подпрограммы п
36 0000013A A3[00000000] <----->mov [max],eax ; запись преобразо
37 ; ----- Сравниваем 'max(A,C)' и 'B'
38 0000013F 8B0D[00000000] <----->mov ecx,[max]
39 00000145 3B0D[0A000000] <----->cmp ecx,[B] ; Сравниваем 'max(A,
40 0000014B 7F0C <----->jg fin ; если 'max(A,C)>B', то п
41 0000014D 8B0D[0A000000] <----->mov ecx,[B] ; иначе 'ecx = B'
42 00000153 890D[00000000] <----->mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 00000159 B8[13000000] <----->mov eax, msg2
46 0000015E E8ACFFFFFF <----->call sprint ; Вывод сообщения 'H
47 00000163 A1[00000000] <----->mov eax,[max]
48 00000168 E819FFFFFF <----->call iprintLF ; Вывод 'max(A,B,C
49 0000016D E869FFFFFF <----->call quit ; Выход
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ть 7Поиск 8Уда-ть 9МенюМС10Выход

```

Рис. 2.10: Листинг кода

Выбрал набор команд в которой есть несколько операнд (рис. 2.11).

```

; ----- Преобразование 'max(A,C)' из символа в число
check_B:
    mov eax,max
    call atoi ; Вызов подпрограммы перевода символа в число
    mov [max],eax ; запись преобразованного числа в 'max'
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)

```

Рис. 2.11: Листинг кода

Удалил один из операндов (рис. 2.12).

```

; ----- Преобразование 'max(A,C)' из символа в число
check_B:
    mov eax,max
    call atoi ; Вызов подпрограммы перевода символа в число
    ;mov [max],eax ; запись преобразованного числа в 'max'
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)

```

Рис. 2.12: Обновленный листинг

Создал исполняемый файл и запустил его (рис. 2.13).

```

dsdemin@fedora:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
dsdemin@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
dsdemin@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 23
Наибольшее число: 12341

```

Рис. 2.13: Результат выполнения программы

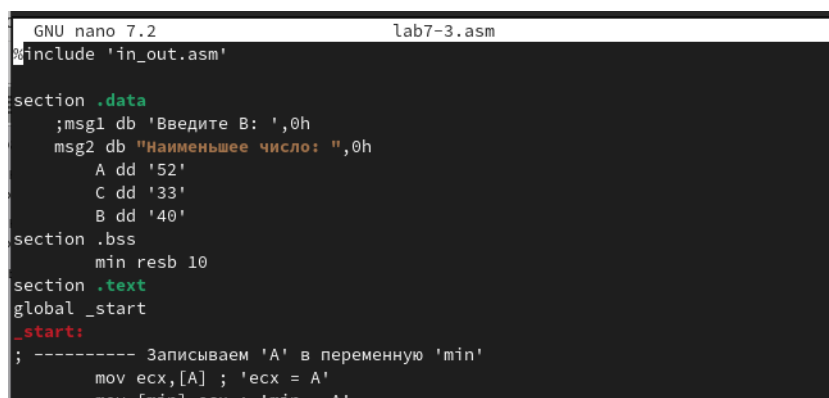
При создании появились объектный, исполняемый и файл листинга (рис. 2.14).

```
Наибольшее число: 12341
dsdemin@fedora:~/work/arch-pc/lab07$ ls
in_out.asm  lab7-1.asm  lab7-2  lab7-2.lst
lab7-1      lab7-1.o    lab7-2.asm  lab7-2.o
dsdemin@fedora:~/work/arch-pc/lab07$
```

Рис. 2.14: Созданные файлы

## 3 Выполнение самостоятельной работы

Создал и написал код в файле lab7-4.asm для нахождения наименьшей из 3 целочисленных переменных a,b среди введенных чисел, соответствующих варианту 8. (рис. 3.1).



```
GNU nano 7.2 lab7-3.asm
#include 'in_out.asm'

section .data
;msg1 db 'Введите B: ',0h
msg2 db "Наименьшее число: ",0h
A dd '52'
C dd '33'
B dd '40'
section .bss
min resb 10
section .text
global _start
_start:
; ----- Записываем 'A' в переменную 'min'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'min = A'
```

Рис. 3.1: Листинг кода

Листинг кода

```
%include 'in_out.asm'
```

```
section .data
```

```
    ;msg1 db 'Введите B: ',0h
```

```
    msg2 db "Наименьшее число: ",0h
```

```
    A dd '52'
```

```
    C dd '33'ы
```

```

        B dd '40'
section .bss
        min resb 10
section .text
global _start
_start:
; ----- Записываем 'A' в переменную 'min'
        mov ecx,[A] ; 'ecx = A'
        mov [min],ecx ; 'min = A'
; ----- Сравниваем 'A' и 'C' (как символы)
        cmp [C],ecx ; 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 ; 'max = C'
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
        mov eax,min
        call atoi ; Вызов подпрограммы перевода символа в число
        mov [min],eax ; запись преобразованного числа в 'min'
; ----- Сравниваем 'min(A,C)' и 'B' (как числа)
        mov ecx,[min]
        cmp [B],ecx ; Сравниваем 'max(A,C)' и 'B'
        jg fin ; если 'min(A,C)<B', то переход на 'fin',
        mov ecx,[B] ; иначе 'ecx = B'
        mov [min],ecx
; ----- Вывод результата
fin:
        mov eax, msg2
        call sprint ; Вывод сообщения 'Наименьшее число: '

```



```

mov eax,[min]
call iprintLF ; Вывод 'min(A,B,C)'
call quit ; Выход

```

Создал исполняемый файл и запустил его. Результат выполнения программы совпадает с ожиданием. (рис. 3.2).


```

dsdemin@fedora:~/work/arch-pc/lab07$ nasm -f elf -l lab7-3.lst lab7-3.asm
dsdemin@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
dsdemin@fedora:~/work/arch-pc/lab07$ ./lab7-3
Наименьшее число: 33

```

Рис. 3.2: Результат выполнения программы

Написал программу для подсчета результата функции  $f(x)$  в соответствии с 8 вариантом (рис. 3.3).



```

GNU nano 7.2 lab7-4.asm Изменён
#include 'in_out.asm'

section .data
    msg1 db 'Введите A: ',0h
    msg2 db 'Введите x: ',0h
    msg_res db 'Результат: ',0h

section .bss
    res resb 10
    A resb 10
    x resb 10

section .text
global _start
_start:

; ----- Вывод сообщения 'Введите A: '
mov eax,msg1
call sprint

; ----- Ввод 'A'
mov ecx,A

[ Прочитано 63 строки ]
^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить ^C Позиция
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^J Выводить ^_ К строке

```

Рис. 3.3: Листинг кода

Листинг

```

#include 'in_out.asm'

section .data

```

```

    msg1 db 'Введите A: ',0h
    msg2 db 'Введите x: ',0h
    msg_res db 'Результат: ',0h

section .bss
    res resb 10
    A resb 10
    x resb 10
section .text
global _start
_start:

; ----- Вывод сообщения 'Введите A: '
mov eax,msg1
call sprint
; ----- Ввод 'A'
mov ecx,A
mov edx,10
call sread
; ----- Преобразование 'A' из символа в число
mov eax,A
call atoi ; Вызов подпрограммы перевода символа в число
mov [A],eax ; запись преобразованного числа в 'A'

; ----- Вывод сообщения 'Введите B: '
mov eax,msg2
call sprint
; ----- Ввод 'x'
mov ecx,x

```

```

mov edx,10
call sread
; ----- Преобразование 'x' из символа в число
mov eax,x
call atoi ; Вызов подпрограммы перевода символа в число
mov [x],eax ; запись преобразованного числа в 'x'

; ----- Сравниваем 'A' и 3

mov ecx, 3
cmp [A],ecx ; Сравниваем 'A' и '3'
jg check_B ; если 'A>7', то переход на метку 'check_B',
mov eax,[A] ; иначе 'f = ax'
mul ecx
mov [res], eax
jmp fin

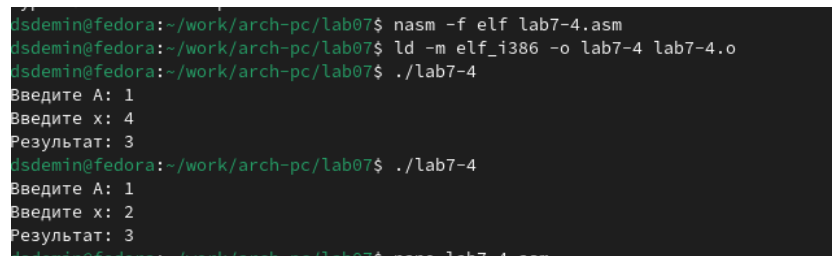
; ----- Вычитание '7 из A'
check_B:
mov eax, [A]
inc eax
mov [res], eax
jmp fin

; ----- Вывод результата
fin:
mov eax, msg_res
call sprint ; Вывод сообщения 'Результат: '
mov eax,[res]

```

```
call iprintLF ; Вывод 'f(x)='  
call quit ; Выход
```

Создал и запустил исполняемый файл. Результат совпадает с ответом, полученным аналитическим путем. (рис. 3.4).



```
dsdemin@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm  
dsdemin@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o  
dsdemin@fedora:~/work/arch-pc/lab07$ ./lab7-4  
Введите A: 1  
Введите x: 4  
Результат: 3  
dsdemin@fedora:~/work/arch-pc/lab07$ ./lab7-4  
Введите A: 1  
Введите x: 2  
Результат: 3  
dsdemin@fedora:~/work/arch-pc/lab07$
```

Рис. 3.4: Результат выполнения программы

## **4 Выводы**

Выполнив данную лабораторную работу, я обрел практические навыки работы и изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. А так же знакомство с назначением и структурой файла листинга.