

Отчёт по лабораторной работе №7

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

Юсупова Ксения Равиловна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
3.1	Реализация переходов в NASM	7
3.2	Изучение структуры файлы листинга	12
3.3	Задание для самостоятельной работы	14
4	Выводы	18

Список иллюстраций

3.1	Создали каталог с помощью команды <code>mkdir</code> и файл <code>lab7-1.asm</code> с помощью команды <code>touch</code>	7
3.2	Заполнили файл по листингу	8
3.3	Создали исполняемый файл и запустили его	8
3.4	Изменили файл	9
3.5	Создали исполняемый файл и запустили его	9
3.6	Изменили файл	10
3.7	Создали исполняемый файл и запустили его	10
3.8	Создали файл <code>lab7-2.asm</code>	10
3.9	Заполнили файл по листингу 7.2 (1 часть)	11
3.10	Заполнили файл по листингу 7.2 (2 часть)	11
3.11	Создали исполняемый файл и запустили его	11
3.12	Создали файл	12
3.13	Изучаем файл	12
3.14	Выбранные три строки кода для разбора	13
3.15	Удалили один операндум из файла	13
3.16	Транслируем файл	13
3.17	Изучили ошибку в файле листинга	14
3.18	Создали файл <code>lab7-3.asm</code>	14
3.19	Заполнили файл <code>lab7-3.asm</code> в соответствии с условием задачи (1 часть)	15
3.20	Заполнили файл <code>lab7-3.asm</code> в соответствии с условием задачи (1 часть)	15
3.21	Создали исполняемый файл и запустили его, убедившись в правильности вывода	15
3.22	Создали файл <code>lab7-3.asm</code>	16
3.23	Заполнили файл <code>lab7-4.asm</code> в соответствии с условием задачи (1 часть)	16
3.24	Заполнили файл <code>lab7-4.asm</code> в соответствии с условием задачи (2 часть)	17
3.25	Создали исполняемый файл и запустили его, убедившись в правильности вывода	17

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.

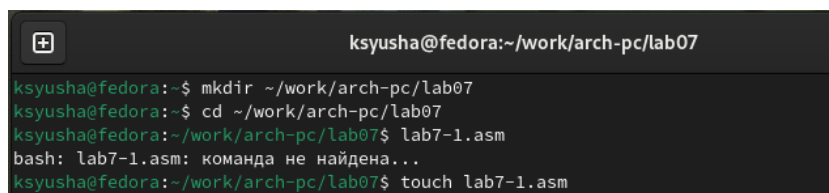
2 Задание

Написать программы для решения системы выражений.

3 Выполнение лабораторной работы

3.1 Реализация переходов в NASM

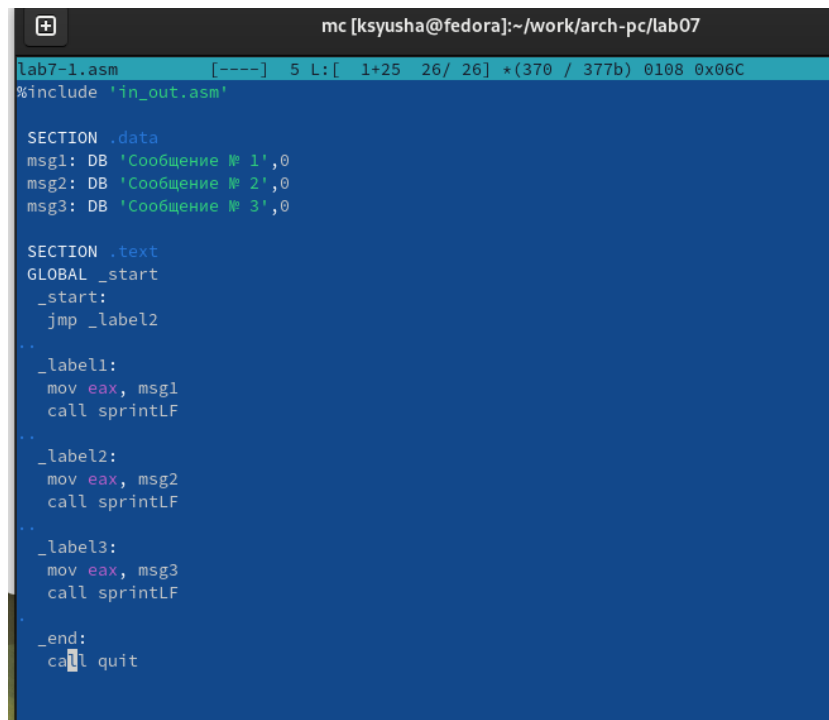
Создаем каталог для программ лабораторной работы № 7, переходим в него и создаём файл lab7-1.asm(рис. 3.1).



```
ksyusha@fedora:~/work/arch-pc/lab07
ksyusha@fedora:~$ mkdir ~/work/arch-pc/lab07
ksyusha@fedora:~$ cd ~/work/arch-pc/lab07
ksyusha@fedora:~/work/arch-pc/lab07$ lab7-1.asm
bash: lab7-1.asm: команда не найдена...
ksyusha@fedora:~/work/arch-pc/lab07$ touch lab7-1.asm
```

Рис. 3.1: Создали каталог с помощью команды mkdir и файл lab7-1.asm с помощью команды touch

Вводим в файл lab7-1.asm текст программы из листинга 7.1. В данной программе рассмотрим пример программы с использованием инструкции jmp. (рис. 3.2).



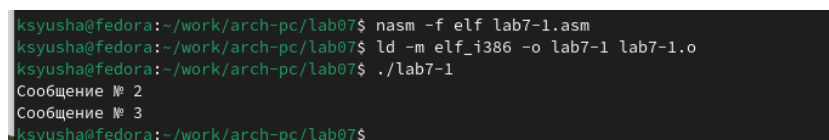
```
mc [ksyusha@fedora]:~/work/arch-pc/lab07
lab7-1.asm [----] 5 L: [ 1+25 26/ 26] *(370 / 377b) 0108 0x06C
#include 'in_out.asm'

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
    call sprintLF
;
_label2:
    mov eax, msg2
    call sprintLF
;
_label3:
    mov eax, msg3
    call sprintLF
;
_end:
    call quit
```

Рис. 3.2: Заполнили файл по листингу

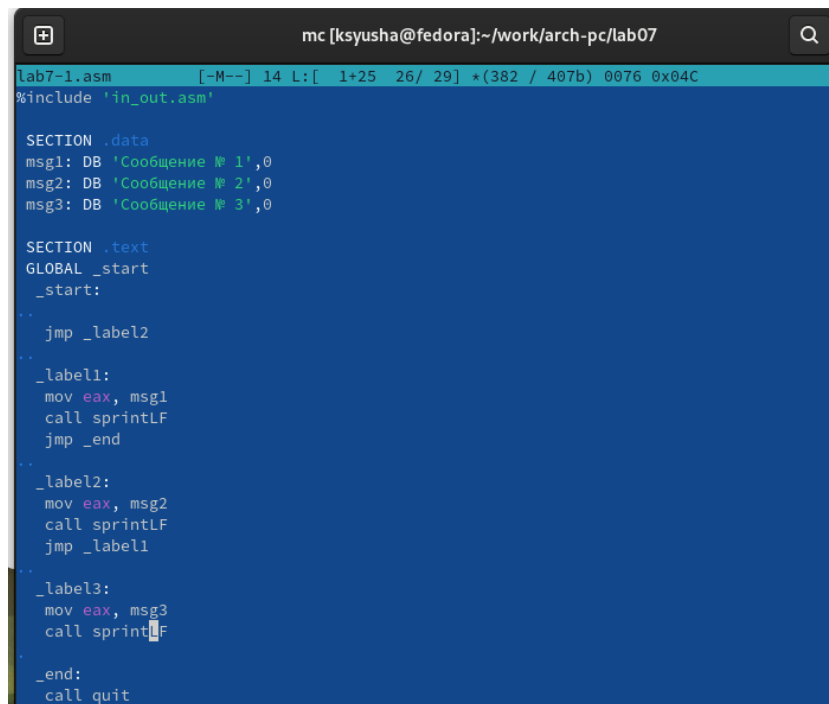
Создаем исполняемый файл и запускаем его(рис. 3.3).



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

Рис. 3.3: Создали исполняемый файл и запустили его

Далее изменяем текст программы в соответствии с листингом 7.2.(рис. 3.4).



```
lab7-1.asm [-M--] 14 L: [ 1+25 26/ 29] *(382 / 407b) 0076 0x04C
#include 'in_out.asm'

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
    call sprintLF
    jmp _end

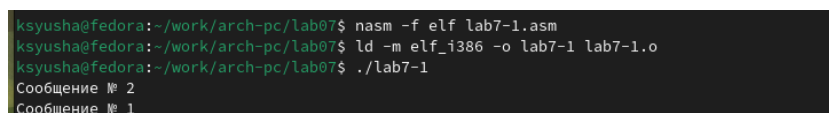
_label2:
    mov eax, msg2
    call sprintLF
    jmp _label1

_label3:
    mov eax, msg3
    call sprintLF

_end:
    call quit
```

Рис. 3.4: Изменили файл

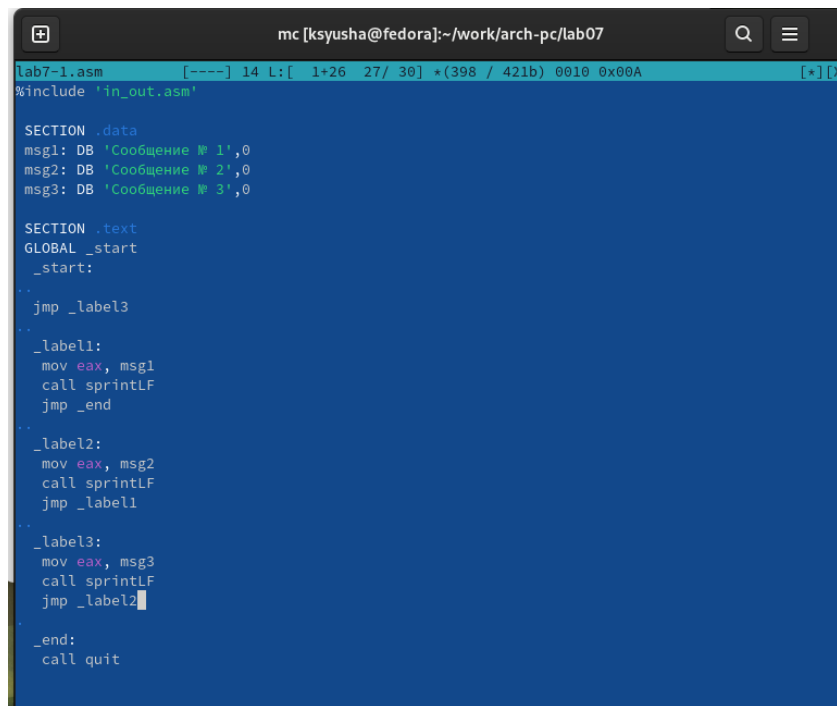
Создаем исполняемый файл и запускаем его((рис. 3.5).



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

Рис. 3.5: Создали исполняемый файл и запустили его

Снова открываем файл для редактирования и изменяем его, чтобы произошел необходимый вывод(рис. 3.6).



```
lab7-1.asm [----] 14 L: [ 1+26 27/ 30] *(398 / 421b) 0010 0x00A [*] [X]
#include 'in_out.asm'

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
call sprintLF
jmp _end

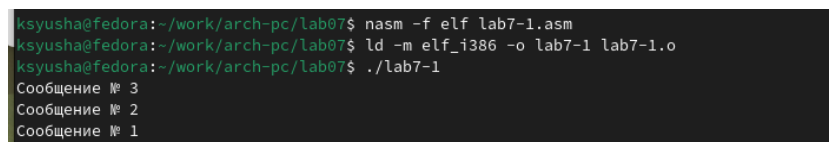
_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

Рис. 3.6: Изменили файл

Создаем исполняемый файл и запускаем его(рис. 3.7).



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

Рис. 3.7: Создали исполняемый файл и запустили его

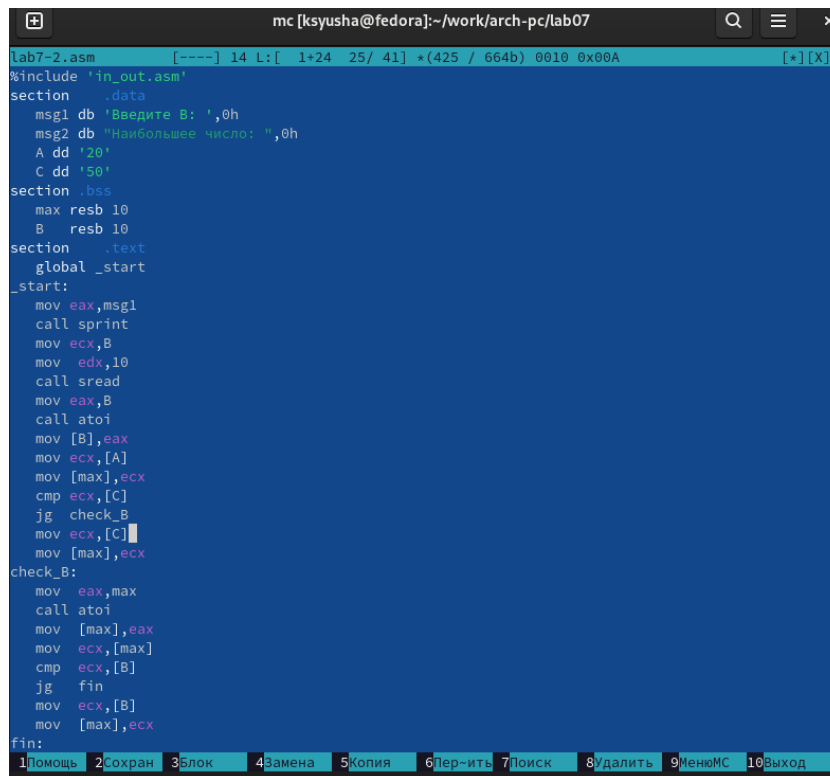
Создаем исполняемый новый файл lab7-2.asm в каталоге ~/work/arch-pc/lab07.c помощью команды touch(рис. 3.8).



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

Рис. 3.8: Создали файл lab7-2.asm

Вводим в файл lab7-2.asm текст программы из листинга 7.3.(рис. 3.9, 3.10).



```
lab7-2.asm [----] 14 L: [ 1+24 25/ 41] *(425 / 664b) 0010 0x00A [*][X]
#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:
    mov eax,msg1
    call sprint
    mov ecx,B
    mov edx,10
    call sread
    mov eax,B
    call atoi
    mov [B],eax
    mov ecx,[A]
    mov [max],ecx
    cmp ecx,[C]
    jg check_B
    mov ecx,[C]
    mov [max],ecx
check_B:
    mov eax,max
    call atoi
    mov [max],eax
    mov ecx,[max]
    cmp ecx,[B]
    jg fin
    mov ecx,[B]
    mov [max],ecx
fin:
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ить 7Поиск 8Удалить 9МенюМС 10Выход
```

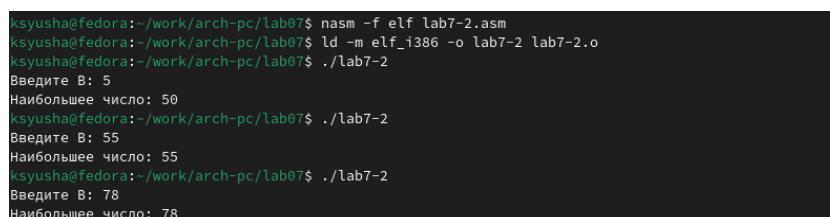
Рис. 3.9: Заполнили файл по листингу 7.2 (1 часть)



```
fin:
    mov eax, msg2
    call sprint
    mov eax,[max]
    call iprintLF
    call quit
1Помощь 2Сохран 3Блок 4Замена 5Копия 6Пер-ить 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 3.10: Заполнили файл по листингу 7.2 (2 часть)

Создаем исполняемый файл и проверяем его работу, вводя разные значения B (рис. 3.11).



```
ksyusha@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
ksyusha@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
ksyusha@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 5
Наибольшее число: 50
ksyusha@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 55
Наибольшее число: 55
ksyusha@fedora:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 78
Наибольшее число: 78
```

Рис. 3.11: Создали исполняемый файл и запустили его

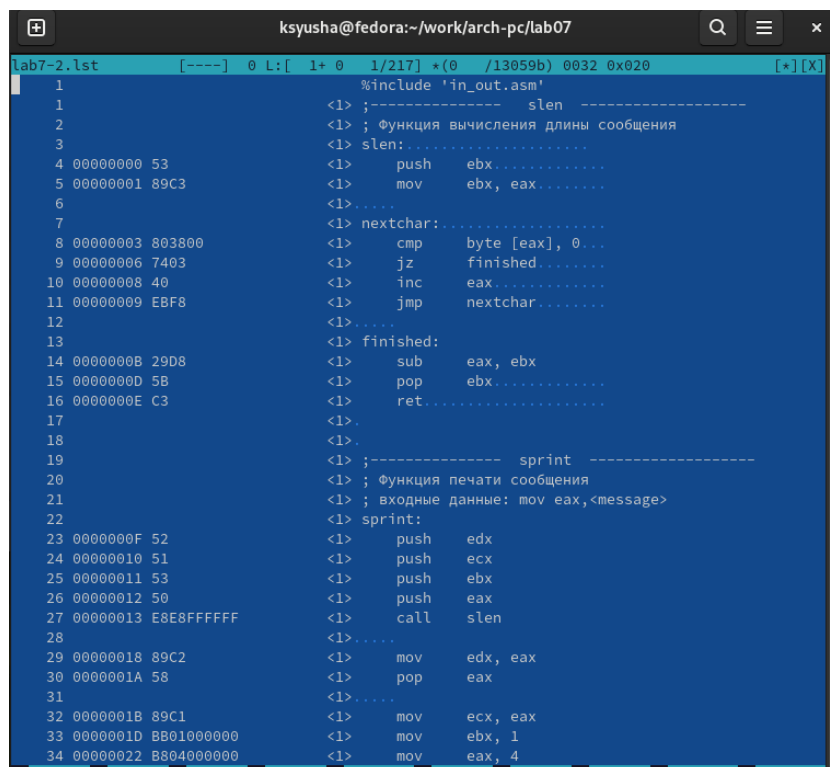
3.2 Изучение структуры файлы листинга

Создаем файл листинга для программы из файла lab7-2.asm(рис. 3.12).

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

Рис. 3.12: Создали файл

Открываем файл листинга с помощью команды mcedit и изучаем его(рис. 3.13).



```
lab7-2.lst  [----]  0 L:[ 1+ 0 1/217] *(0 /13059b) 0032 0x020  [*][X]
1  %include 'in_out.asm'
1  <1> ;----- slen -----
2  <1> ; Функция вычисления длины сообщения
3  <1> slen:-----
4  00000000 53 <1> push ebx-----
5  00000001 89C3 <1> mov ebx, eax-----
6  <1>-----
7  <1> nextchar:-----
8  00000003 803800 <1> cmp byte [eax], 0...
9  00000006 7403 <1> jz finished-----
10 00000008 40 <1> inc eax-----
11 00000009 EBF8 <1> jmp nextchar-----
12 <1>-----
13 <1> finished:-----
14 0000000B 29D8 <1> sub eax, ebx
15 0000000D 5B <1> pop ebx-----
16 0000000E C3 <1> ret-----
17 <1>-----
18 <1>-----
19 <1> ;----- sprint -----
20 <1> ; Функция печати сообщения
21 <1> ; входные данные: mov eax, <message>
22 <1> sprint:-----
23 0000000F 52 <1> push edx
24 00000010 51 <1> push ecx
25 00000011 53 <1> push ebx
26 00000012 50 <1> push eax
27 00000013 E8E8FFFFFF <1> call slen
28 <1>-----
29 00000018 89C2 <1> mov edx, eax
30 0000001A 58 <1> pop eax
31 <1>-----
32 0000001B 89C1 <1> mov ecx, eax
33 0000001D B801000000 <1> mov ebx, 1
34 00000022 B804000000 <1> mov eax, 4
```

Рис. 3.13: Изучаем файл

Внимательно ознакомились с форматом и содержимым файла. Подробно объясним содержимое трёх строк файла листинга по выбору:

Строка 65: 00000045- адрес в сегменте кода, B800000000-машинный код, mov ebx,1 - исходный текст программы, присвоение переменной ebx значения 1.

Строка 66: 0000004A- адрес в сегменте кода, B803000000-машинный код, mov eax,3 - исходный текст программы, присвоение переменной eax значения 3.

Строка 67: 0000004F- адрес в сегменте кода, CD80-машинный код, int 80h - исходный текст программы, вызов ядра. (рис. 3.14).

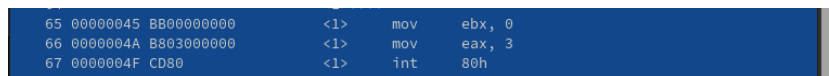


Рис. 3.14: Выбранные три строки кода для разбора

Открываем файл и удаляем один операндум(рис. 3.15).

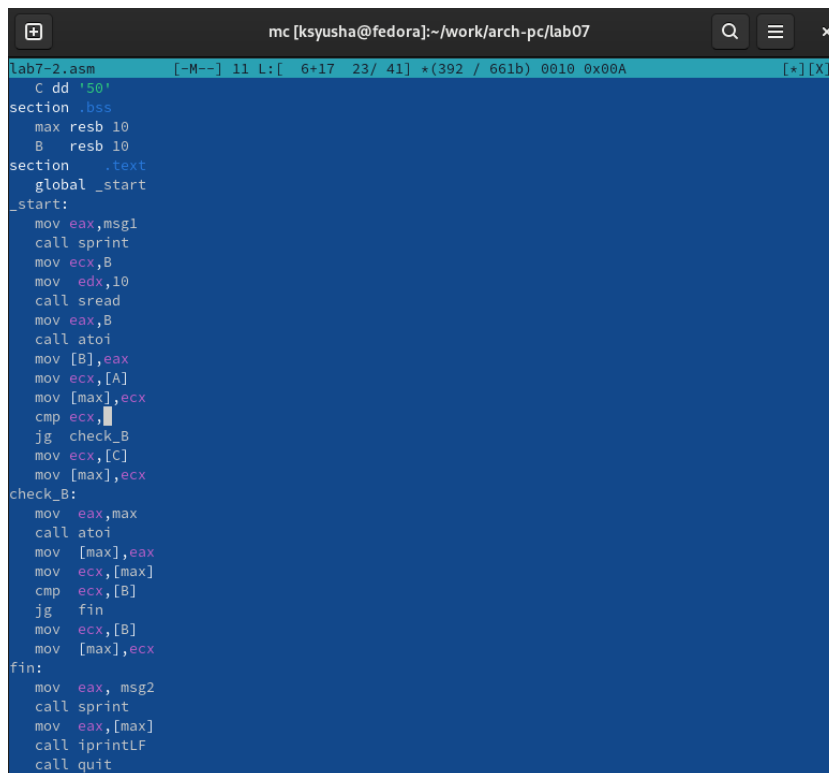


Рис. 3.15: Удалили один операндум из файла

Выполняем трансляцию с получением файла листинга и замечаем, что при трансляции файла появляется ошибка(рис. 3.16).

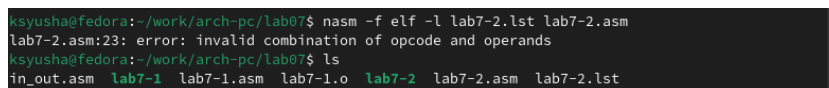
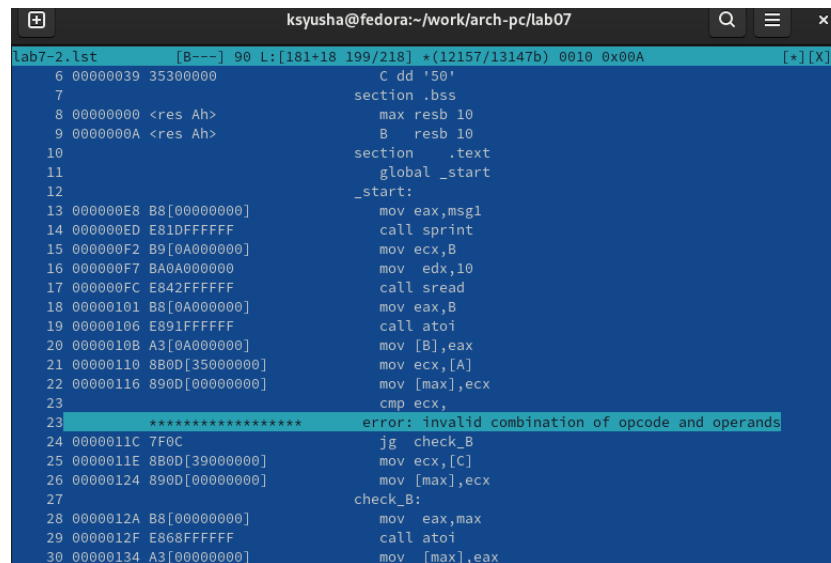


Рис. 3.16: Транслируем файл

В новом файле листинга также появилась ошибка, которая возникла при попытке трансляции файла. Выходные файлы, кроме файла листинга, не создаются(рис.

3.17).



```
ksyusha@fedora:~/work/arch-pc/lab07
lab7-2.lst [B---] 90 L:[181+18 199/218] *(12157/13147b) 0010 0x00A [*][X]
6 00000039 35300000 C dd '50'
7 section .bss
8 00000000 <res Ah> max resb 10
9 0000000A <res Ah> B resb 10
10 section .text
11 global _start
12 _start:
13 000000E8 B8[00000000] mov eax,msg1
14 000000ED E81DFFFFFF call sprint
15 000000F2 B9[0A000000] mov ecx,B
16 000000F7 BA0A000000 mov edx,10
17 000000FC E842FFFFFF call sread
18 00000101 B8[0A000000] mov eax,B
19 00000106 E891FFFFFF call atoi
20 0000010B A3[0A000000] mov [B],eax
21 00000110 8B0D[35000000] mov ecx,[A]
22 00000116 890D[00000000] mov [max],ecx
23 cmp ecx,
23 ***** error: invalid combination of opcode and operands
24 0000011C 7F0C jg check_B
25 0000011E 8B0D[39000000] mov ecx,[C]
26 00000124 890D[00000000] mov [max],ecx
27 check_B:
28 0000012A B8[00000000] mov eax,max
29 0000012F E868FFFFFF call atoi
30 00000134 A3[00000000] mov [max],eax
```

Рис. 3.17: Изучили ошибку в файле листинга

3.3 Задание для самостоятельной работы

Вариант 12

1. Напишите программу нахождения наименьшей из 3 целочисленных переменных а, b и с. (Значения переменных выбраны из табл. 7.5 в соответствии с 12 вариантом)

Создаем новый файл lab7-3.asm для выполнения 1 задания(рис. 3.18).



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

Рис. 3.18: Создали файл lab7-3.asm

Открываем файл и пишем программу, которая выберет наименьшее число из трех.(рис. 3.19, 3.20).

```

lab7-3.asm [----] 12 L: [ 1+17 18/ 41] *(327 / 671b) 0010 0x00A [*][X]
#include 'in_out.asm'
section .data
    msg1 db 'Введите B: ',0h
    msg2 db "Наименьшее значение: ",0h
    A dd '99'
    C dd '26'
section .bss
    min resb 10
    B resb 10
section .text
    global _start
_start:
    mov eax,msg1
    call sprint
    mov ecx,B
    mov edx,10
    call sread
    mov eax,B
    call atoi
    mov [B],eax
    mov ecx,[A]
    mov [min],ecx
    cmp ecx,[C]
    jl check_B
    mov ecx,[C]
    mov [min],ecx
check_B:
    mov eax,min
    call atoi
    mov [min],eax
    mov ecx,[min]
    cmp ecx,[B]
    jl fin
    mov ecx,[B]
    mov [min],ecx
fin:

```

Рис. 3.19: Заполнили файл lab7-3.asm в соответствии с условием задачи (1 часть)

```

fin:
    mov eax, msg2
    call sprint
    mov eax,[min]
    call iprintLF
    call quit

```

Рис. 3.20: Заполнили файл lab7-3.asm в соответствии с условием задачи (1 часть)

Создаем исполняемый файл и запускаем его(рис. 3.21).

```

ksyusha@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
ksyusha@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-3 lab7-3.o
ksyusha@fedora:~/work/arch-pc/lab07$ ./lab7-3
Введите B: 29
Наименьшее значение: 26

```

Рис. 3.21: Создали исполняемый файл и запустили его, убедившись в правильности вывода

2. Напишите программу, которая для введенных с клавиатуры значений x и a

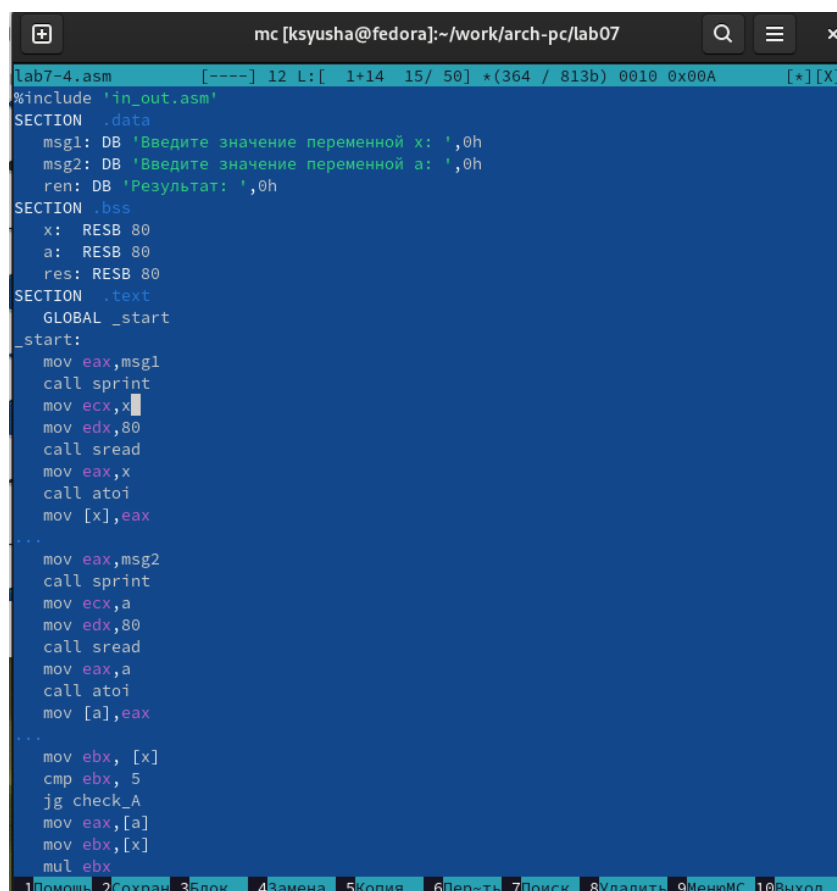
вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. (Вид функции $f(x)$ был выбран в соответствии с 12 вариантом)

Создаем новый файл lab7-4.asm для выполнения 2 задания(рис. 3.22).

```
ksyusha@fedora:~/work/arch-pc/lab07$ touch lab7-4.asm
```

Рис. 3.22: Создали файл lab7-3.asm

Открываем файл и пишем программу, которая вычислит значение функции, в зависимости от введенных данных(рис. 3.23, 3.24).



```
lab7-4.asm  [----] 12 L: [ 1+14 15/ 50] *(364 / 813b) 0010 0x00A  [*] [X]
#include 'in_out.asm'
SECTION .data
    msg1: DB 'Введите значение переменной x: ',0h
    msg2: DB 'Введите значение переменной a: ',0h
    ren: DB 'Результат: ',0h
SECTION .bss
    x: RESB 80
    a: RESB 80
    res: RESB 80
SECTION .text
    GLOBAL _start
_start:
    mov eax,msg1
    call sprint
    mov ecx,x
    mov edx,80
    call sread
    mov eax,x
    call atoi
    mov [x],eax
    ...
    mov eax,msg2
    call sprint
    mov ecx,a
    mov edx,80
    call sread
    mov eax,a
    call atoi
    mov [a],eax
    ...
    mov ebx, [x]
    cmp ebx, 5
    jg check_A
    mov eax,[a]
    mov ebx,[x]
    mul ebx
```

Рис. 3.23: Заполнили файл lab7-4.asm в соответствии с условием задачи (1 часть)


```

    mov     edx, eax
    jmp     fin
check_A:
    mov     ebx, [x]
    sub     ebx, 5
    mov     edx, ebx
    jmp     fin
fin:
    mov     eax, ren
    call    sprint
    mov     eax, edx
    call    iprintLF
    call    quit

```

1 Помощь 2 Сохран 3 Блок 4 Замена 5 Копия 6 Пер-ть 7 Поиск 8 Удалить 9 МенюМС 10 Выход

Рис. 3.24: Заполнили файл lab7-4.asm в соответствии с условием задачи (2 часть)

Создаем исполняемый файл и запускаем его (рис. 3.25)

```

ksyusha@fedora:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm
ksyusha@fedora:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-4 lab7-4.o
ksyusha@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите значение переменной x: 3
Введите значение переменной a: 7
Результат: 21
ksyusha@fedora:~/work/arch-pc/lab07$ ./lab7-4
Введите значение переменной x: 6
Введите значение переменной a: 4
Результат: 1

```

Рис. 3.25: Создали исполняемый файл и запустили его, убедившись в правильности вывода

4 Выводы

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