

# **Отчёт по лабораторной работе №8**

**Дисциплина: Архитектура компьютера**

**Батова Ирина Сергеевна, НММбд-01-22**

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Задание для самостоятельной работы</b>	<b>15</b>
<b>4</b>	<b>Выводы</b>	<b>18</b>

## Список иллюстраций

2.1	Создание необходимых для работы каталогов и файлов . . . . .	6
2.2	Ввод листинга 8.1 . . . . .	7
2.3	Запуск программы из файла 'lab8-1.asm' . . . . .	7
2.4	Ввод изменений в программу из файла 'lab8-1.asm' . . . . .	8
2.5	Запуск измененной программы из файла 'lab8-1.asm' . . . . .	9
2.6	Повторный ввод изменений в программу из файла 'lab8-1.asm' . .	10
2.7	Запуск повторно измененной программы из файла 'lab8-1.asm' .	11
2.8	Создание файла 'lab8-2.asm' . . . . .	11
2.9	Ввод листинга 8.3 . . . . .	12
2.10	Запуск программы из файла 'lab8-2.asm' . . . . .	13
2.11	Получение файла листинга программы из файла 'lab8-2.asm' . .	13
2.12	Листинг программы из файла 'lab8-2.asm' . . . . .	13
2.13	Удаление у команды 'cmp' второго операнда . . . . .	14
2.14	Получение файла листинга . . . . .	14
2.15	Файл листинг - ошибка . . . . .	14
3.1	Ввод программы в файл 'lab8-3.asm' . . . . .	15
3.2	Запуск программы из файла 'lab8-3.asm' . . . . .	16
3.3	Ввод программы в файл 'lab8-4.asm', часть 1 . . . . .	16
3.4	Ввод программы в файл 'lab8-4.asm', часть 2 . . . . .	17
3.5	Запуск программы из файла 'lab8-4.asm' . . . . .	17

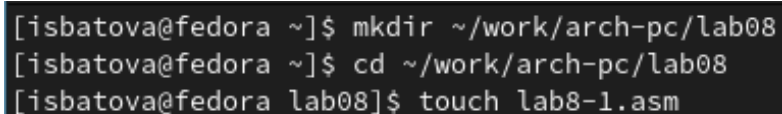
## **Список таблиц**

# 1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

## 2 Выполнение лабораторной работы

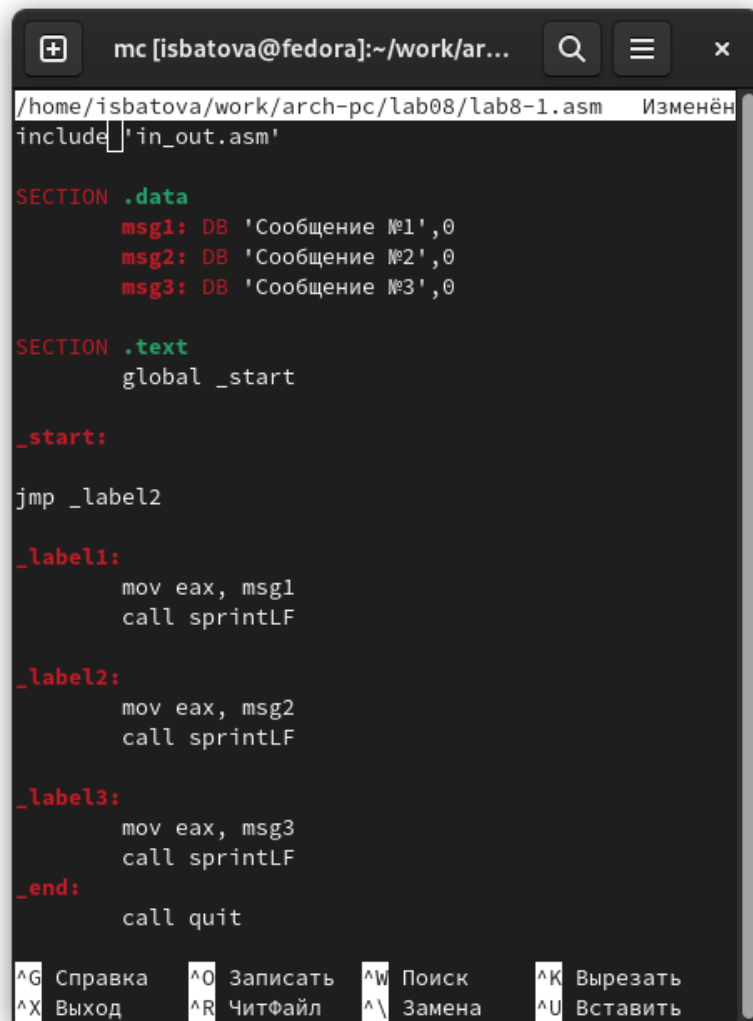
1. Создаем каталог 'lab08' с помощью команды `mkdir`, переходим в него с помощью команды `cd` и создаем в нем файл 'lab8-1.asm' с помощью команды `touch` (рис. 2.1).



```
[isbatova@fedora ~]$ mkdir ~/work/arch-pc/lab08  
[isbatova@fedora ~]$ cd ~/work/arch-pc/lab08  
[isbatova@fedora lab08]$ touch lab8-1.asm
```

Рис. 2.1: Создание необходимых для работы каталогов и файлов

2. Открываем файл 'lab8-1.asm' и вводим листинг 8.1 из лабораторной работы (рис. 2.2).



```
mc [isbatova@fedora]:~/work/ar...
/home/isbatova/work/arch-pc/lab08/lab8-1.asm Изменён
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

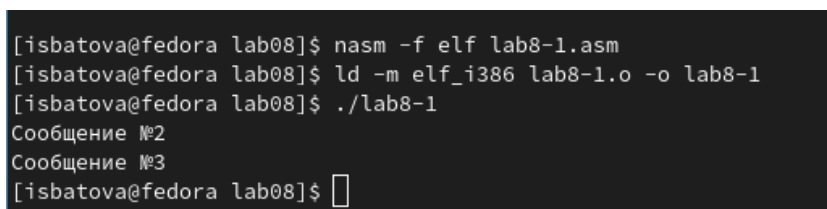
_label3:
    mov eax, msg3
    call sprintLF

_end:
    call quit

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

Рис. 2.2: Ввод листинга 8.1

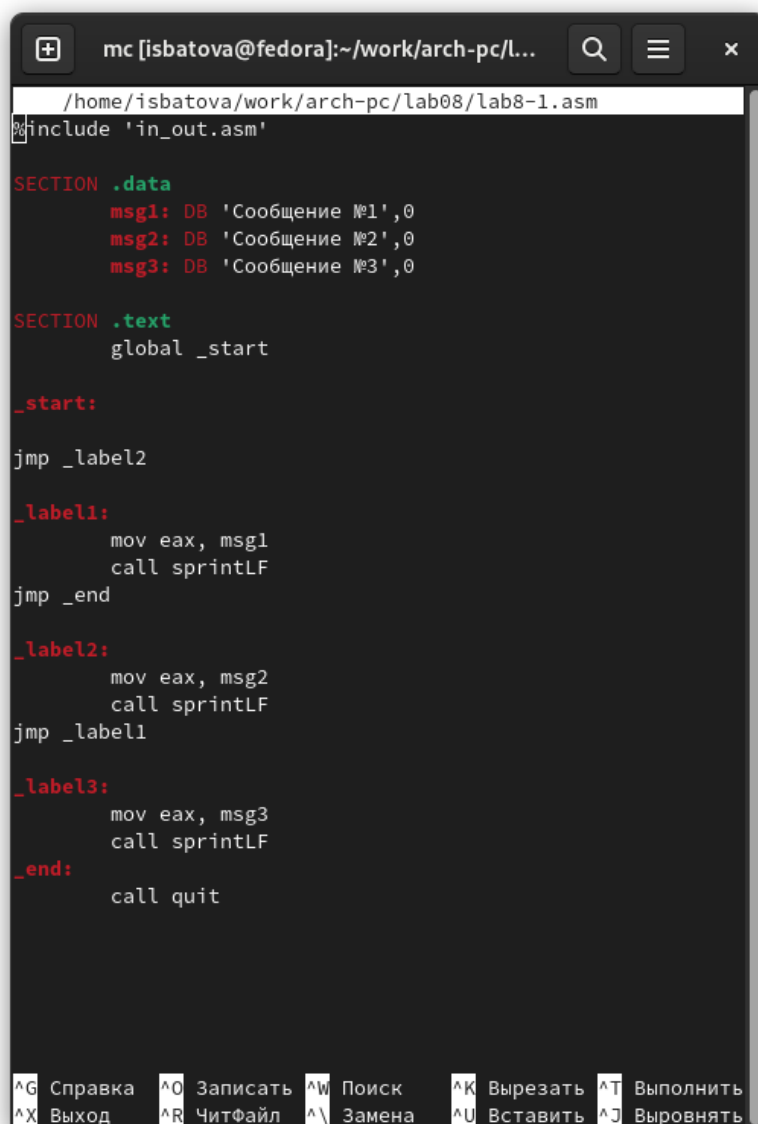
Создаем исполняемый файл и запускаем его (рис. 2.3). Программа выводит правильный результат, значит, она написана корректно.



```
[isbatova@fedora lab08]$ nasm -f elf lab8-1.asm
[isbatova@fedora lab08]$ ld -m elf_i386 lab8-1.o -o lab8-1
[isbatova@fedora lab08]$ ./lab8-1
Сообщение №2
Сообщение №3
[isbatova@fedora lab08]$
```

Рис. 2.3: Запуск программы из файла 'lab8-1.asm'

Далее вновь открываем файл 'lab8-1.asm' и редактируем его так, чтобы она выводила сначала 'Сообщение №2', а потом 'Сообщение №1' (рис. 2.4).



```
mc [isbatova@fedora]:~/work/arch-pc/l...
/home/isbatova/work/arch-pc/lab08/lab8-1.asm
%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
```

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить  
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^J Выводить

Рис. 2.4: Ввод изменений в программу из файла 'lab8-1.asm'

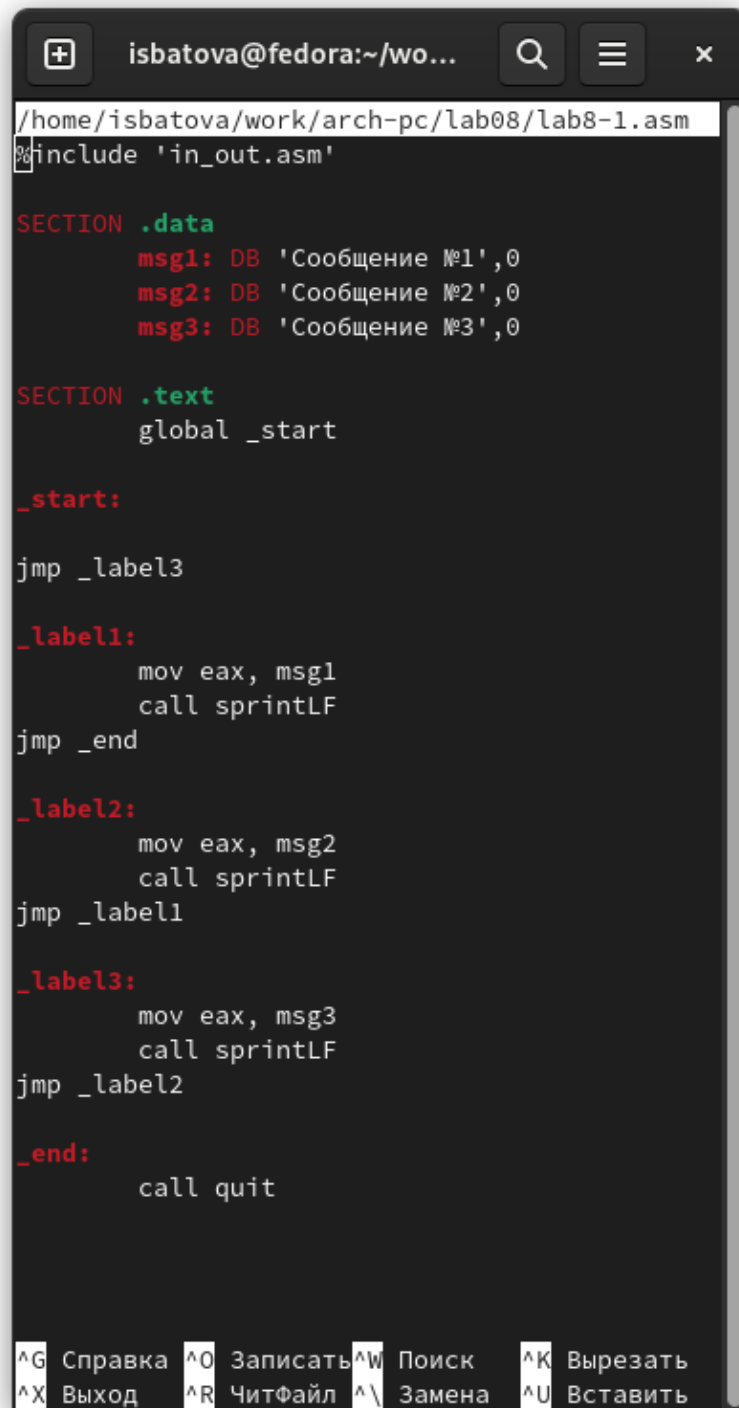
Создаем исполняемый файл и запускаем его (рис. 2.5). Программа выводит правильный результат, значит, она написана корректно.



```
[isbatova@fedora lab08]$ nasm -f elf lab8-1.asm
[isbatova@fedora lab08]$ ld -m elf_i386 lab8-1.o -o lab8-1
[isbatova@fedora lab08]$ ./lab8-1
Сообщение №2
Сообщение №1
[isbatova@fedora lab08]$
```

Рис. 2.5: Запуск измененной программы из файла 'lab8-1.asm'

Далее редактируем файл 'lab8-1.asm' так, чтобы сообщения выводились в обратной последовательности: 'Сообщение №3', 'Сообщение №2', 'Сообщение №1' (рис. 2.6).



The image shows a terminal window with a dark background. The title bar at the top reads 'isbatova@fedora:~/wo...'. The terminal content is assembly code for a file named 'lab8-1.asm'. The code includes a preprocessor directive, two section declarations (one for data and one for text), and several labels with instructions. At the bottom, there is a menu bar with keyboard shortcuts and their corresponding actions in Russian.

```
/home/isbatova/work/arch-pc/lab08/lab8-1.asm
%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
```

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

Рис. 2.6: Повторный ввод изменений в программу из файла 'lab8-1.asm'

Создаем исполняемый файл и запускаем его (рис. 2.7). Программа выводит

правильный результат, значит, она написана корректно.

```
[isbatova@fedora lab08]$ nasm -f elf lab8-1.asm
[isbatova@fedora lab08]$ ld -m elf_i386 lab8-1.o -o lab8-1
[isbatova@fedora lab08]$ ./lab8-1
Сообщение №3
Сообщение №2
Сообщение №1
```

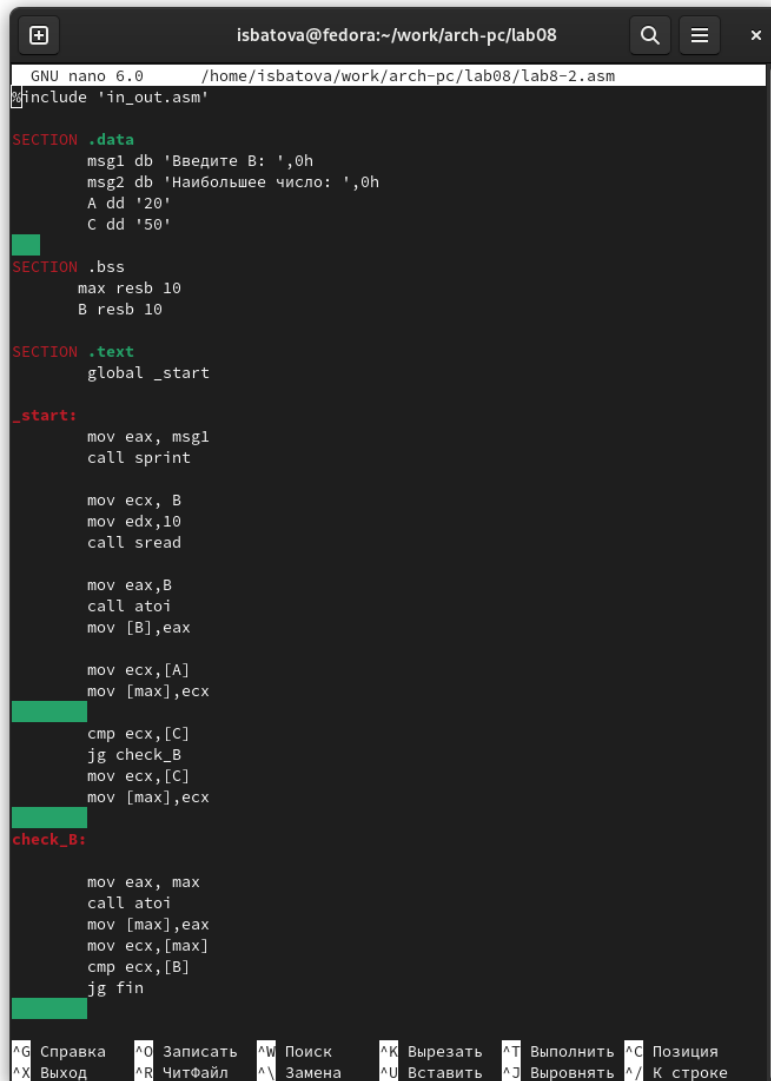
Рис. 2.7: Запуск повторно измененной программы из файла 'lab8-1.asm'

3. Создаем файл 'lab8-2.asm' с помощью команды 'touch' (рис. 2.8).

```
[isbatova@fedora lab08]$ touch lab8-2.asm
[isbatova@fedora lab08]$
```

Рис. 2.8: Создание файла 'lab8-2.asm'

Открываем файл 'lab8-2.asm' и вводим листинг 8.3 из лабораторной работы (рис. 2.9).



```
GNU nano 6.0 /home/isbatova/work/arch-pc/lab08/lab8-2.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:
    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
```

Рис. 2.9: Ввод листинга 8.3

Создаем исполняемый файл и запускаем его (рис. 2.10). Проверяем работу программы, вводя несколько чисел - программа выводит правильный результат, значит, она написана корректно.

```

[isbatova@fedora lab08]$ nasm -f elf lab8-2.asm
[isbatova@fedora lab08]$ ld -m elf_i386 lab8-2.o -o lab8-2
[isbatova@fedora lab08]$ ./lab8-2
Введите B: 15
Наибольшее число: 50
[isbatova@fedora lab08]$ ./lab8-2
Введите B: 34
Наибольшее число: 50
[isbatova@fedora lab08]$ ./lab8-2
Введите B: 87
Наибольшее число: 87

```

Рис. 2.10: Запуск программы из файла 'lab8-2.asm'

4. Далее нам нужно получить файл листинга. Для этого вводим команду 'nasm -f elf -l lab8-2.lst lab8-2.asm' (рис. 2.11). Далее открываем файл листинга с помощью редактора mcedit (рис. 2.12).

```

[isbatova@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
[isbatova@fedora lab08]$ mcedit lab8-2.lst

```

Рис. 2.11: Получение файла листинга программы из файла 'lab8-2.asm'

```

 6 00000036 32300000          A dd '20'
 7 0000003A 35300000          C dd '50'
 8 .....
 9                               SECTION .bss
10 00000000 <res Ah>          max resb 10
11 0000000A <res Ah>          B resb 10
12 .....
13                               SECTION .text
14                               <----->global _start
15 .....
16                               _start:
17 000000E8 B8[00000000]      mov eax, msg1
18 000000ED E81DFFFFFF      call sprint
19 .....
20 000000F2 B9[0A000000]      mov ecx, B
21 000000F7 BA0A000000      mov edx, 10
22 000000FC E842FFFFFF      call sread
23 .....
24 00000101 B8[0A000000]      mov eax, B
25 00000106 E891FFFFFF      call atoi
26 0000010B A3[0A000000]      mov [B], eax
27 .....
28 00000110 8B0D[36000000]      mov ecx, [A]
29 00000116 890D[00000000]      mov [max], ecx
30 .....

```

Рис. 2.12: Листинг программы из файла 'lab8-2.asm'

Рассмотрим строки 20, 21, 22.

1. 20, 21, 22 - номер строки.
2. 000000F2, 000000F7, 000000FC - это адрес строки.
3. B9[0A000000], BA0A000000, E842FFFFFF - это машинный код.
4. 'mov ecx,B', 'call atoi', mov [B], eax - это исходный текст программы.

Далее открываем файл 'lab8-2.asm' и убираем у команды 'cmp' второй операнд (рис. 2.13).

```
cmp ecx
jg check_B
mov ecx,[C]
mov [max],ecx
```

Рис. 2.13: Удаление у команды 'cmp' второго операнда

Выполняем трансляцию с получением файла листинга (рис. 2.14). Программа выводит ошибку, при этом файл листинга создается. Если открыть его, мы увидим, что в файле листинга также обозначена ошибка отсутствия одного операнда (рис. 2.15).

```
[isbatova@fedora lab08]$ nasm -f elf -l lab8-2.lst lab8-2.asm
lab8-2.asm:31: error: invalid combination of opcode and operands
[isbatova@fedora lab08]$
```

Рис. 2.14: Получение файла листинга

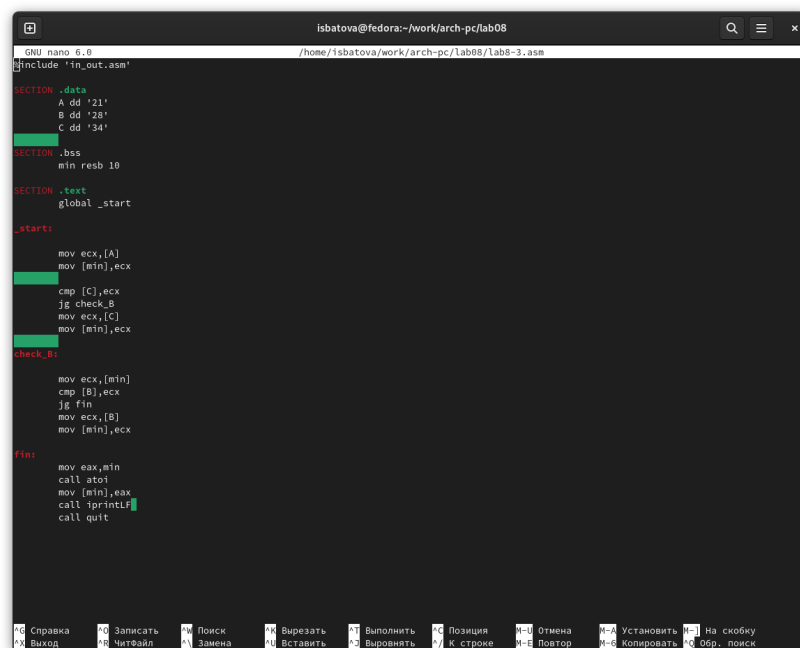
```
30 .....
31 ..... cmp ecx
31 ..... ***** error: invalid combination of opcode and operands
32 0000011C 7F0C ..... jg check_B
33 0000011E 8B0D[3A000000] ..... mov ecx,[C]
34 00000124 890D[00000000] ..... mov [max],ecx
35 .....
```

Рис. 2.15: Файл листинг - ошибка

### 3 Задание для самостоятельной работы

1. Мой номер варианта 11 - поэтому значения a, b и c для первого задания, согласно таблице, 21, 28 и 34. Значит, программа должна выводить число 21 (наименьшее).

Создаем файл 'lab8-3.asm' и пишем в нем программу для вывода наименьшего числа (рис. 3.1).



```
GNU nano 6.0 /home/isbatova/work/arch-pc/lab08/lab8-3.asm
#include "in_out.asm"

SECTION .data
A dd '21'
B dd '28'
C dd '34'

SECTION .bss
min resb 10

SECTION .text
global _start

_start:
    mov ecx, [A]
    mov [min], ecx

    cmp [C], ecx
    jg check_B
    mov ecx, [C]
    mov [min], ecx

check_B:
    mov ecx, [min]
    cmp [B], ecx
    jg fin
    mov ecx, [B]
    mov [min], ecx

fin:
    mov eax, min
    call atoi
    mov [min], eax
    call iprintf
    call quit
```

Рис. 3.1: Ввод программы в файл 'lab8-3.asm'

Создаем исполняемый файл и запускаем его (рис. 3.2). Программа выводит правильный результат, значит, она написана корректно.

```

[isbatova@fedora lab08]$ nasm -f elf lab8-3.asm
[isbatova@fedora lab08]$ ld -m elf_i386 lab8-3.o -o lab8-3
[isbatova@fedora lab08]$ ./lab8-3
21
[isbatova@fedora lab08]$

```

Рис. 3.2: Запуск программы из файла 'lab8-3.asm'

2. Для второго задания используем функцию

- $4a, x = 0$
- $4a + x, x \neq 0$

Создаем файл 'lab8-4.asm' и вводим в него программу (рис. 3.3), (рис. 3.4).

```

#include 'in_out.asm'

SECTION .data
    msg1 db 'Введите x: ',0h
    msg2 db 'Введите a: ',0h
    otvet: db 'Ответ: ',0
    B dd '0'

SECTION .bss
    x resb 10
    a resb 10

SECTION .text
    global _start

_start:
    mov eax, msg1
    call sprint

    mov ecx, x
    mov edx, 10
    call sread

    mov eax, msg2
    call sprint

    mov ecx, a
    mov edx, 10
    call sread

    mov eax, x
    call atoi
    mov esi, eax

```

Рис. 3.3: Ввод программы в файл 'lab8-4.asm', часть 1



```

        cmp eax,0
        jnz big_1
        mov eax,a
        call atoi
        mov ecx, 4
        mul ecx
        jmp fin

big_1:
        mov eax,a
        call atoi
        mov ecx, 4
        mul ecx
        add eax,esi

fin:
        mov esi,eax
        mov eax,otvet
        call sprint
        mov eax,esi
        call iprintLF

        call quit

```

Рис. 3.4: Ввод программы в файл 'lab8-4.asm', часть 2

Создаем исполняемый файл и запускаем его (рис. 3.5). Вводим пары чисел соответственно данным в таблице в лабораторной работы - 0,3 и 1,2. Если проверить аналитически, получаются такие же ответы, значит, программа написана корректно.

```

[isbatova@fedora lab08]$ nasm -f elf lab8-4.asm
[isbatova@fedora lab08]$ ld -m elf_i386 lab8-4.o -o lab8-4
[isbatova@fedora lab08]$ ./lab8-4
Введите x: 0
Введите a: 3
Ответ: 12
[isbatova@fedora lab08]$ ./lab8-4
Введите x: 1
Введите a: 2
Ответ: 9

```

Рис. 3.5: Запуск программы из файла 'lab8-4.asm'

## 4 Выводы

В данной лабораторной работе мной были изучены команды условного и безусловного переходов, а также приобретены навыки написания программ с использованием переходов. Помимо этого, я узнала о назначении и структуре файла листинга.