РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 7

Студент: Белакова Полина Вячеславовна

Ст.билет: 1032252589

Группа: НКАбд-01-25

МОСКВА

2025 г

Цель работы

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

Порядок выполнения лабораторной работы

Реализация переходов в NASM

1. Создаю каталог для программам лабораторной работы № 7, перехожу в него и создаю файл lab7-1.asm (рисунок 1).

```
pvbelakova2@localhost-live:~$ mkdir ~/work/arch-pc/lab07
pvbelakova2@localhost-live:~$ cd ~/work/arch-pc/lab07
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ touch lab7-1.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab07$
```

Рисунок 1 - создание папки lab07, перемещение в нее и создание файла lab7-1.asm.

2. Ввожу в файл lab7-1.asm текст программы из листинга 7.1 (рисунок 2).

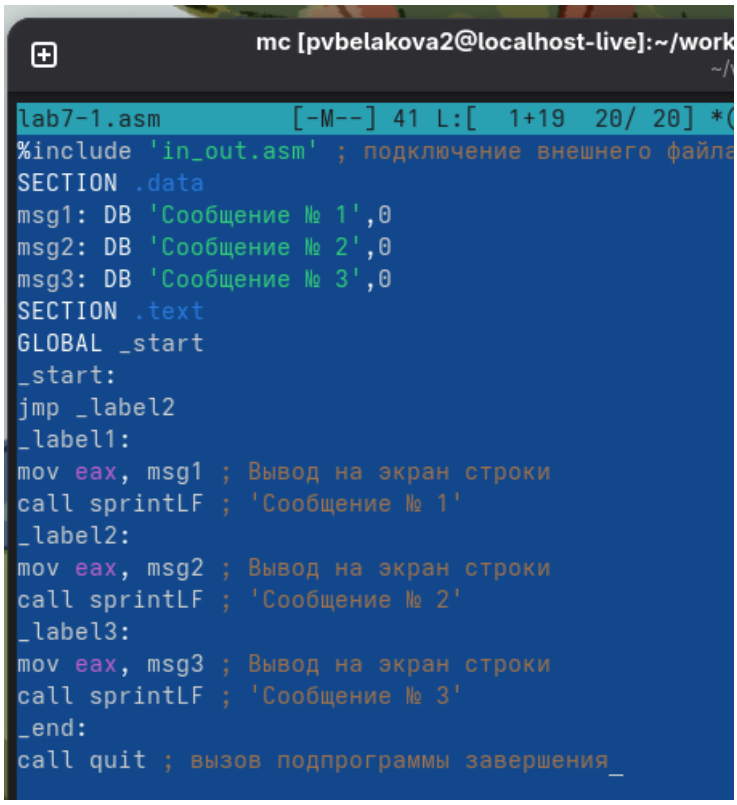

Рисунок 2 - код листинга 7.1

Создаю исполняемый файл и запускаю его (рисунок 3).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
pvbelakova2@localhost-live:~/work/arch-pc/lab07$
```

Рисунок 3 - создание исполняемого файла lab7-1 и его запуск.

Изменяю текст программы в соответствии с листингом 7.2 (рисунок 4).

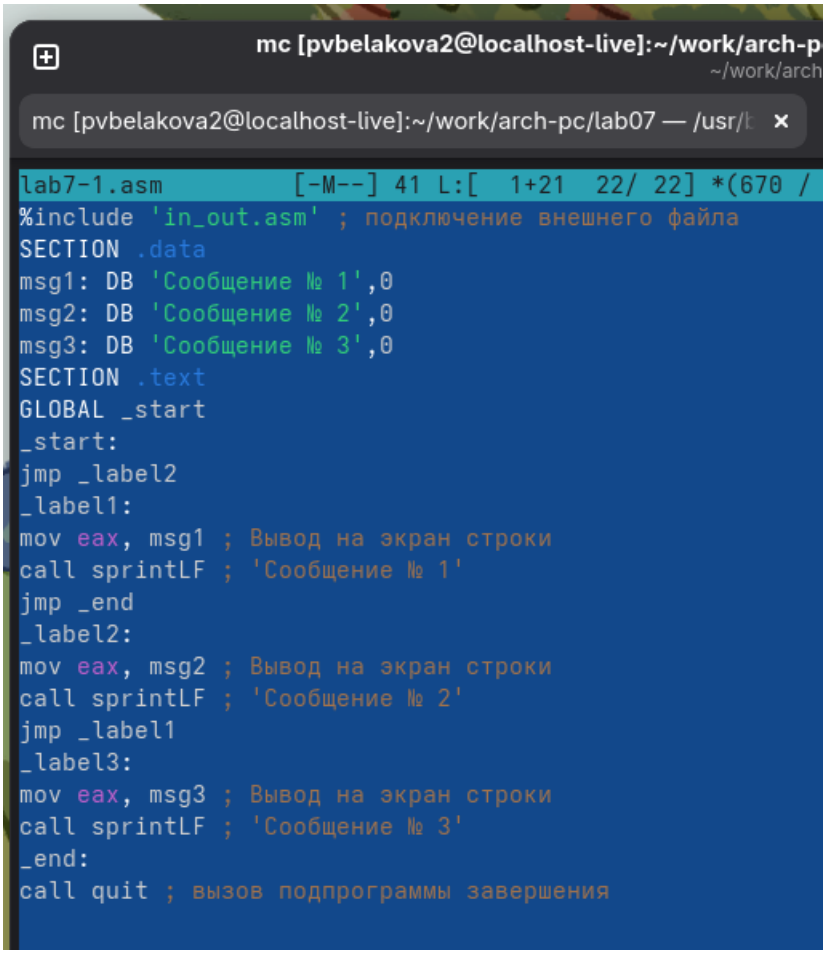

Рисунок 4 - код листинга 7.2.

Создаю исполняемый файл и проверяю его работу (рисунок 5).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm -o lab7-1_2.o
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1_2 lab7-1_2.o
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ./lab7-1_2
Сообщение № 2
Сообщение № 1
pvbelakova2@localhost-live:~/work/arch-pc/lab07$
```

Рисунок 5 - создание исполняемого файла lab7-1_2 и его запуск.

Изменяю текст программы добавив или изменив инструкции jmp (рисунок 6).

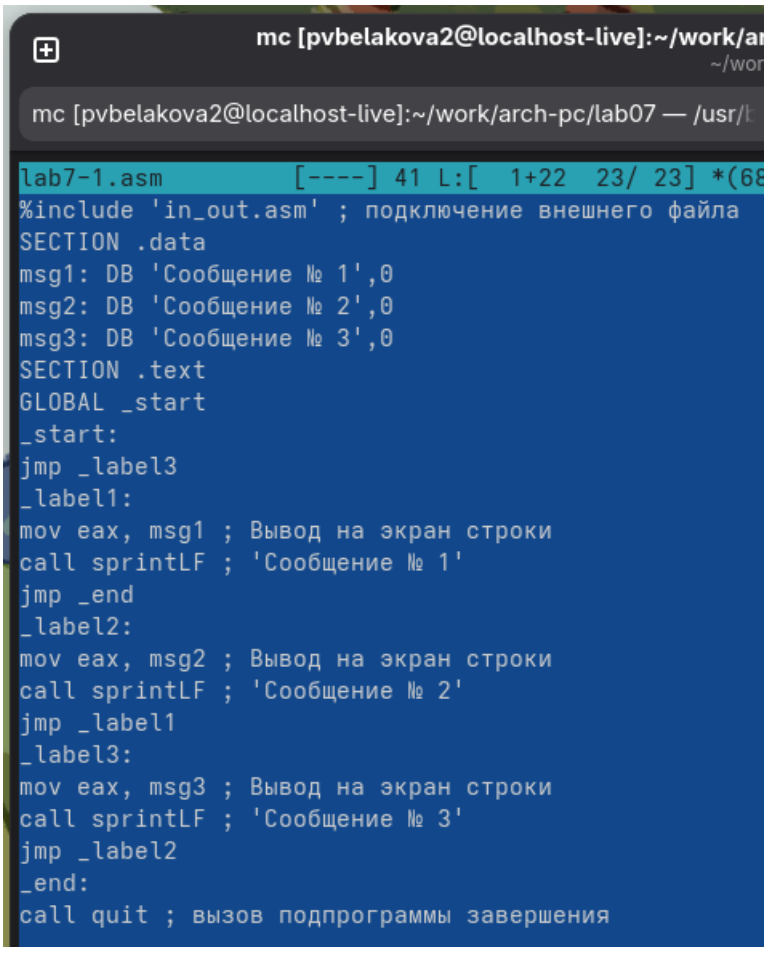

Рисунок 6 - изменненный код.

Создаю исполняемый файл и проверяю его работу (рисунок 7).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm -o lab7-1_3.o
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1_3 lab7-1_3.o
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ./lab7-1_3
Сообщение № 3
Сообщение № 2
Сообщение № 1
pvbelakova2@localhost-live:~/work/arch-pc/lab07$
```

Рисунок 7 - создание исполняемого файла lab7-1_3 и его запуск.

3. Создаю файл lab7-2.asm (рисунок 8).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ touch lab7-2.asm
```

Рисунок 8 - создание файла lab7-2.asm.

Ввожу в lab7-2.asm. код листинга 7.3. (рисунок 9).

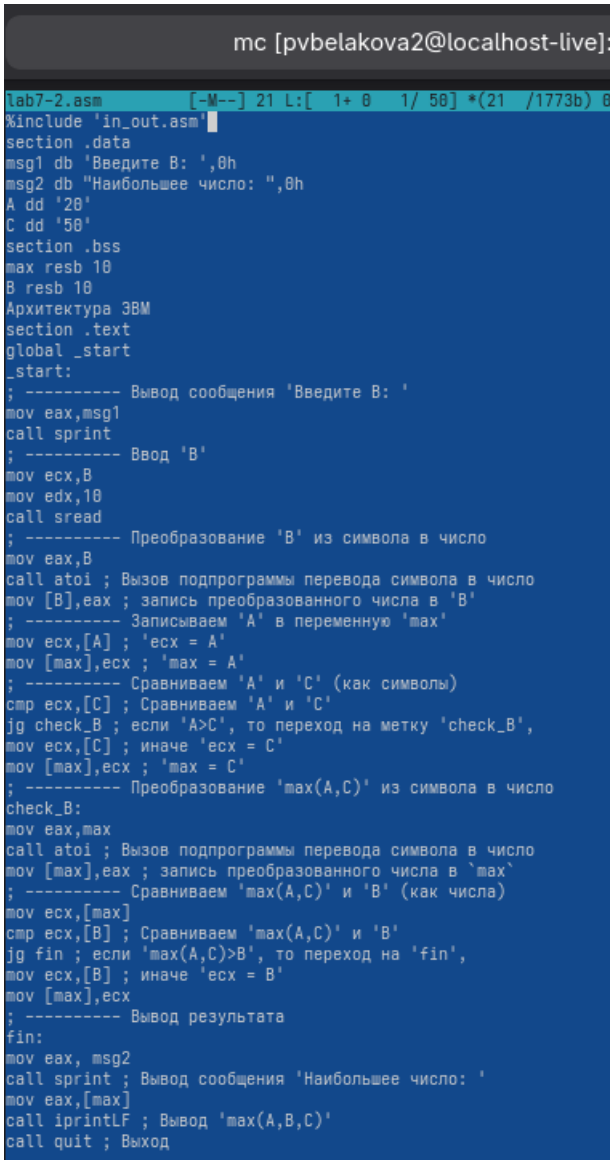

Рисунок 9 - код листинга 7.3.

Создаю исполняемый файл и проверяю его работу для разных значений B (рисунок 10).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-2 lab7-2.o
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 2
Наибольшее число: 50
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ./lab7-2
Введите B: 100
Наибольшее число: 100
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ./lab7-2
Введите B: -10
Наибольшее число: 50
pvbelakova2@localhost-live:~/work/arch-pc/lab07$
```

Рисунок 10 - создание исполняемого файла lab7-1_3 и его запуск с разными значениями B.

Изучение структуры файлы листинга

4. Создаю файл листинга для программы из файла lab7-2.asm (рисунок 11).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
```

Рисунок 11 - создание файла листинга.

Открываю файл листинга lab7-2.lst (рисунок 12).

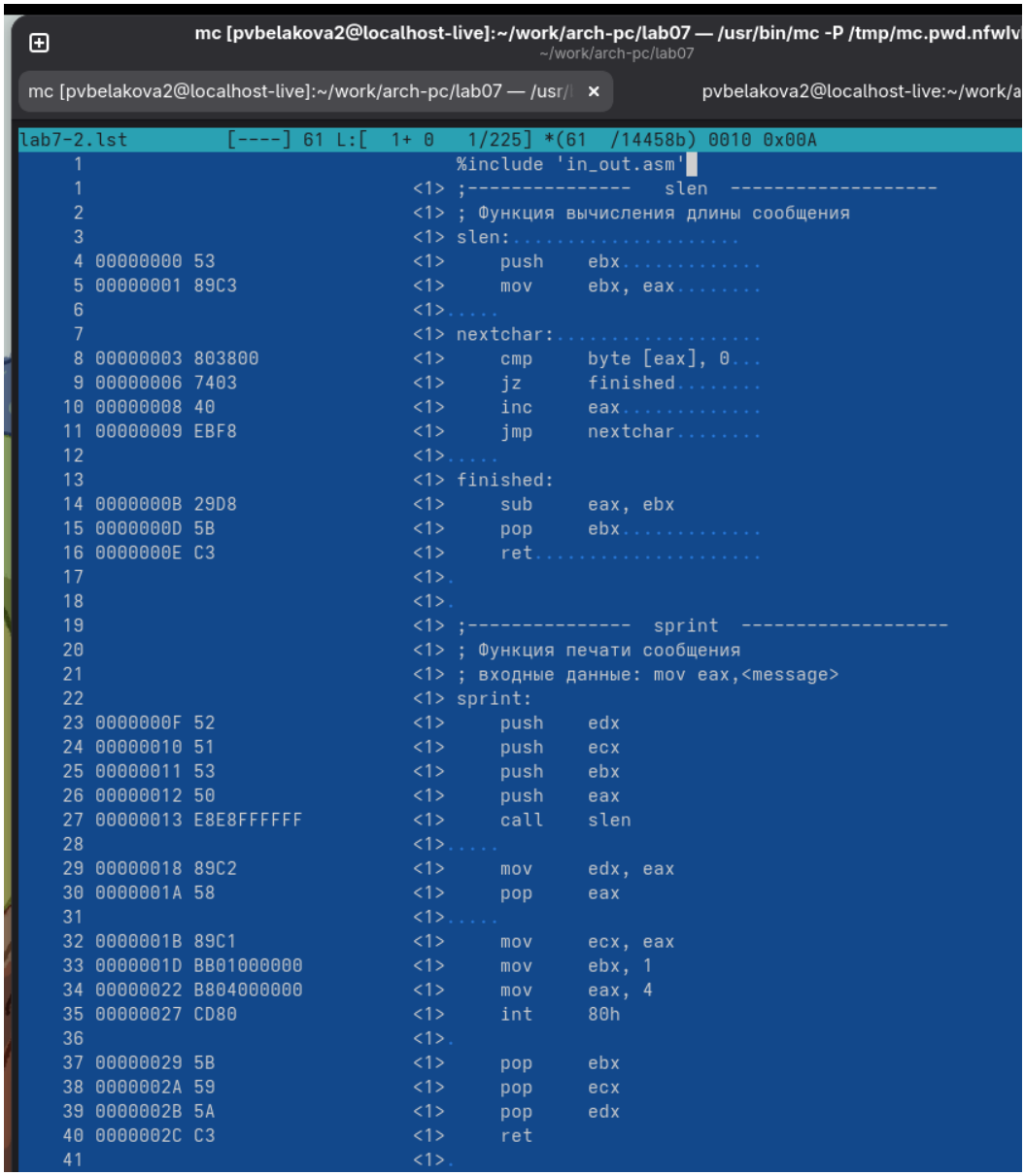

Рисунок 12 - часть кода файла листинга.

Строка 4 - первая инструкция в функции slen, сохраняет значение указанного регистра на вершине стека. Стек используется для временного хранения данных и сохранения состояния регистров.

Строка 10 - Эта инструкция находится внутри цикла nextchar в функции slen, inc увеличивает значение указанного операнда на единицу.

Строка 23 - первая инструкция в функции sprint, инструкция push сохраняет значение указанного регистра на вершине стека.

Открываю файл с программой lab7-2.asm и в инструкции mov eax, msg1 удаляю операнд msg1 (рисунок 13).

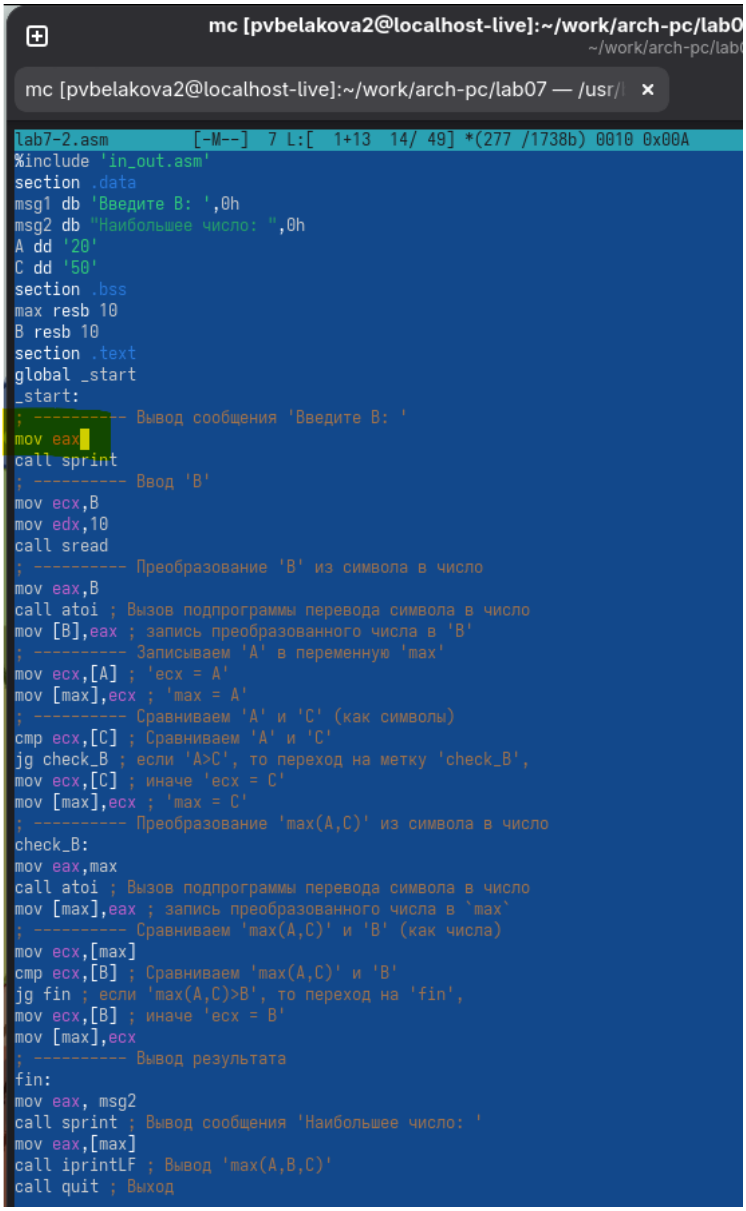

Рисунок 13 - код файла lab7-2.asm без операнда msg1.

Выполняю трансляцию с получением файла листинга (рисунок 14).

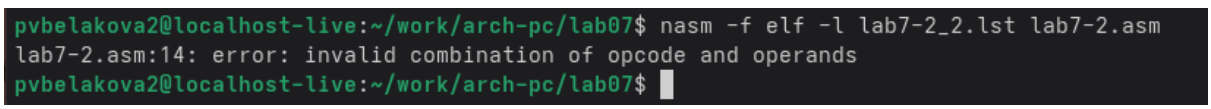

Рисунок 14 - получение файла листинга.

Какие выходные файлы создаются в этом случае? Что добавляется в листинге?

В этом случае, так как код содержит синтаксическую ошибку, создается только файл листинга lab7-2_2.lst. В листинг файле появляется указание ошибки (рисунок 15).

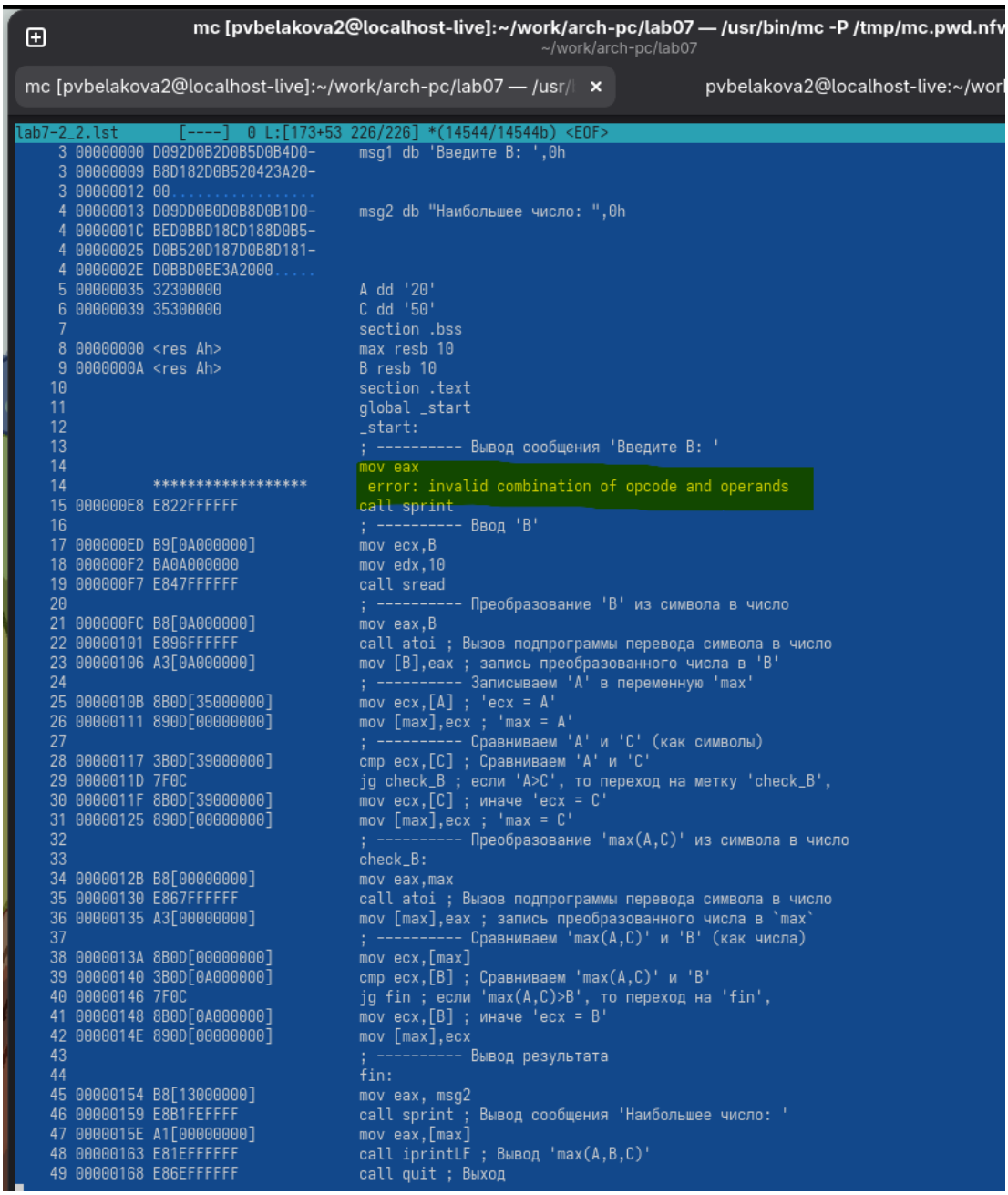

Рисунок 15 - часть листинг файла с указание ошибки.

Задание для самостоятельной работы (вариант 10)

1. Программа для нахождения наименьшей из 3 целочисленных переменных *a*, *b* и c (рисунок 16).

mc [pvbelakova2@localhost-live]:~/work/arch-pc/lab07 — /usr/

lab7-task1.asm [-M--] 48 L:[1+25 26/ 36] *(698 / 973b) 0010 0x0

```
%include 'in_out.asm'
section .data
msg db "Наименьшее число: ",0h
A dd 41
B dd 62
C dd 35
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 'B'
cmp ecx,[B] ; Сравниваем 'A' и 'B'
jl compare_AC ; если 'A<B', то переход на метку 'compare_AC',
mov ecx,[B] ; иначе 'ecx = B'
mov [min],ecx ; 'min = B'

; --- Сравниваем 'min(A,B)' и 'C'
compare_AC:
mov ecx,[min]
cmp ecx,[C] ; Сравниваем 'min(A,B)' и 'C'
jl fin ; если 'min(A,B)<C', то переход на 'fin',
mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'min = C'

; --- Вывод результата
fin:
mov eax, msg
call sprint ; Вывод сообщения 'Наименьшее число: '
mov eax,[min]
call iprintLF ; Вывод 'min(A,B,C)'
call quit ; Выход
```

Рисунок 16 - код программы для решения 10 варианта.

Создайте исполняемый файл и проверьте его работу (рисунок 17).

```
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ nasm -f elf lab7-task1.asm
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-task1 lab7-task1.o
pvbelakova2@localhost-live:~/work/arch-pc/lab07$ ./lab7-task1
Наименьшее число: 35
pvbelakova2@localhost-live:~/work/arch-pc/lab07$
```

Рисунок 17 - создание исполняемого файла lab7-task и его запуск.

2. Программа, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений (рисунок 18).

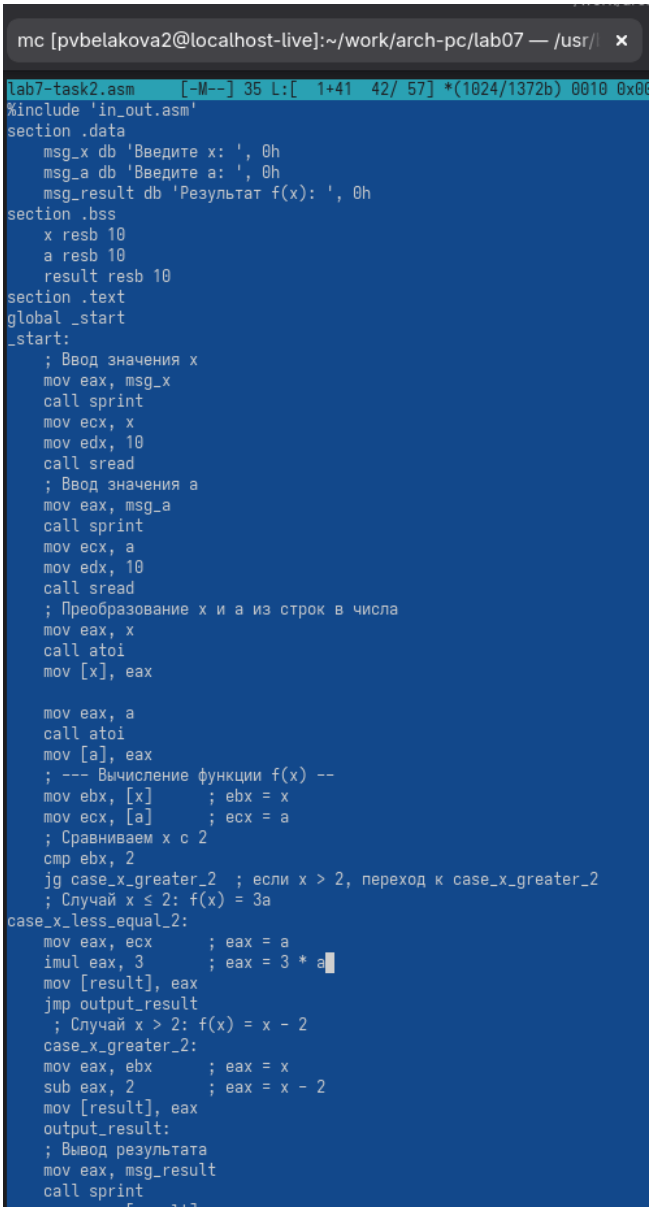

Рисунок 18 - часть кода программы.

Создаю исполняемый файл и проверяю его работу (рисунок 19).

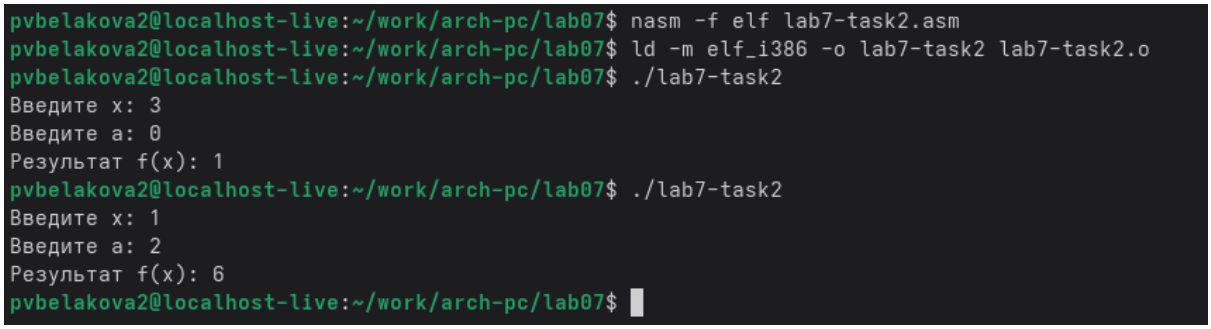

Рисунок 19 - создание исполняемого файла lab7-task2 и его запуск со значениями из таблицы.

Выводы

В ходе лабораторной работы были изучены команды условного и безусловного переходов. Приобретены навыки написания программ с использованием переходов. Изучено с назначение и структура файла листинга.