Отчет по лабораторной работе №7

Дисциплина: архитектура компьютера

Курушин Георгий Романович

Содержание

# 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

Здесь приводится описание задания в соответствии с рекомендациями методического пособия и выданным вариантом.

# 3 Теоретическое введение

Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов: • условный переход – выполнение или не выполнение перехода в определенную точку программы в зависимости от проверки условия. • безусловный переход – выполнение передачи управления в определенную точку про- граммы без каких-либо условий.

# 4 Выполнение лабораторной работы

## 4.1 Реализация переходов в NASM

Создаю каталог для программ лабораторной работы №7 (рис. -fig. 1).

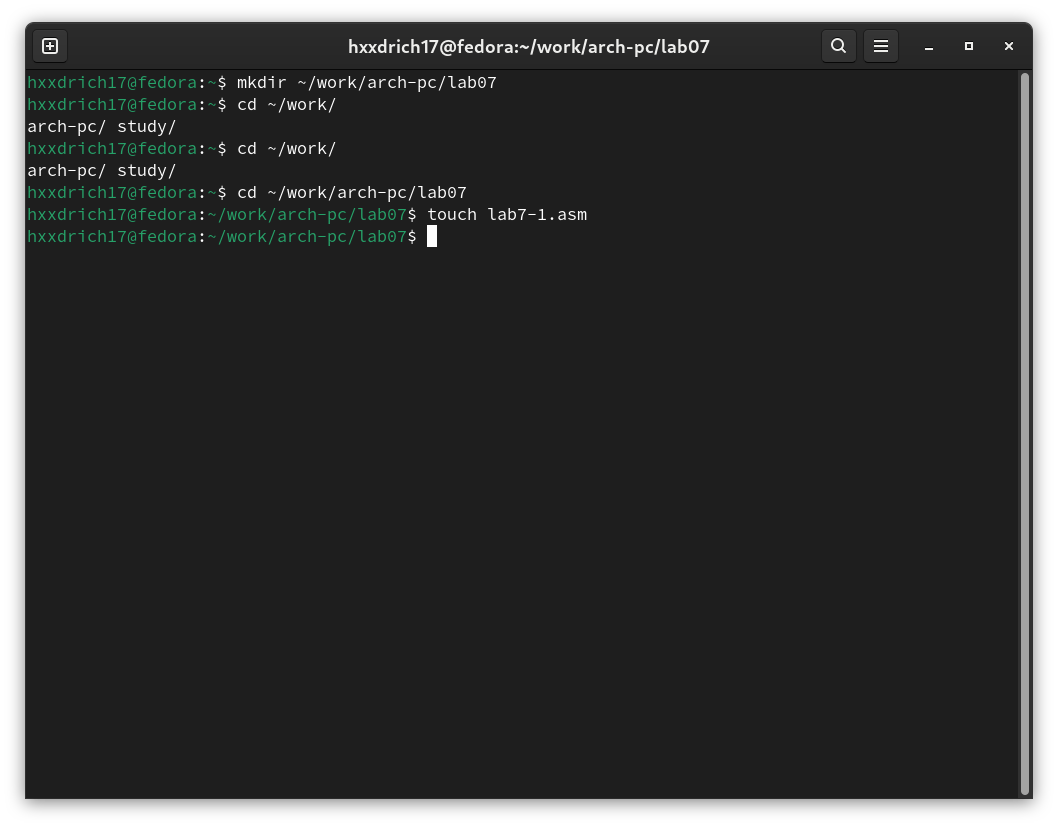


Рис. 1: Создание каталога и файла для программы

Копирую код из листинга в файл будущей программы. (рис. -fig. 2).

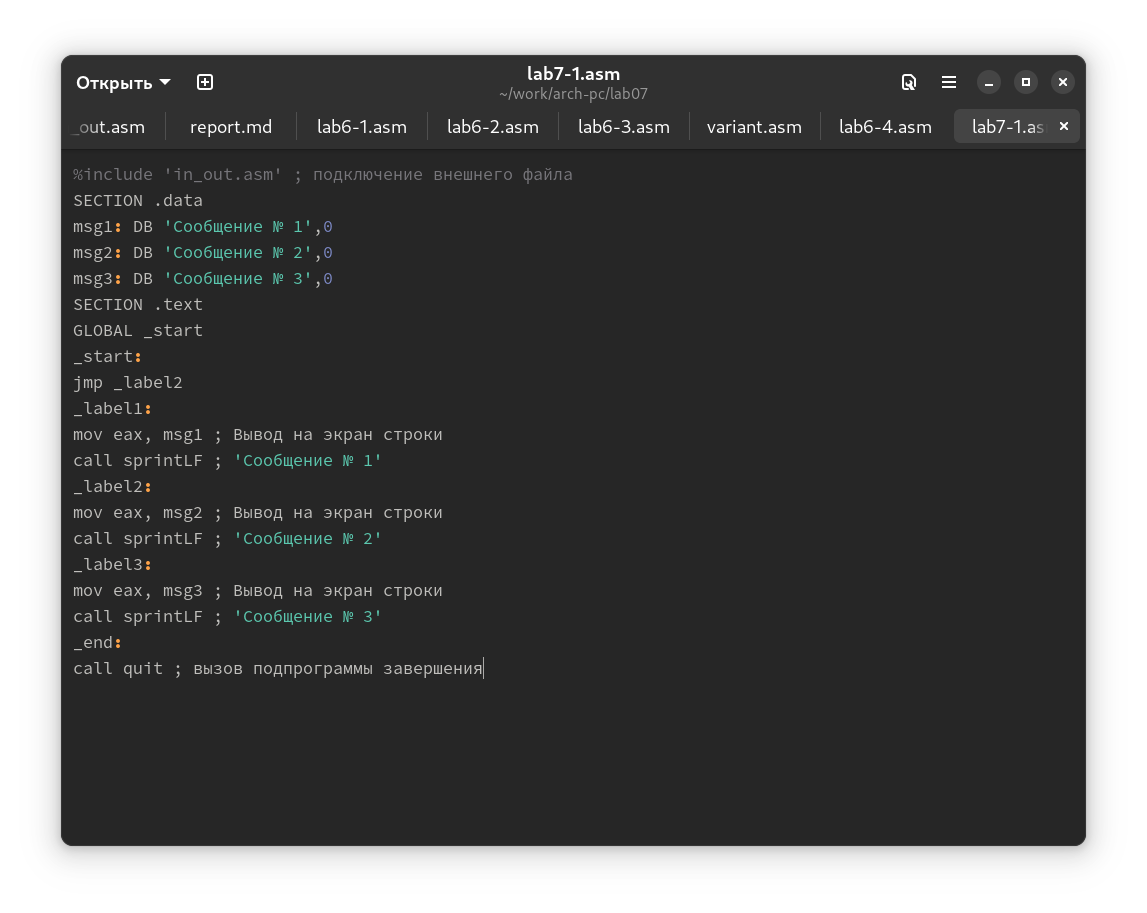


Рис. 2: Сохранение программы

При запуске программы я убедился в том, что неусловный переход действительно изменяет порядок выполнения инструкций (рис. -fig. 3).

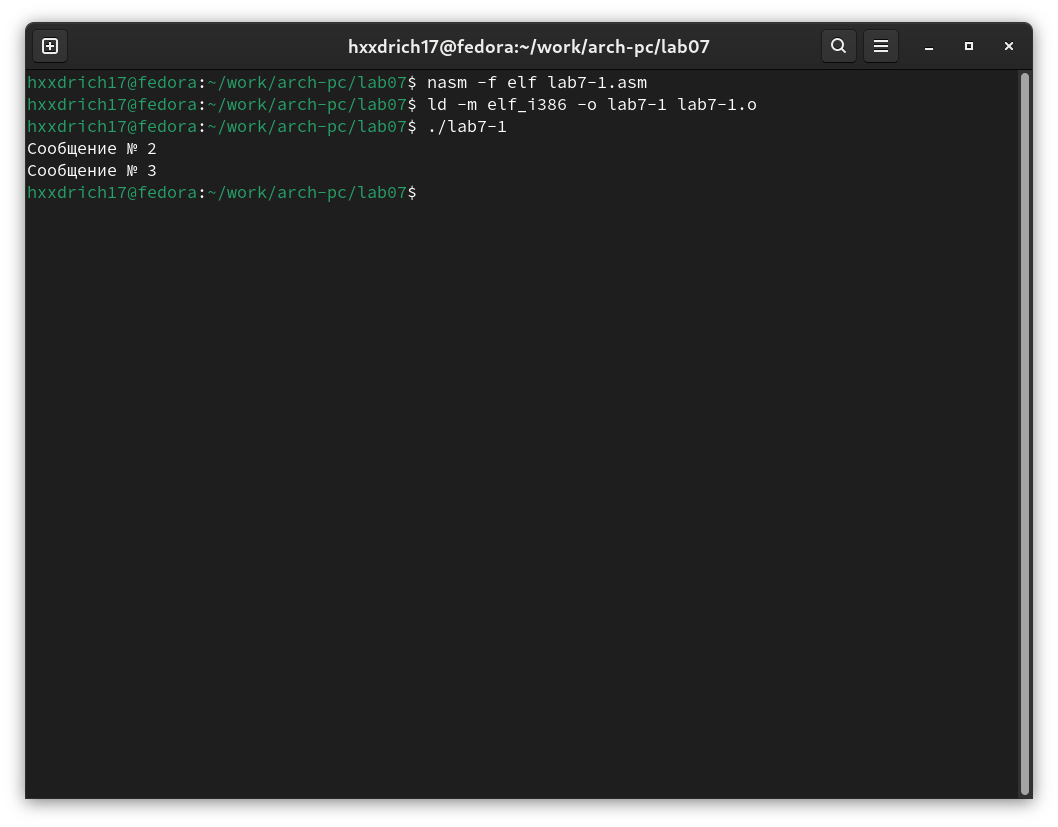


Рис. 3: Запуск программы

Изменяю программу таким образом, чтобы поменялся порядок выполнения функций (рис. -fig. 4).

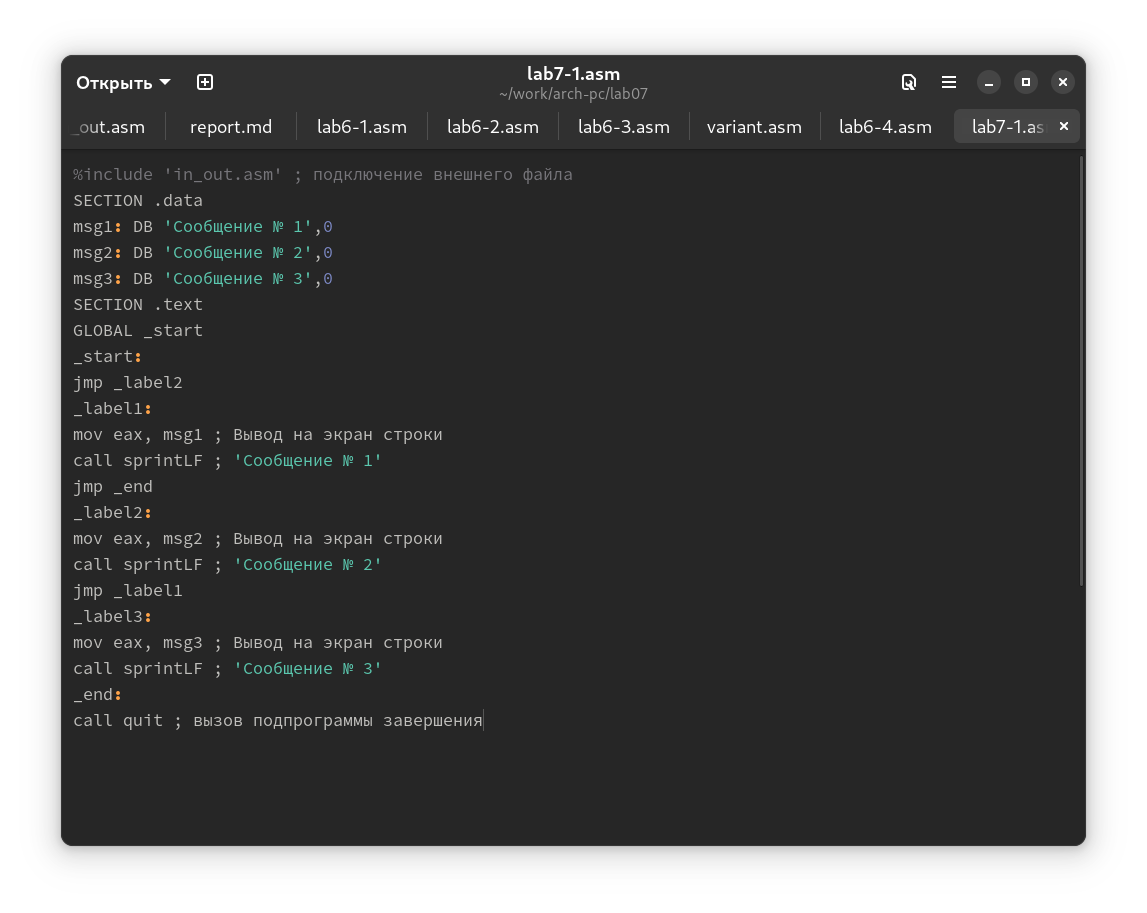


Рис. 4: Изменение программы

Запускаю программу и проверяю, что примененные изменения верны (рис. -fig. 5).

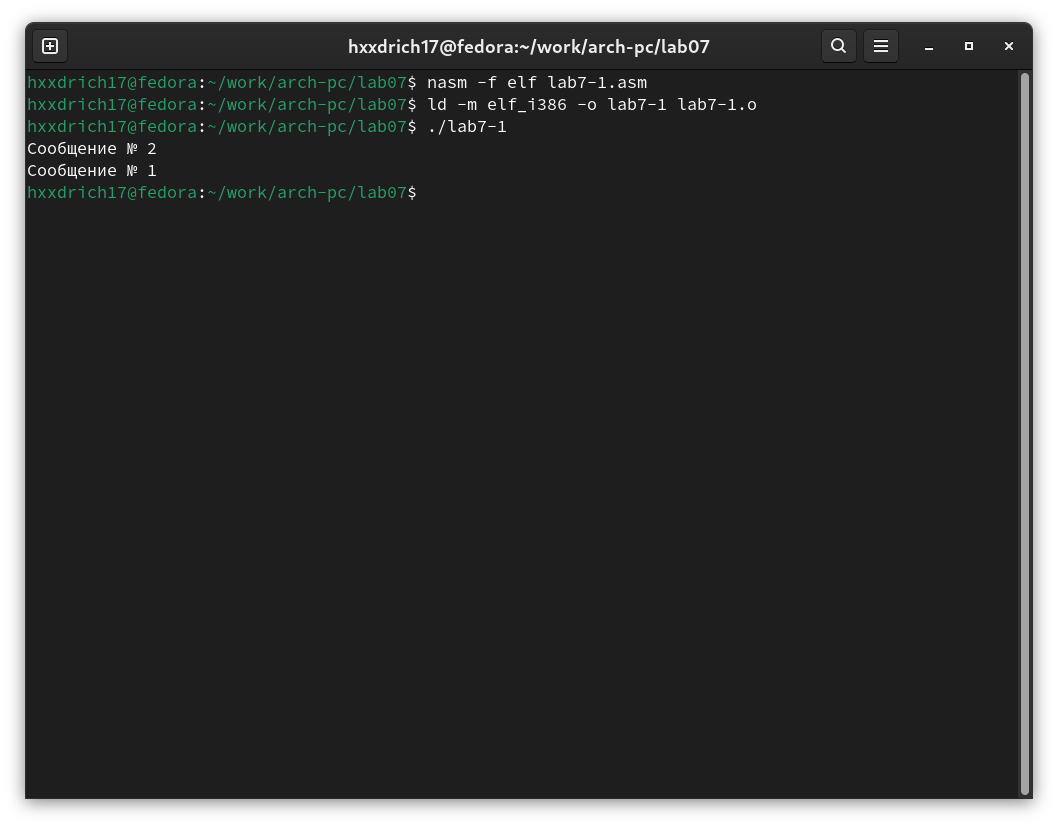


Рис. 5: Запуск изменеенной программы

Теперь изменяю текст программы так, чтобы все три сообщения вывелись в обратном порядке (рис. -fig. 6).

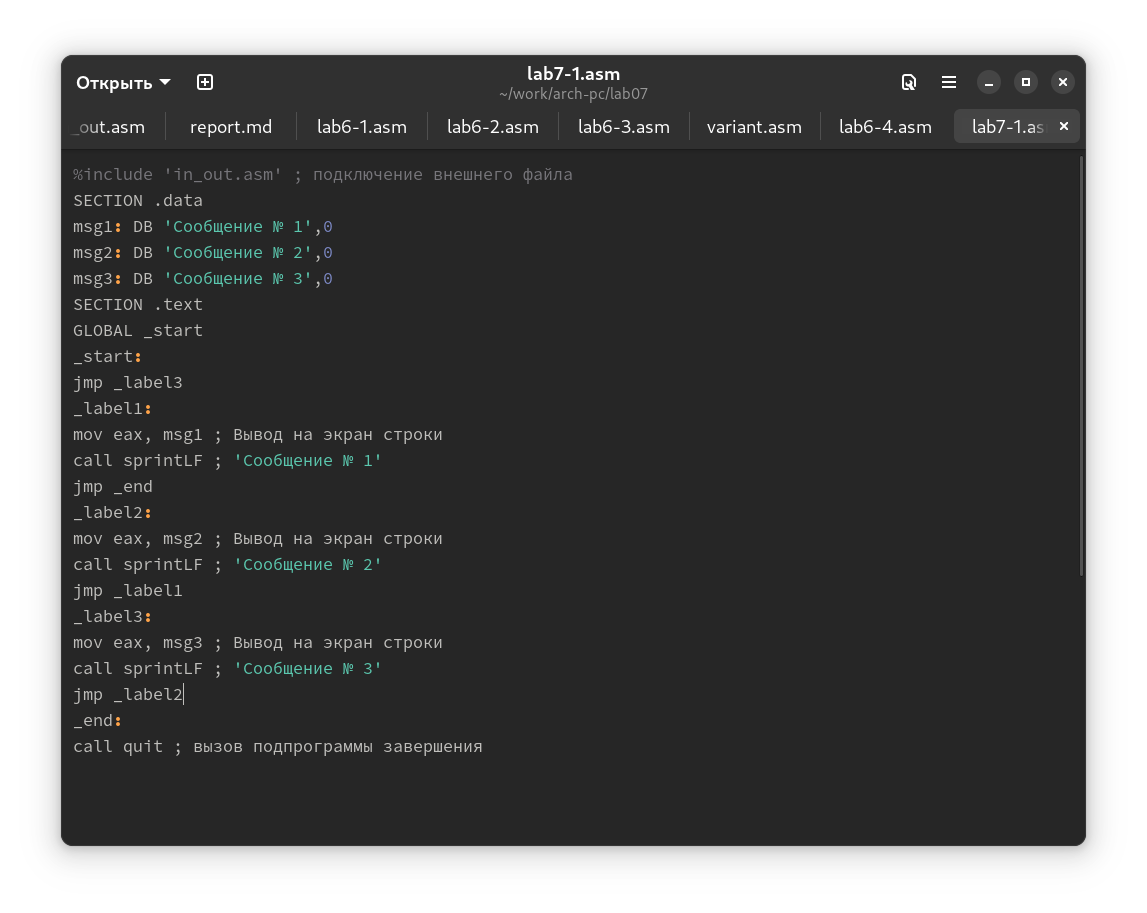


Рис. 6: Изменение программы

Работа выполнена корректно, программа в нужном мне порядке выводит сообщения (рис. -fig. 7).

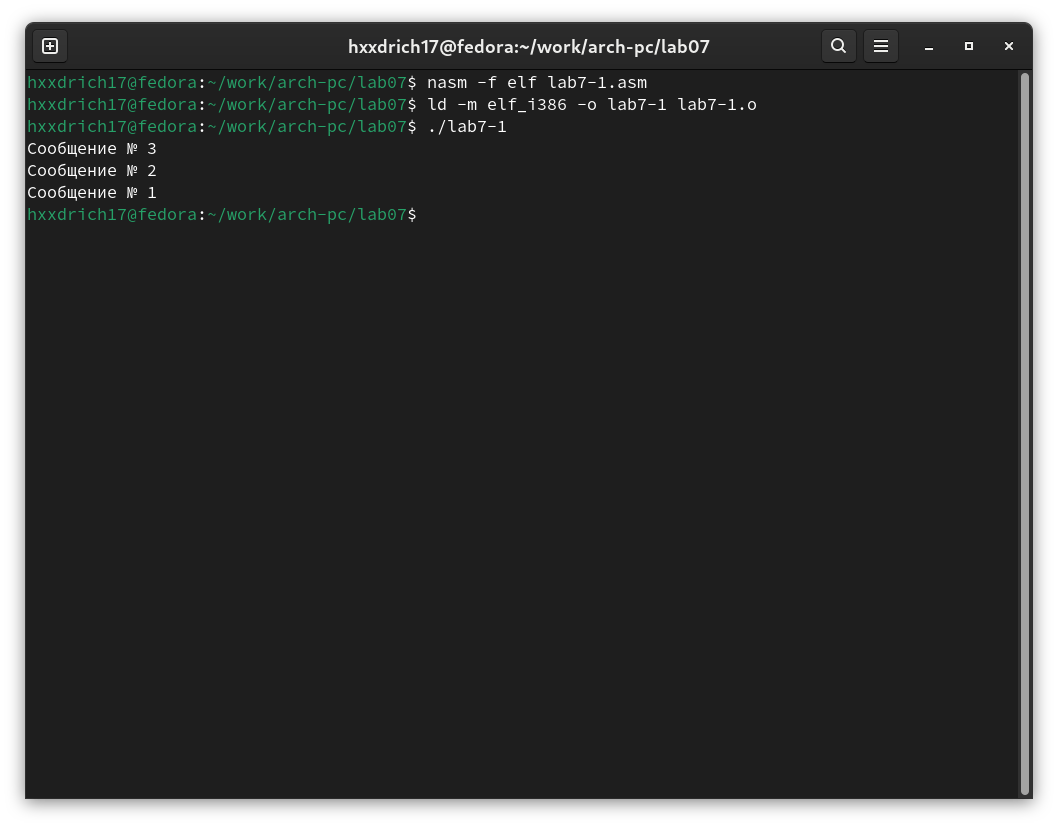


Рис. 7: Проверка изменений

Создаю новый рабочий файл и вставляю в него код из следующего листинга (рис. -fig. 8).

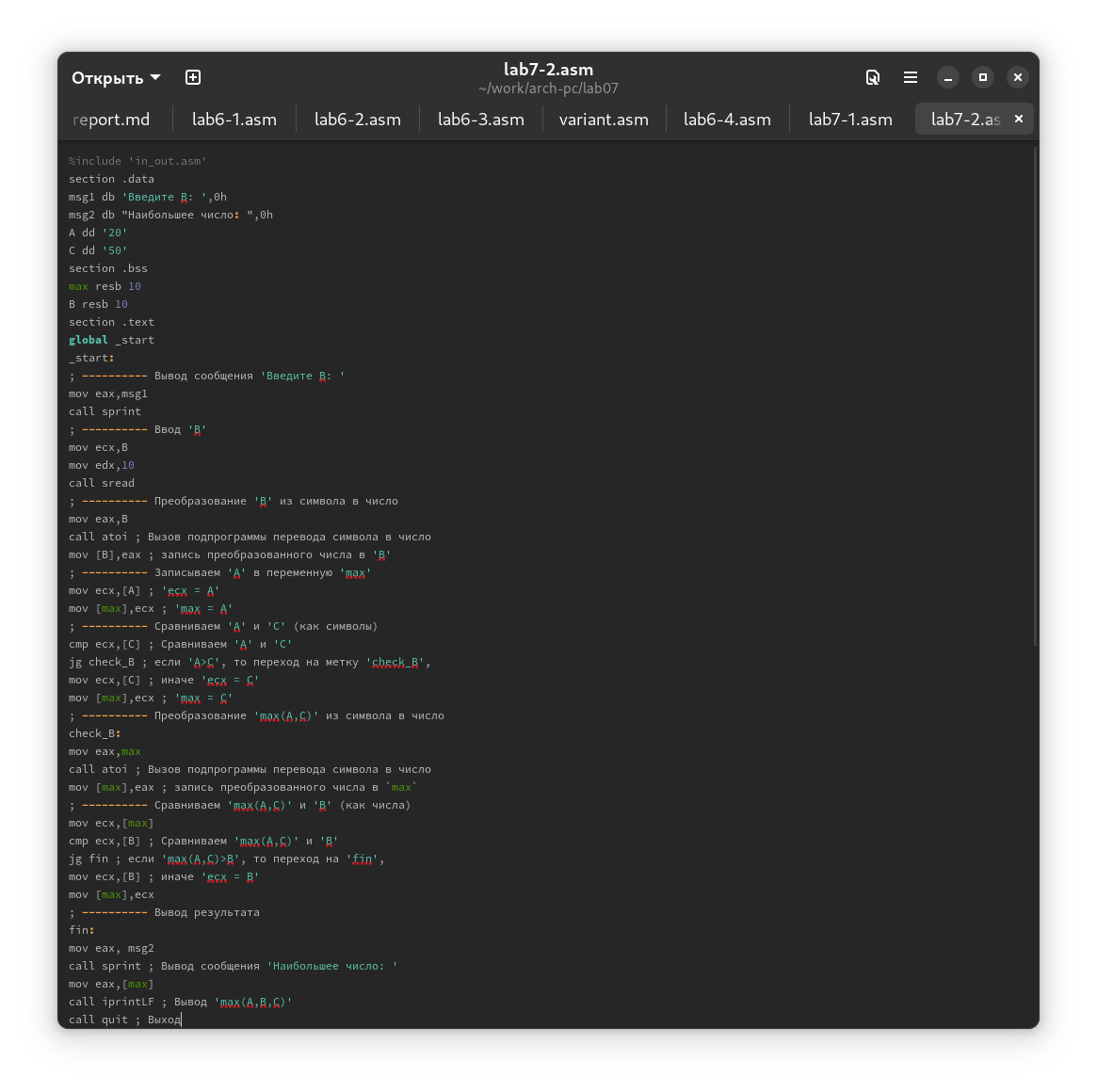


Рис. 8: Сохранение новой программы

Программа выводит значение переменной с максимальным значением, проверяю работу программы с разными входными данными (рис. -fig. 9).

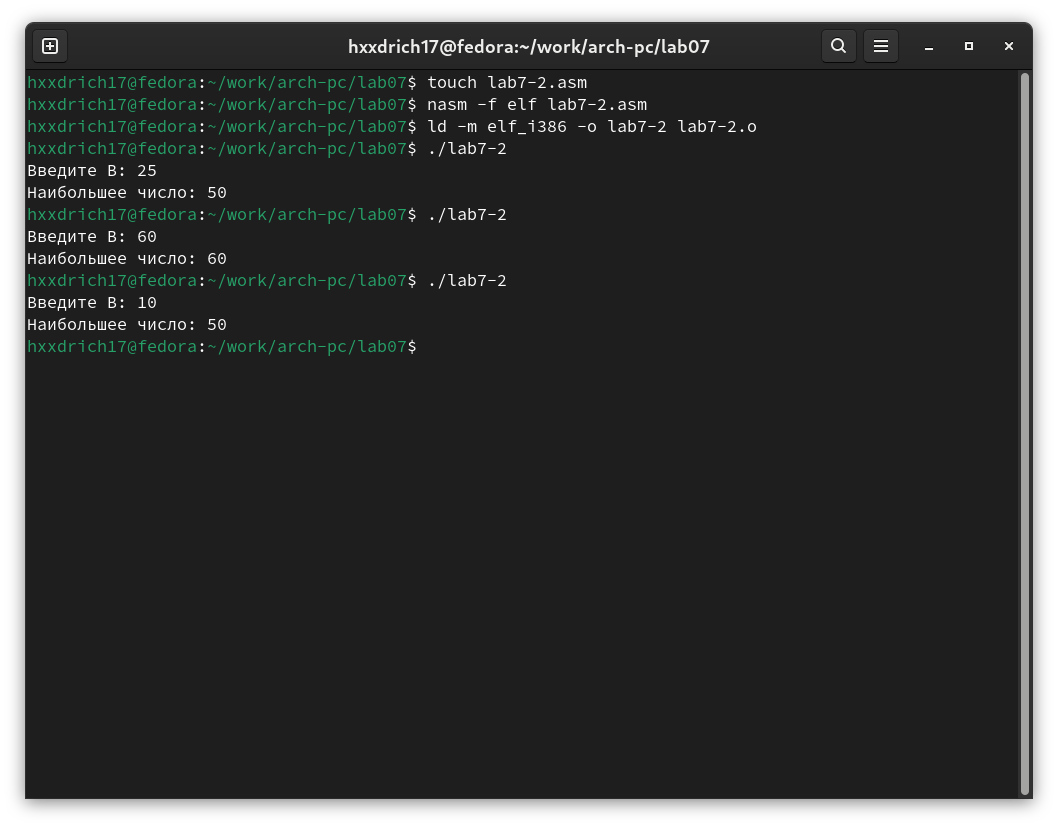


Рис. 9: Проверка программы из листинга

## 4.2 Изучение структуры файла листинга

Создаю файл листинга с помощью флага -l команды nasm и открываю его с помощью текстового редактора mcedit (рис. -fig. 10).

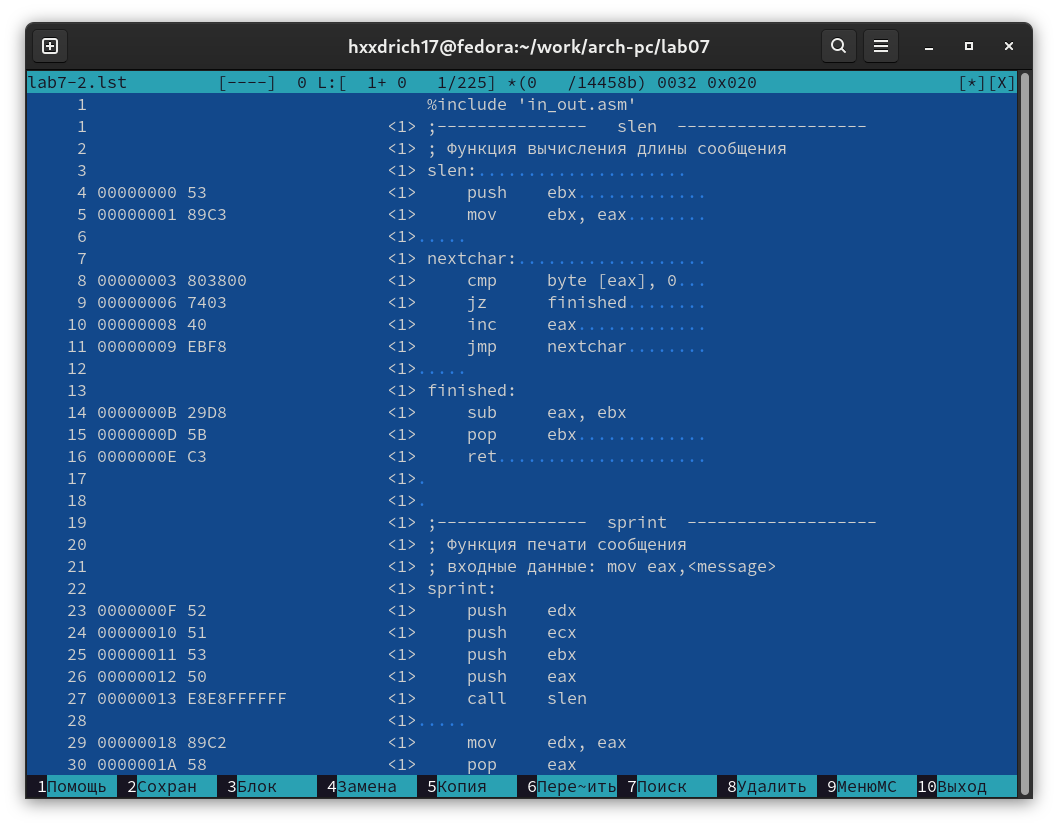


Рис. 10: Проверка файла листинга

Первое значение в файле листинга - номер строки, и он может вовсе не совпадать с номером строки изначального файла. Второе вхождение - адрес, смещение машинного кода относительно начала текущего сегмента, затем непосредственно идет сам машинный код, а заключает строку исходный текст прогарммы с комментариями.

Удаляю один операнд из случайной инструкции, чтобы проверить поведение файла листинга в дальнейшем (рис. -fig. 11).

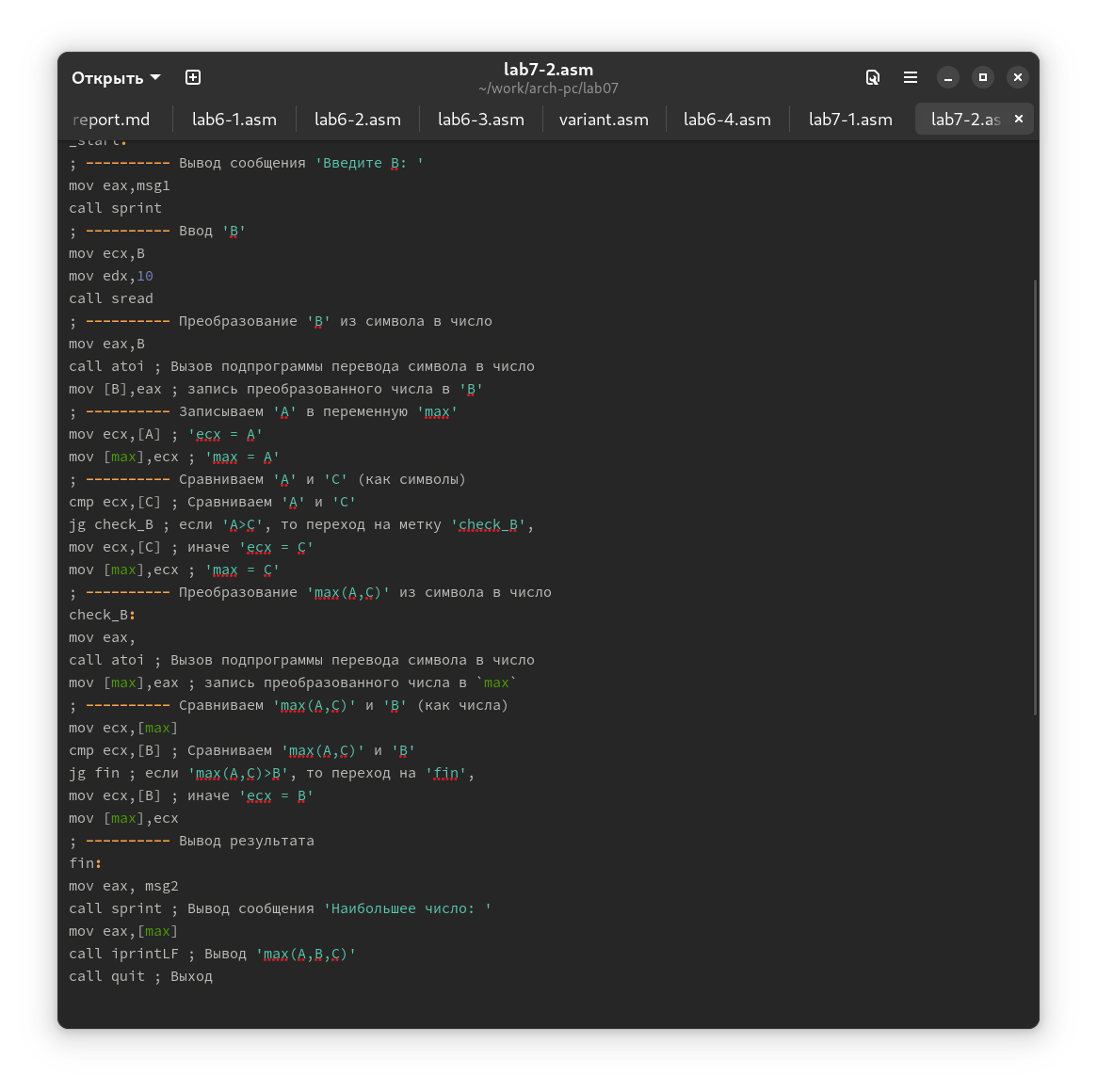


Рис. 11: Удаление операнда из программы

В новом файле листинга показывает ошибку, которая возникла при попытке трансляции файла. Никакие выходные файлы при этом помимо файла листинга не создаются. (рис. -fig. 12).

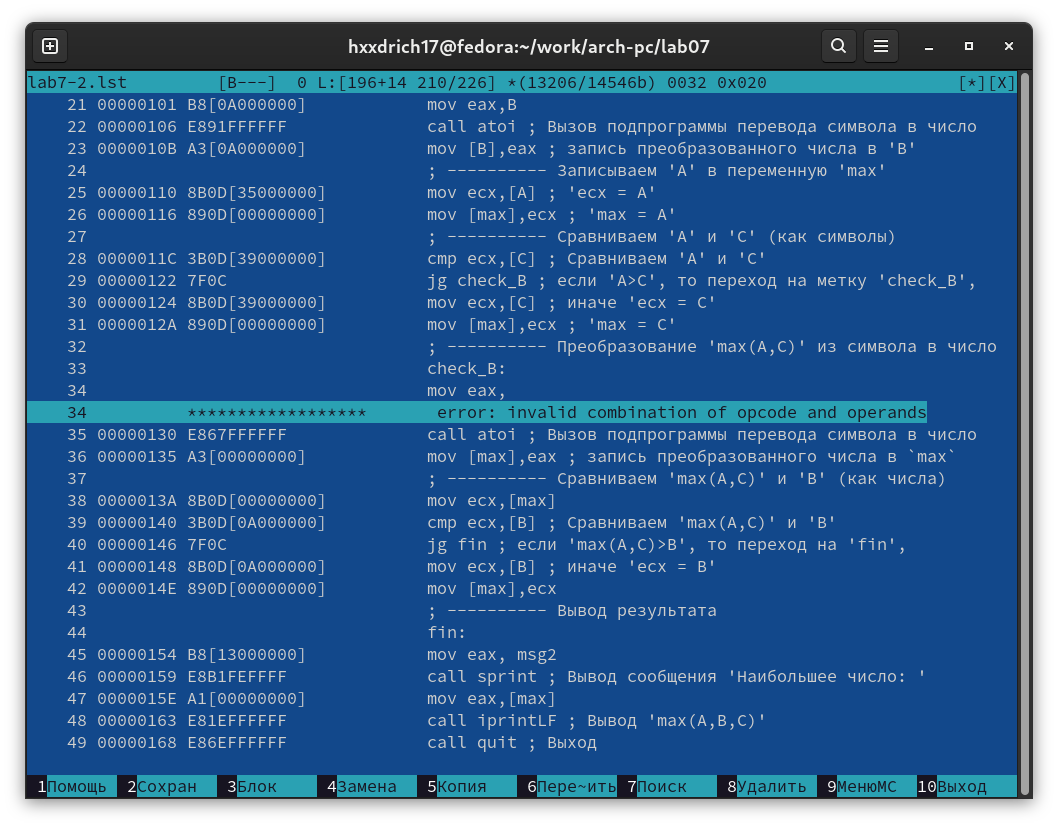


Рис. 12: Просмотр ошибки в файле листинга

## 4.3 Задания для самостоятельной работы

Буду использовать свой вариант - шестнадцатый - из предыдудщей лабораторной работы. Возвращаю операнд к функции в программе и изменяю ее так, чтобы она выводила переменную с наименьшим значением (рис. -fig. 13).

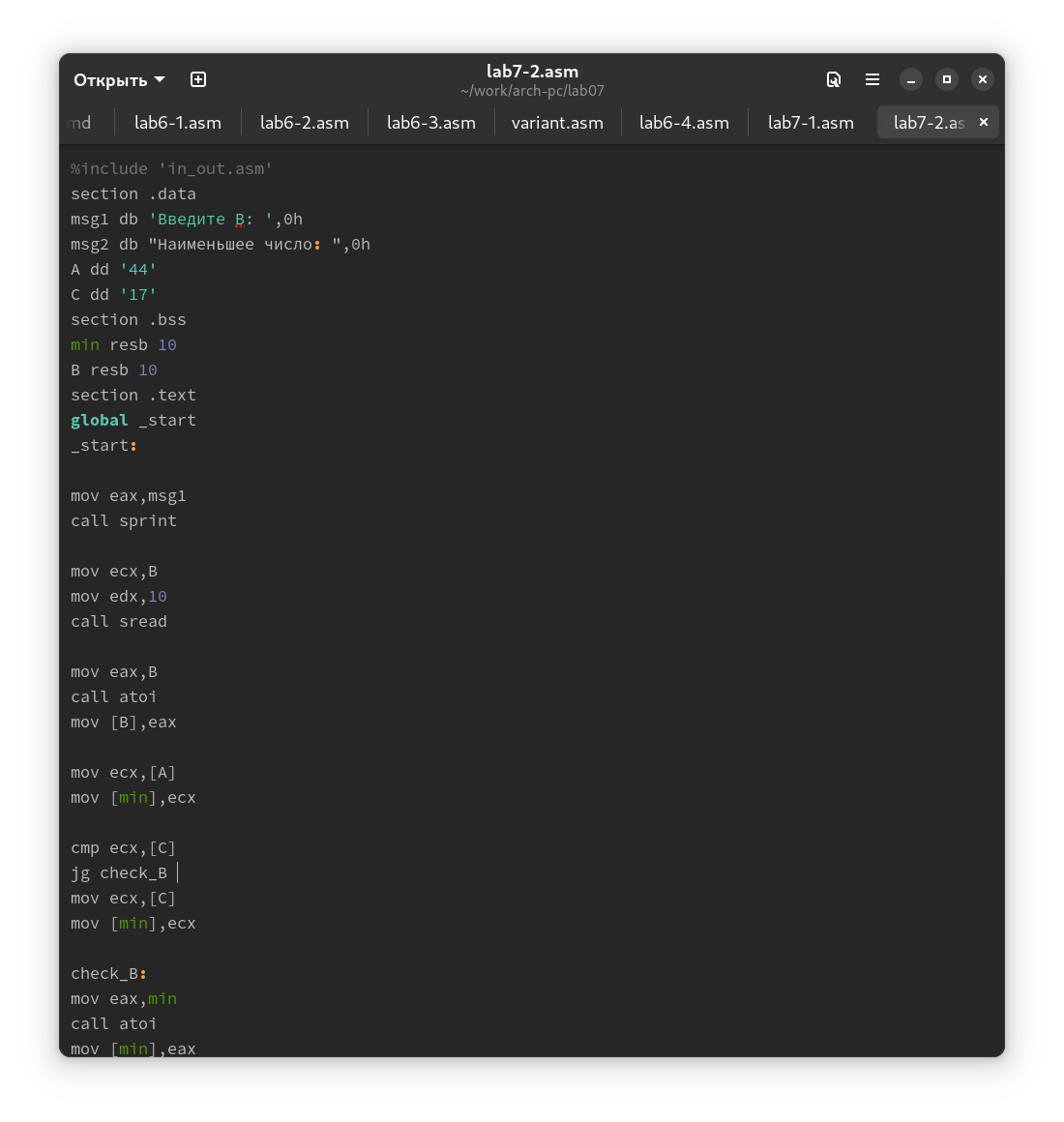


Рис. 13: Первая программа самостоятельной работы

Код первой программы:

%include 'in\_out.asm'  
section .data  
msg1 db 'Введите B: ',0h  
msg2 db "Наименьшее число: ",0h  
A dd '44'  
C dd '17'  
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]   
jg check\_B   
mov ecx,[C]   
mov [min],ecx   
  
check\_B:  
mov eax,min  
call atoi   
mov [min],eax  
  
mov ecx,[min]  
cmp ecx,[B]   
jb fin  
mov ecx,[B]  
mov [min],ecx  
  
fin:  
mov eax, msg2  
call sprint  
mov eax,[min]  
call iprintLF  
call quit

Проверяю корректность написания первой программы (рис. -fig. 14).

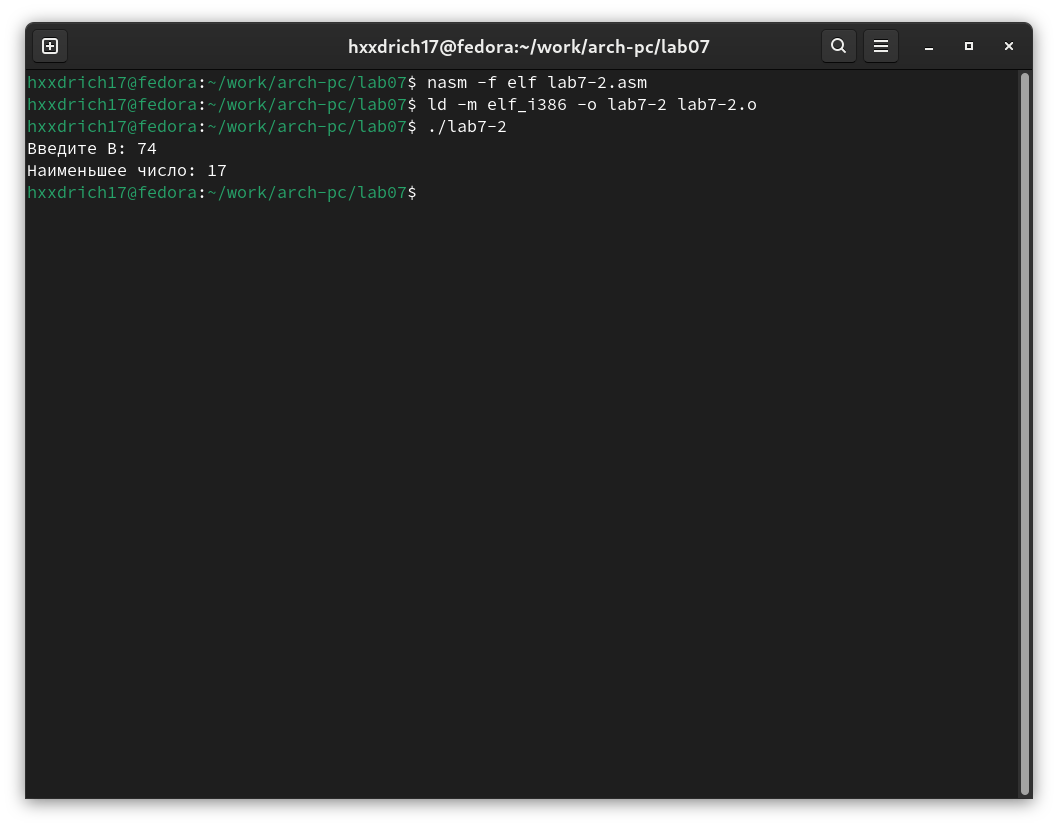


Рис. 14: Проверка работы первой программы

Пишу программу, которая будет вычислять значение заданной функции согласно моему варианту для введенных с клавиатурых переменных a и x (рис. -fig. 15).

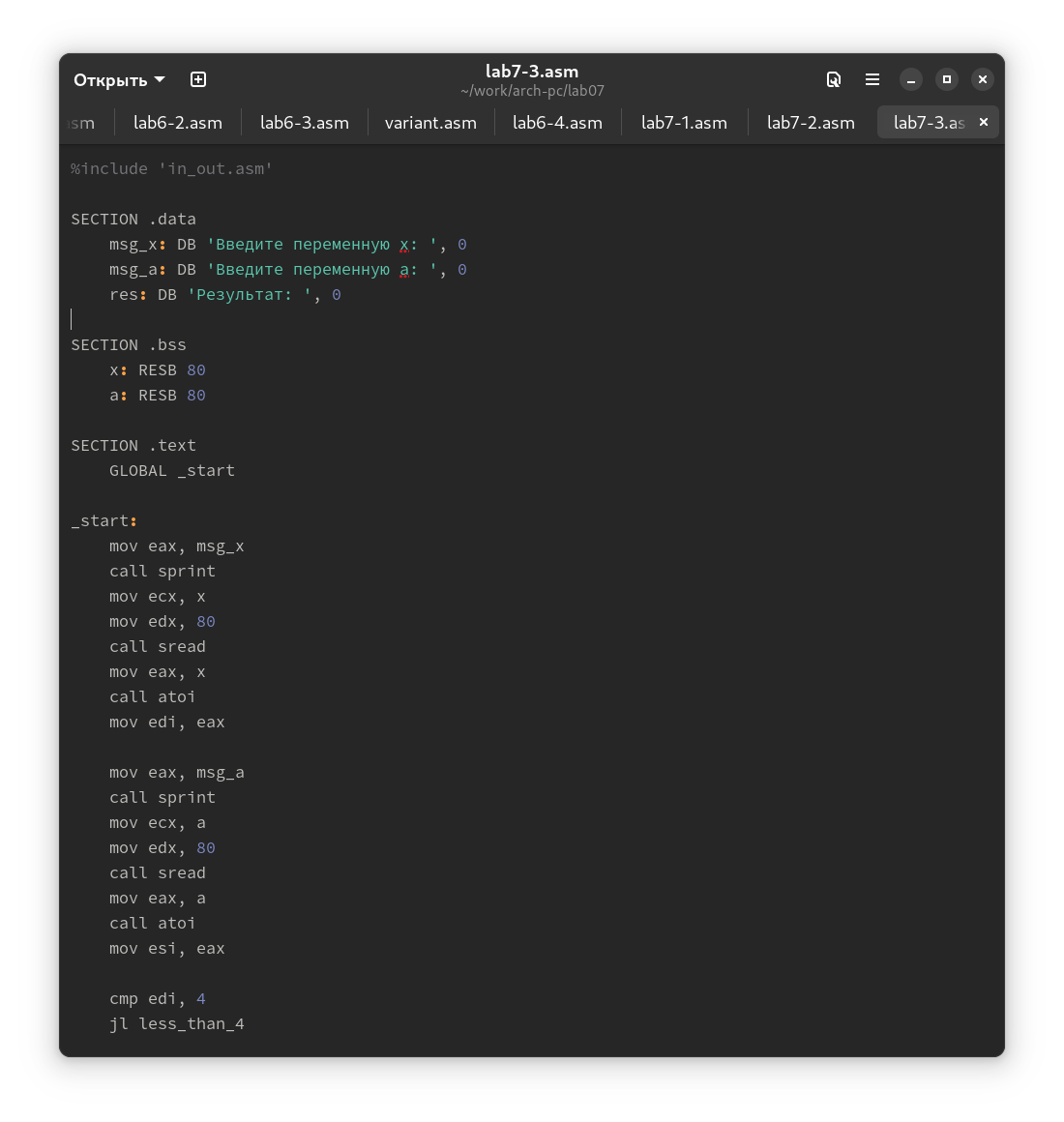


Рис. 15: Вторая программа самостоятельной работы

Код второй программы:

%include 'in\_out.asm'  
  
SECTION .data  
 msg\_x: DB 'Введите переменную x: ', 0  
 msg\_a: DB 'Введите переменную a: ', 0  
 res: DB 'Результат: ', 0  
  
SECTION .bss  
 x: RESB 80  
 a: RESB 80  
  
SECTION .text  
 GLOBAL \_start  
  
\_start:  
 mov eax, msg\_x  
 call sprint  
 mov ecx, x  
 mov edx, 80  
 call sread  
 mov eax, x  
 call atoi  
 mov edi, eax  
  
 mov eax, msg\_a  
 call sprint  
 mov ecx, a  
 mov edx, 80  
 call sread  
 mov eax, a  
 call atoi  
 mov esi, eax   
  
 cmp edi, 4  
 jl less\_than\_4   
  
 ; Ветка x >= 4  
 mov eax, edi  
 imul eax, esi  
 jmp print\_result  
  
less\_than\_4:  
 mov eax, edi  
 add eax, 4  
  
print\_result:  
 mov edi, eax  
 mov eax, res  
 call sprint  
 mov eax, edi  
 call iprintLF  
 call quit

Транслирую и компоную файл, запускаю и проверяю работу программмы для различных значений a и x (рис. -fig. 16).

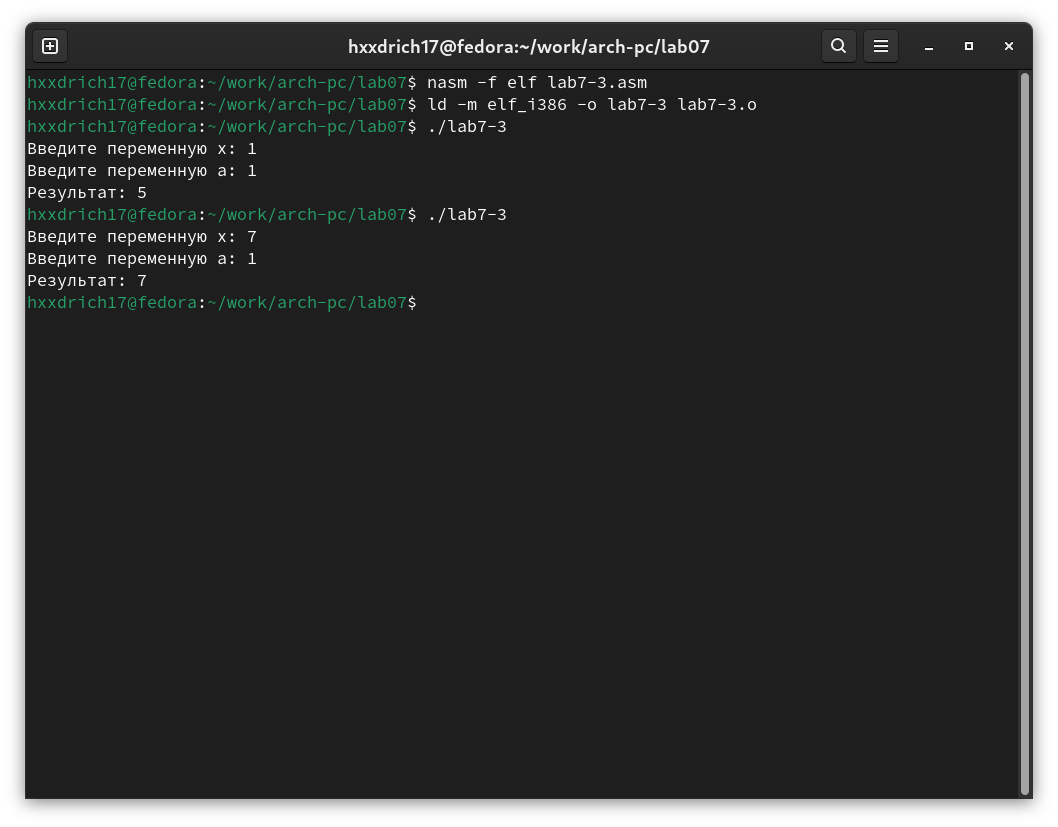


Рис. 16: Проверка работы второй программы

# 5 Выводы

При выполнении лабораторной работы я изучил команды условных и безусловных переходов, а также приобрел навыки написания программ с использованием переходов, познакомился с назначением и структурой файлов листинга.

# Список литературы

1. [Курс на ТУИС](https://esystem.rudn.ru/course/view.php?id=112)
2. [Лабораторная работа №7](https://esystem.rudn.ru/pluginfile.php/2089087/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%967.%20%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D1%8B%20%D0%B1%D0%B5%D0%B7%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B8%20%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%BF%D0%B5%D1%80%D0%B5%D1%85%D0%BE%D0%B4%D0%BE%D0%B2%20%D0%B2%20Nasm.%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%82%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9.pdf)
3. [Программирование на языке ассемблера NASM Столяров А. В.](https://esystem.rudn.ru/pluginfile.php/2088953/mod_resource/content/2/%D0%A1%D1%82%D0%BE%D0%BB%D1%8F%D1%80%D0%BE%D0%B2%20%D0%90.%20%D0%92.%20-%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%BD%D0%B0%20%D1%8F%D0%B7%D1%8B%D0%BA%D0%B5%20%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0%20NASM%20%D0%B4%D0%BB%D1%8F%20%D0%9E%D0%A1%20Unix.pdf)