Отчёт по лабораторной работе №7

Дисциплина: архитектура компьютера

Учаева Алёна Сергеевна

Содержание

Список иллюстраций

# 1 Цель работы

Изучить команды условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Ознакомиться с назначением и структурой файла листинга.

# 2 Задание

1. Реализация переходов в NASM
2. Изучение структуры файлы листинга
3. Задания для самостоятельной работы

# 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. 1).

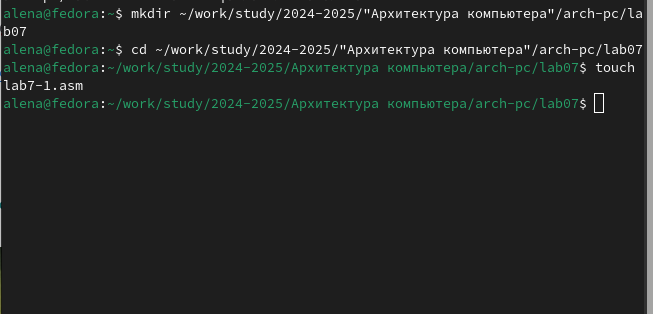


Рис. 1: Создание каталога

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

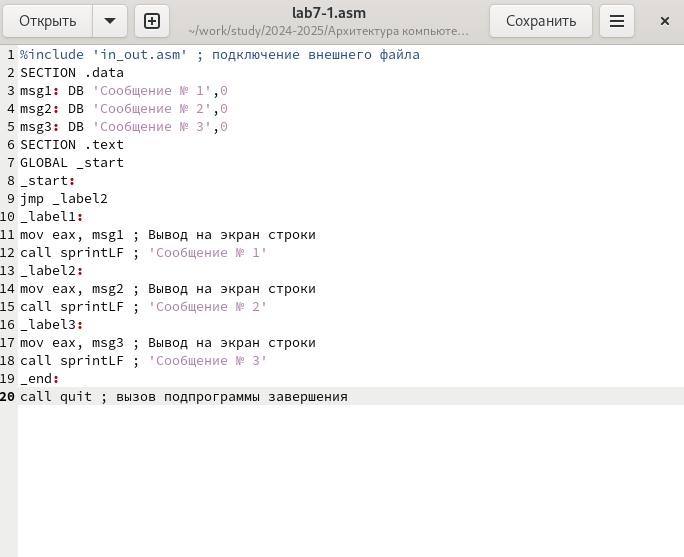


Рис. 2: Редактирование файла

Создаю исполняемый файл и запускаю его (рис. 3).

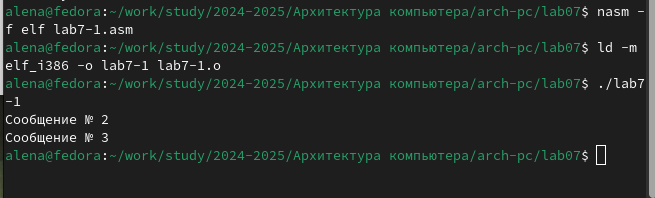


Рис. 3: Запуск исполняемого файла

Редактирую программу, чтобы поменялся порядок выполнения функций (рис. 4).

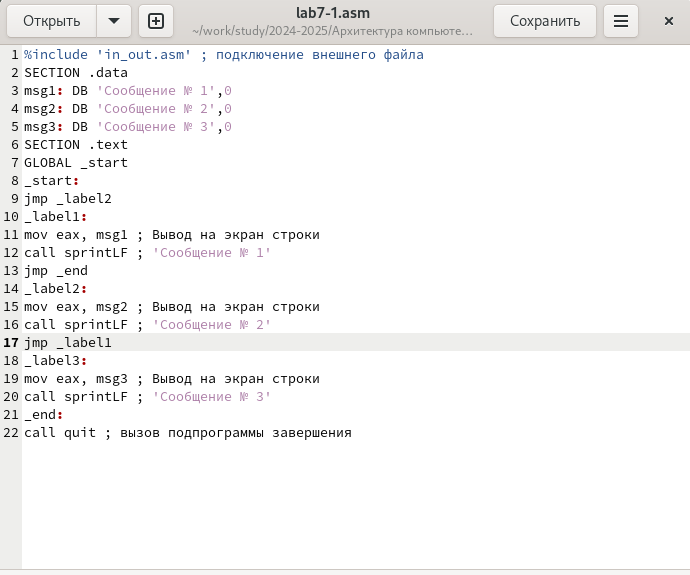


Рис. 4: Редактирование файла

Создаю исполняемый файл и запускаю его (рис. 5).

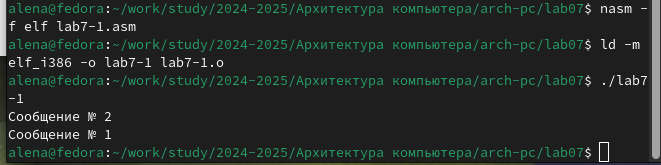


Рис. 5: Запуск исполняемого файла

Редактирую текст программы, чтобы сообщения вывелись в обратном порядке (рис. 6).

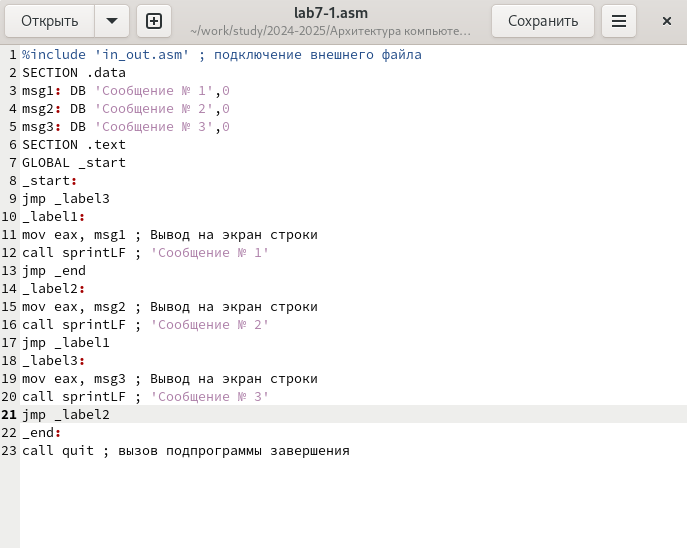


Рис. 6: Редактирование файла

Создаю исполняемый файл и запускаю его (рис. 7).

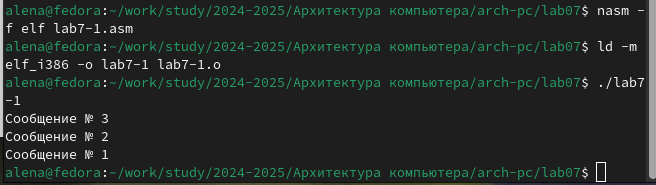


Рис. 7: Запуск исполняемого файла

Создаю файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 и ввожу в него текст программы из листинга 7.3 (рис. 8).

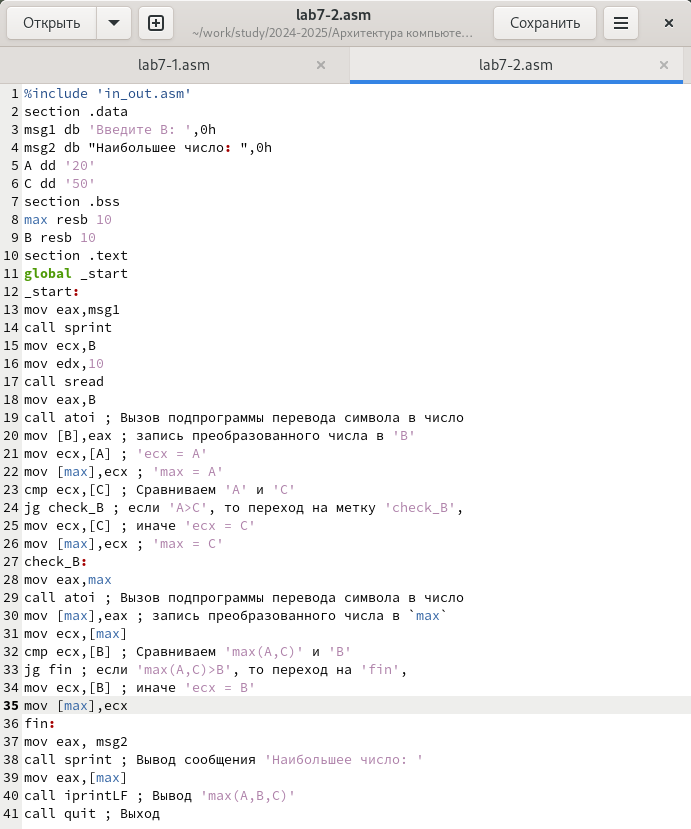


Рис. 8: Редактирование файла

Создаю исполняемый файл и запускаю его. Программа выводит значение переменной с максимальным значением, проверяю работу программы с разными входными данными (рис. 9).

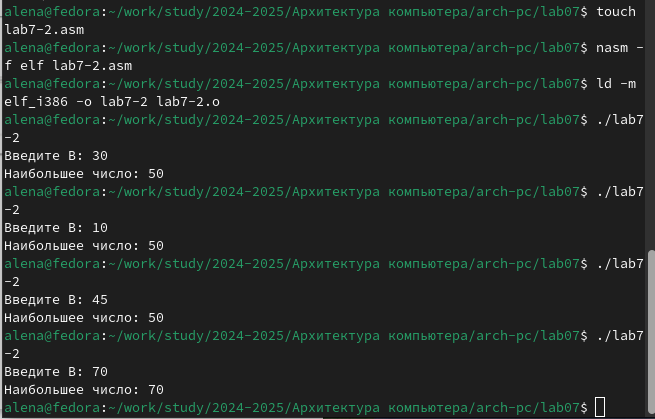


Рис. 9: Запуск исполняемого файла

## 4.2 Изучение структуры файлы листинга

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

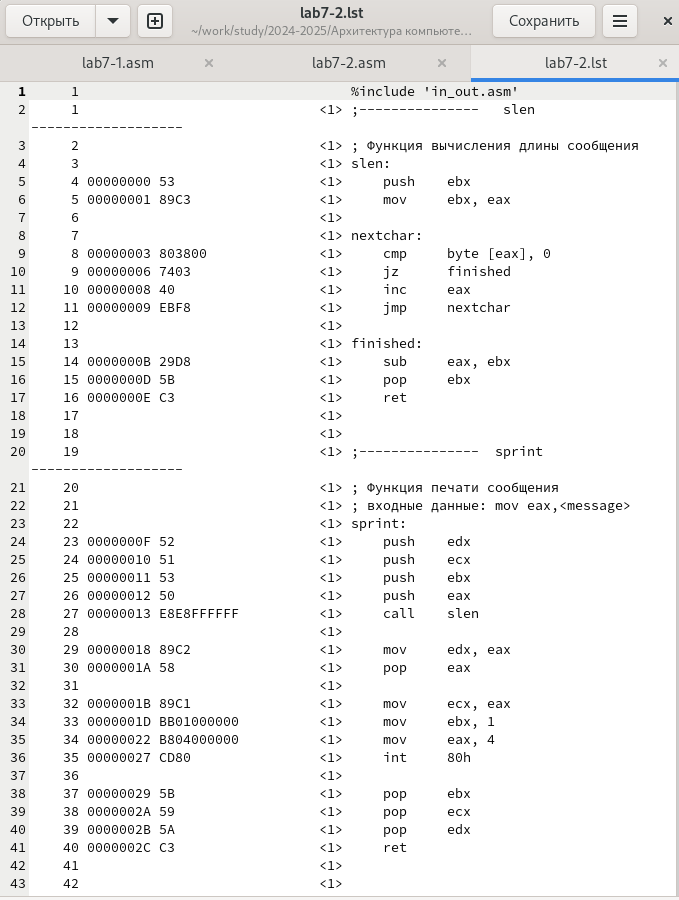


Рис. 10: Создание файла

Строка 5: 5 00000001 89C3 Данная команда перемещает данные из одного регистра в другой, сохраняет начальный адрес строки в регистре ebx. Строка 8: 00000003 803800 Данная команда проверяет, является ли текущий символ нулевым (конец строки). Строка 27: 00000013 E8E8FFFFFF Данная команда вызывает функцию slen, которая вычисляет длину строки.

Далее удаляю один операнд из случайной инструкции (рис. 11).

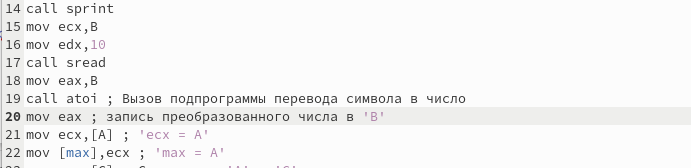


Рис. 11: Редактирование файла

Выполняю трансляцию с получением файла листинга, показывает ошибку.Никакие выходные при этом не создаются (рис. 12).

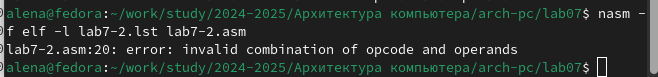


Рис. 12: Трансляция файла

## 4.3 Задания для самостоятельной работы

Вариант №9(мой вариант из лабораторной работы №6):

Редактирую программу для нахождения наименьшей из 3 целочисленных переменных (рис. 13).

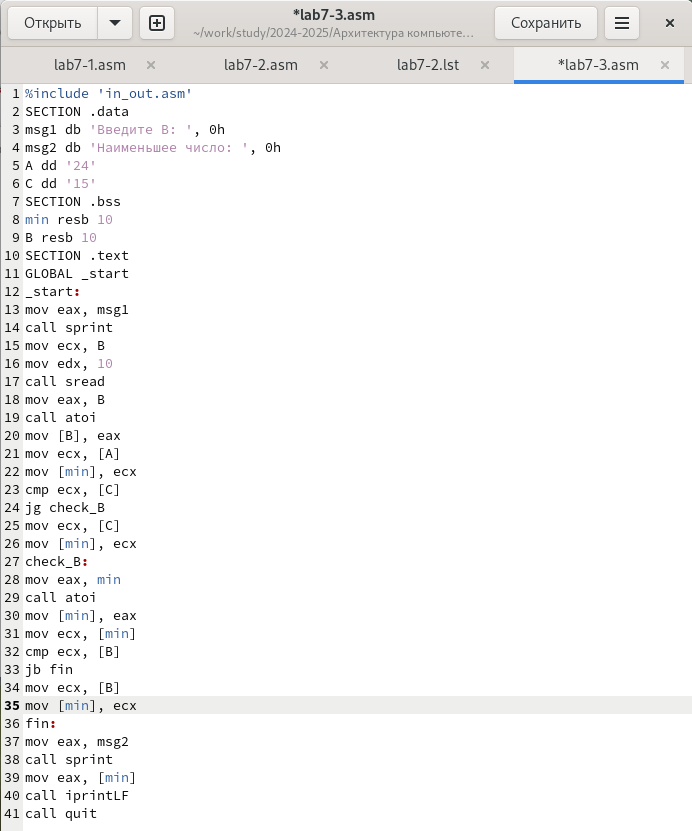


Рис. 13: Редактирование файла

Создаю исполняемый файл и запускаю его, программа работает корректно (рис. 14).

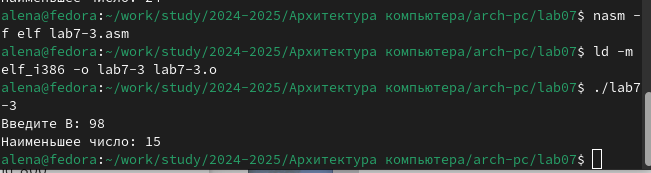


Рис. 14: Запуск исполняемого файла

%include 'in\_out.asm'  
SECTION .data  
msg1 db 'Введите В: ', 0h  
msg2 db 'Наименьшее число: ', 0h  
A dd '24'  
C dd '15'  
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

1. Пишу программу, которая вычисляет значение заданной функции и выводит результат вычислений.(рис. 15).

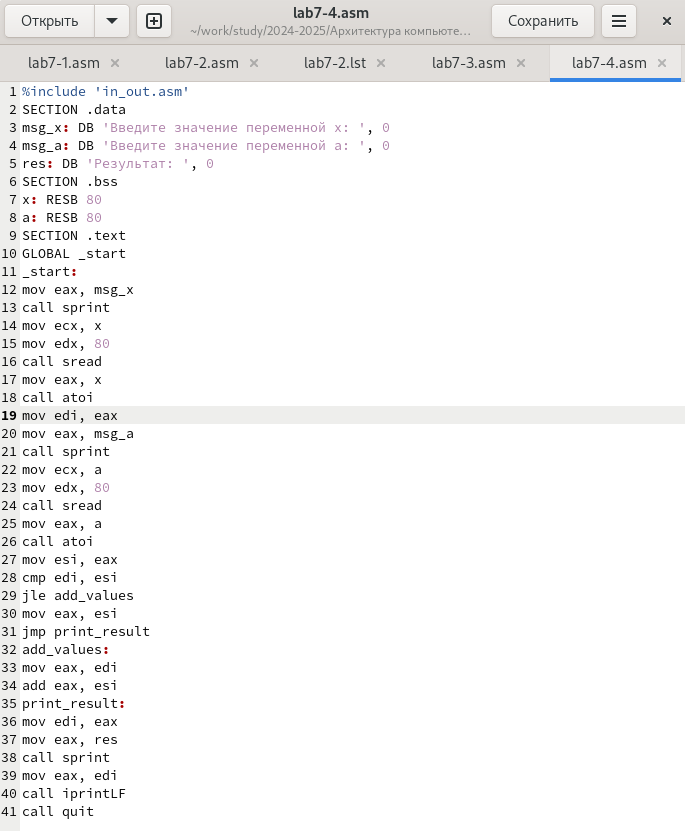


Рис. 15: Создание программы

Создаю исполняемый файл и запускаю его, программа работает корректно (рис. 16).

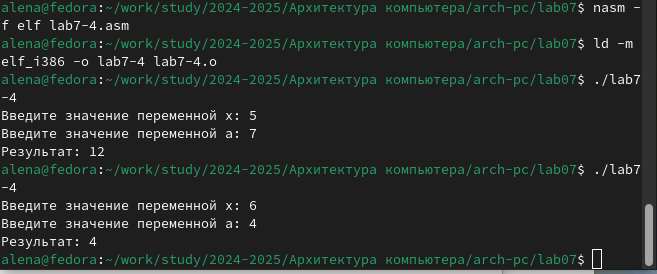


Рис. 16: Запуск исполняемого файла

%include 'in\_out.asm'  
SECTION .data  
msg\_x: DB 'Введите значение переменной x: ', 0  
msg\_a: DB 'Введите значение переменной 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, esi  
jle add\_values  
mov eax, esi  
jmp print\_result  
add\_values:  
mov eax, edi  
add eax, esi  
print\_result:  
mov edi, eax  
mov eax, res  
call sprint  
mov eax, edi  
call iprintLF  
call quit

# 5 Выводы

При выполнении данной лабораторной работы я изучила команды условного и безусловного переходов. Приобрела навыки написания программ с использованием переходов. Ознакомилась с назначением и структурой файла листинга.

# Список литературы

1. [Архитектура ЭВМ](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)