Отчёт по лабораторной работе №7

Дисциплина: Архитектура компьютера

Пономарева Татьяна Александровна

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 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 программы без каких-либо условий.

# 3 Выполнение лабораторной работы

## 3.1 Реализация переходов в NASM

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

Терминал. Создание каталога lab07. Переход в каталог lab07. Создание файла lab7-1.asm

Рис. 1: Терминал. Создание каталога lab07. Переход в каталог lab07. Создание файла lab7-1.asm

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

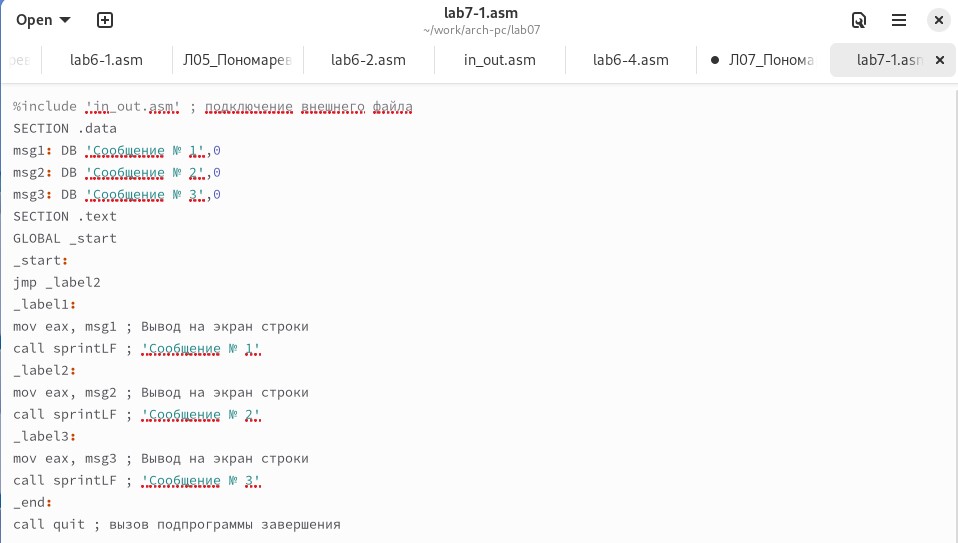


Рис. 2: Окно Text Editor. Содержание файла lab7-1.asm

Создаю исполняемый файл и запускаю его (рис. 3). Программа изменяет порядок вывода сообщений при использовании безусловного перехода.

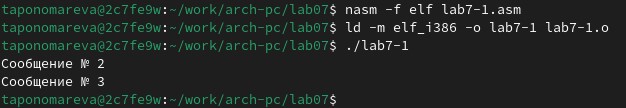


Рис. 3: Терминал. Создание исполняемого файла lab7-1 и его запуск

Изменяю файл lab7-1.asm в соответствии с Листингом 7.2 (рис. 4).

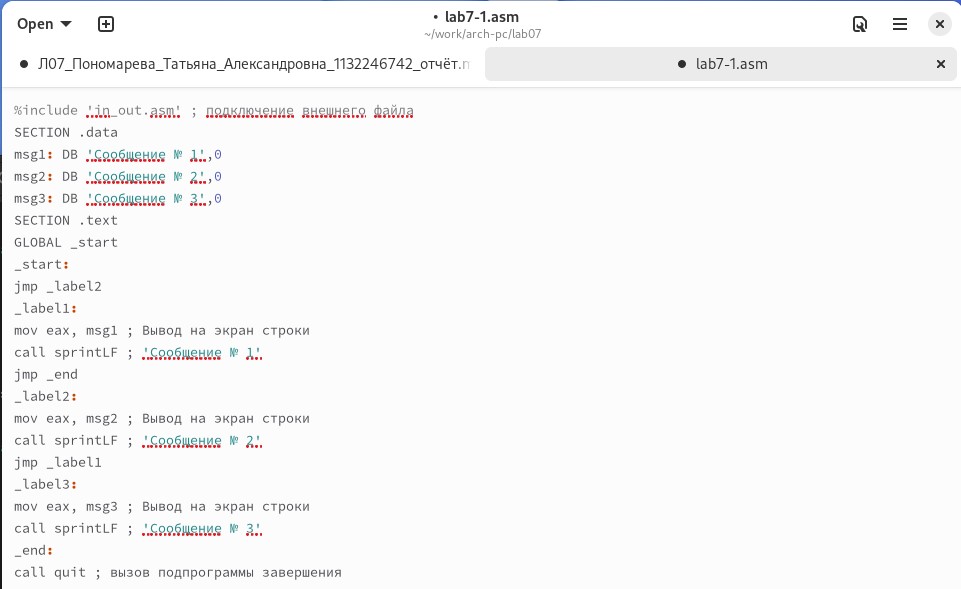


Рис. 4: Окно Text Editor. Содержание измененного файла lab7-1.asm

Создаю исполняемый файл, запускаю его и проверяю его работу (рис. 5). Программа выводит следующее: Сообщение №2, Сообщение №1.

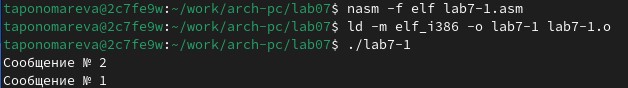


Рис. 5: Терминал. Создание исполняемого файла lab7-1 и его запуск

Изменяю lab7-1.asm (рис. 6).

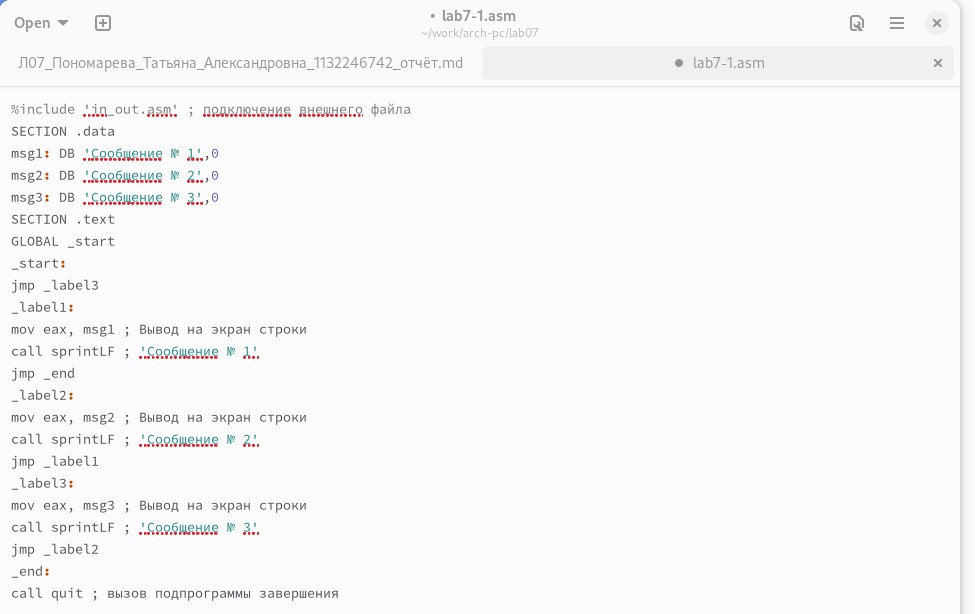


Рис. 6: Окно Text Editor. Содержание измененного файла lab7-1.asm

Создаю исполняемый файл, запускаю его и проверяю корректность его работы (рис. 7). При помощи неусловного перехода получаем такой порядок вывода: Сообщение №3, Сообщение №2, Сообщение №1.

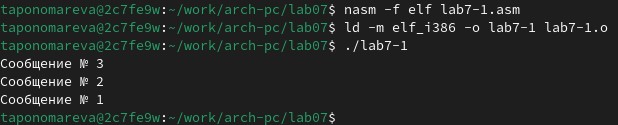


Рис. 7: Терминал. Создание исполняемого файла lab7-1 и его запуск

Создаю файл lab7-2.asm в каталоге ~/work/arch-pc/lab07 (рис. 8).

Терминал. Создание файла lab7-2.asm

Рис. 8: Терминал. Создание файла lab7-2.asm

Ввожу текст программы из листинга 7.3 в lab7-2.asm (рис. 9).

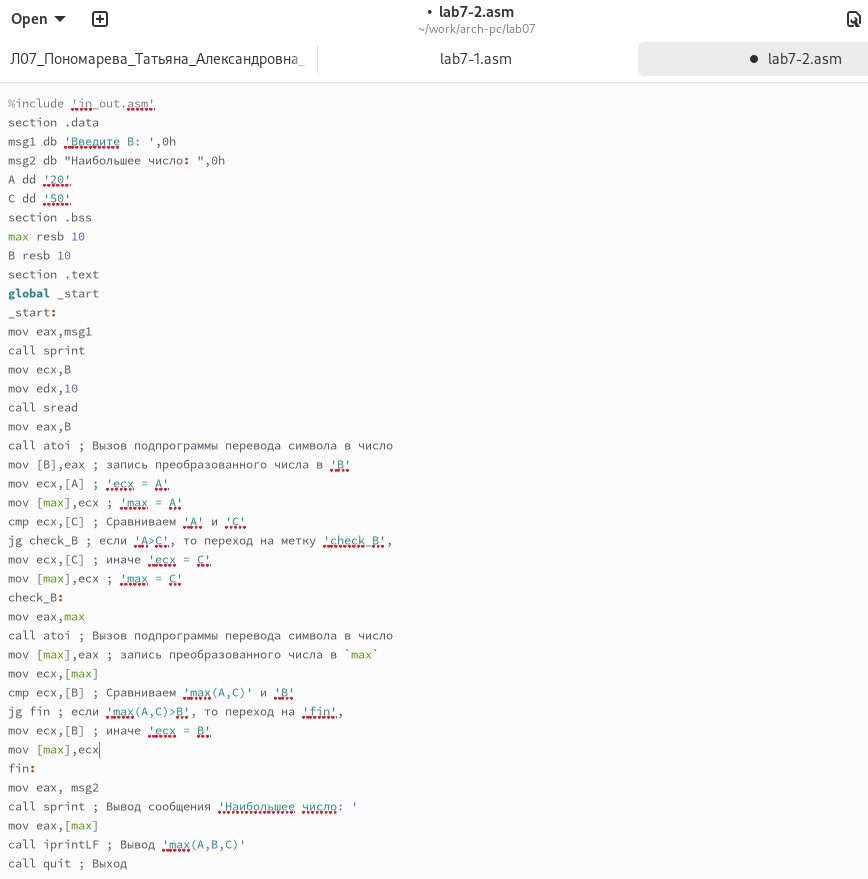


Рис. 9: Окно Text Editor. Содержание файла lab7-2.asm

Создаю исполняемый файл lab7-2, запускаю его и проверяю корректность его работы (рис. 10). Программа выводит наибольшее число.

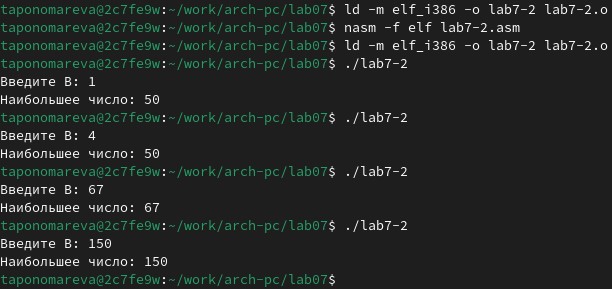


Рис. 10: Терминал. Создание исполняемого файла lab7-2 и проверка его работы на корректность

## 3.2 Изучение структуры файлы листинга

Создаю файл листинга для программы из файла lab7-2.asm при помощи команды nasm -f elf -l lab7-2.lst lab7-2.asm (рис. 11).

Терминал. Создание файла листинга lab7-2.lst

Рис. 11: Терминал. Создание файла листинга lab7-2.lst

Открываю файл листинга lab7-2.lst с помощью mcedit (рис. 12).

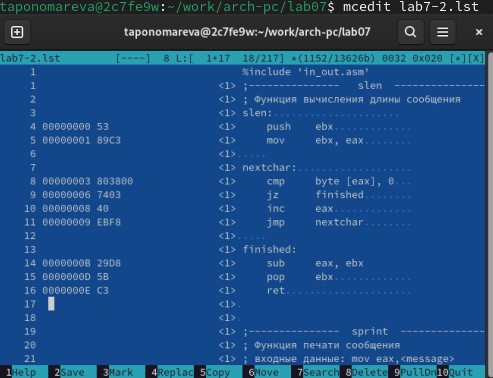


Рис. 12: Содержание листинга lab7-2.lst

Объяснение трех строк из листинга lab7-2.lst:

8 00000003 803800 <1> cmp byte [eax], 0 - Эта строка проверяет, равен ли байт по адресу, хранящемуся в EAX, нулю.  
  
9 00000006 7403 <1> jz finished - Если байт по адресу [eax] равен 0, выполнение программы переходит к метке finished. Если байт не равен 0, выполнение продолжится со следующей команды. 74 — код команды jz, а 03 — смещение (в байтах) для перехода. В данном случае переход произойдет на 3 байта вперед.  
  
10 00000008 40 <1> inc eax - Если байт по адресу [eax] не равен 0, то значение регистра EAX увеличивается на 1.  
  
8, 9, 10 - номера строк кода. 00000003, 00000006, 00000008 - смещения инструкций в памяти. 803800, 7403, 40 - машинный код инструкции, например, 803800 для byte [eax], 0 (80 указывает на инструкцию, которая выполняет арифметическую или логическую операцию между байтом в памяти (или регистре) и 8-битным числовым значением., 38 говорит процессору выполнить команду cmp (сравнение) для значения в памяти, адрес которой содержится в eax, 00 - значение, с которым мы сравниваем); 7403 для jz finished (03 - смещение на 3 байта вперед); 40 для inc eax.

Открываю файл lab7-2.asm и удаляю один из операндов (рис. 13).

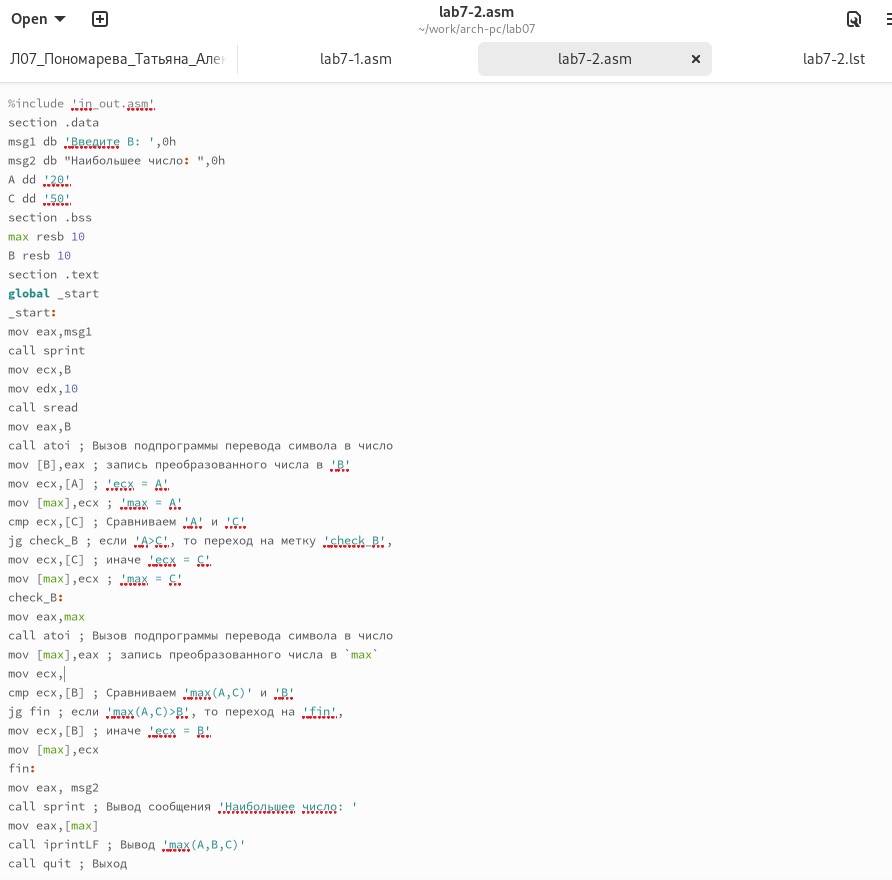


Рис. 13: Окно Text Editor. Содержание файла lab7-2.asm без операнда

Создаю файл листинга для программы из измененного файла lab7-2.asm и проверяю его на корректную работу (рис. 14).

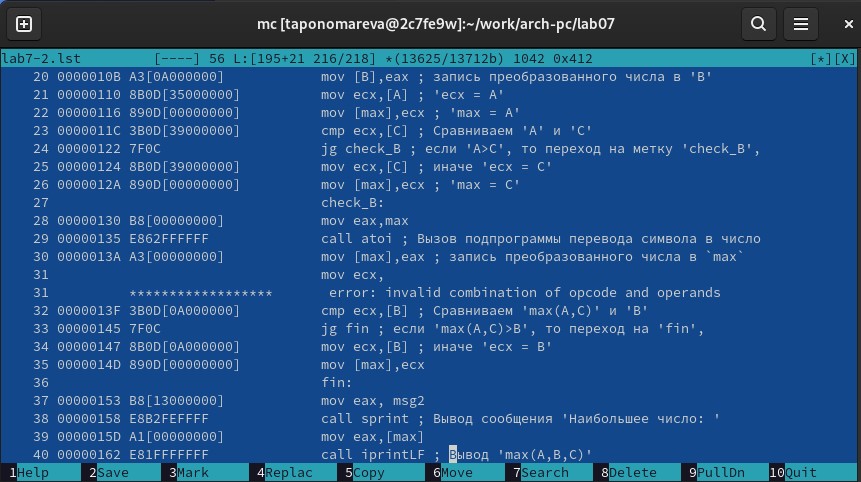


Рис. 14: Окно Midnight Commander. Содержание листинга lab7-2.lst

Файл листинга lab7-2.lst дает ошибку при транслировании файла. В этом случае никакие выходные файлы не создаются и ничего в листинг не добавляется.

# 4 Задания для самостоятельной работы

1. Из лабораторной работы №6 у меня вариант 3. Значения a, b, c: 94, 5, 58 Код программы, находящей наименьшее из 3 переменных из файла lab7-4-1.asm

%include 'in\_out.asm'  
  
SECTION .data  
msg1 db 'Введите B: ', 0h  
msg2 db 'Наименьшее число: ', 0h  
A dd 94   
C dd 58   
  
SECTION .bss  
B resd 1 ; Переменная для хранения числа B  
min resd 1 ; Переменная для хранения минимального числа  
  
SECTION .text  
GLOBAL \_start  
\_start:  
  
 ; Вывод сообщения "Введите B: "  
 mov eax, msg1  
 call sprint  
  
 ; Чтение строки и преобразование в число  
 mov ecx, B  
 mov edx, 10  
 call sread  
 mov eax, B  
 call atoi  
 mov [B], eax ; Сохраняем введенное число B  
  
 ; Инициализируем min значением A  
 mov eax, [A]  
 mov [min], eax  
  
 ; Сравнение с C  
 mov eax, [min]  
 cmp eax, [C]  
 jg check\_B  
 mov eax, [C]  
 mov [min], eax  
  
check\_B:  
 ; Сравнение с B  
 mov eax, [min]  
 cmp eax, [B]  
 jb fin  
 mov eax, [B]  
 mov [min], eax  
  
fin:  
 ; Вывод сообщения "Наименьшее число: "  
 mov eax, msg2  
 call sprint  
  
 ; Вывод результата  
 mov eax, [min]  
 call iprintLF  
  
 ; Завершение программы  
 call quit

Создаю исполняемый файл lab7-4-1, запускаю его и проверяю корректность его работы (рис. 15). Программа выводит наименьшее число.

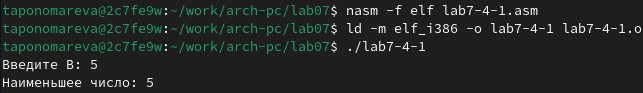


Рис. 15: Терминал. Создание исполняемого файла lab7-4-1 и его запуск

1. Вариант 3. Программа, которая принимает на вход x, a и выводит значение функции f(x).

%include 'in\_out.asm'  
  
SECTION .data  
msgX db 'Enter x: ', 0   
msgA db 'Enter a: ', 0   
resultMsg db 'f(x) = ', 0   
   
   
SECTION .bss  
x resd 1   
a resd 1   
result resd 1   
  
SECTION .text  
GLOBAL \_start  
  
\_start:  
 mov eax, msgX   
 call sprint  
 mov ecx, x   
 mov edx, 10   
 call sread  
 mov eax, x  
 call atoi   
 mov [x], eax   
   
 mov eax, msgA   
 call sprint  
 mov ecx, a   
 mov edx, 10   
 call sread  
 mov eax, a  
 call atoi   
 mov [a], eax   
  
 mov eax, [x]   
 cmp eax, 3   
 je compute\_3x   
  
 mov eax, [a]   
 add eax, 1   
 jmp display\_result   
  
compute\_3x:  
 mov eax, [x]   
 imul eax, 3   
  
display\_result:  
 mov [result], eax   
 mov eax, resultMsg   
 call sprint  
 mov eax, [result]   
 call iprintLF   
 call quit

Создаю исполняемый файл lab7-4-2, запускаю его и проверяю корректность его работы (рис. 16).

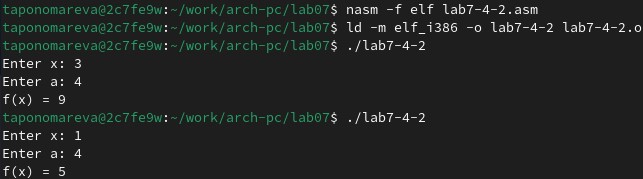


Рис. 16: Терминал. Создание исполняемого файла lab7-4-2 и его запуск

Загружаю на GitHub.

# 5 Выводы

В ходе лабораторной работы были изучены команды условного и безусловного переходов. Были приобретены навыки написания программ с использованием переходов. Также было произведено знакомство с назначением и структурой файла 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)