Отчет по лабораторной работе №7

Дисциплина: архитектура компьютера

Ермакова Анастасия Алексеевна

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

Цель данной лабораторной работы - изучение команды условного и безусловного переходов, преобретение навыков написания программ с использованием переходов, знакомство с назначением и структурой файла листинга.

# 2 Задание

1. Реализация переходов в NASM
2. Изучение структуры файла листинга
3. Выполнение заданий 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, перехожу в него, создаю там файл lab7-1.asm и открываю его для редактирования с помощью mousepad. (рис. 1).

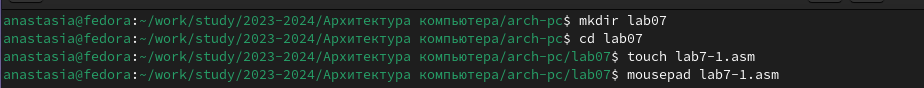


Рис. 1: Создание директории и файла

Ввожу в файл текст листинга 7.1. (рис. 2).

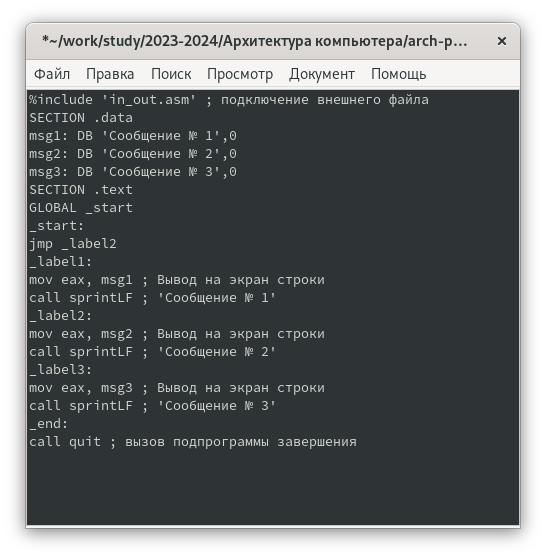


Рис. 2: Текст программы файла lab7-1.asm

Создаю исполняемый файл и запускаю его. Результат работы данной программы совпадаем с результатом в файле на ТУИС. (рис. 3).

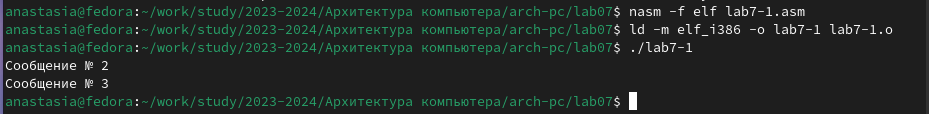


Рис. 3: Результат работы программы

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

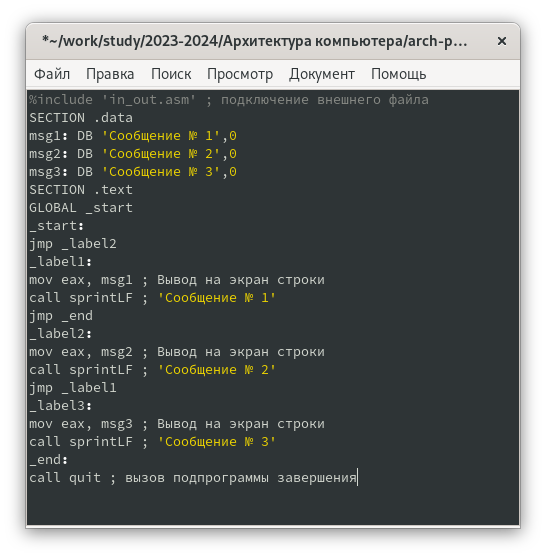


Рис. 4: Измененный текст файла

Создаю исполняемый файл и запускаю его. Программа выводит сначала Сообщение № 2, а затем Сообщение № 1, что соответствует заданию. (рис. 5).

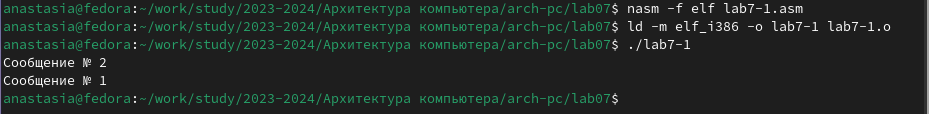


Рис. 5: Результат работы программы

Изменяю текст программы так, чтобы все три сообщения вывелись в обратном порядке. (рис. 6).

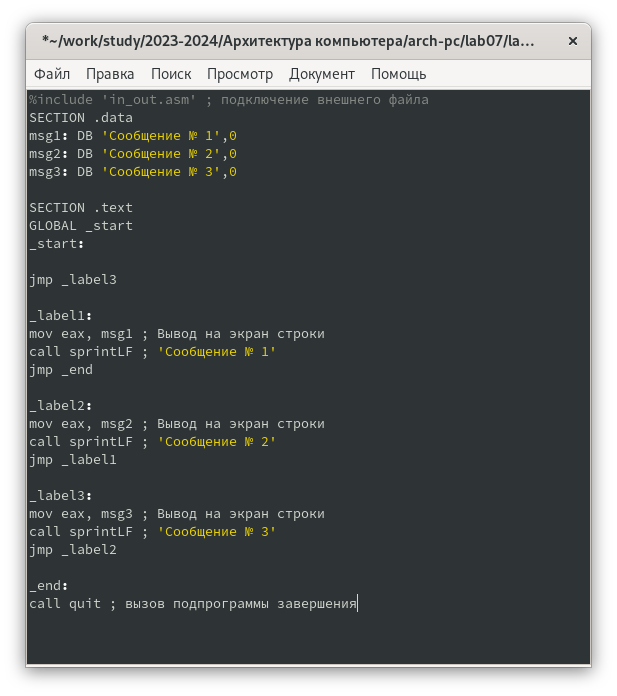


Рис. 6: Измененный текст программы

Создаю исполняемый файл и запускаю его. Программа работает верно. (рис. 7).

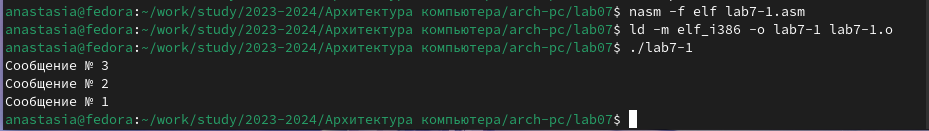


Рис. 7: Результат работы программы

Создаю файл lab7-2.asm в текущем каталоге и открываю его для редактирования с помощью mousepad. (рис. 8).

Рис. 8: Создание файла lab7-2.asm

Рис. 8: Создание файла lab7-2.asm

Ввожу в него текст программы из листинга 7.3. (рис. 9).



Рис. 9: Текст программы

Создаю исполняемый файл и запускаю его. Проверяю его работу для разных значений В (15, 35, 65). Программа работает исправно. (рис. 10).

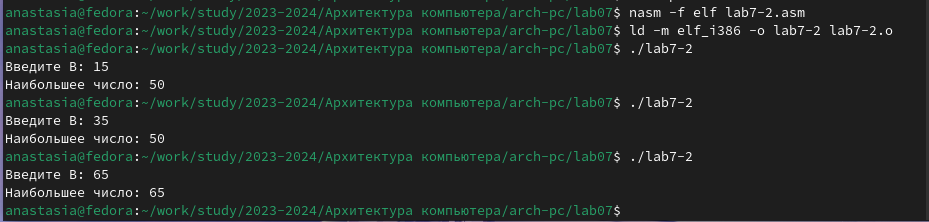


Рис. 10: Результат работы программы

## 4.2 Изучение структуры файла листинга

Создаю файл листинга для программы из файла lab7-2.asm и открваю его с помощью mcedit. (рис. 11).

Рис. 11: Создание файла листинга

Рис. 11: Создание файла листинга

Внимательно ознакамливаюсь с его форматом и содержимым. (рис. 12). Объясню следующие три строки листинга (строки 8, 9 и 10):

* cmp byte [eax], 0 (строка 8): Эта команда сравнивает байт, на который указывает регистр eax, с нулем. Если байт равен 0, флаг нуля (ZF) устанавливается в 1.
* jz finished (строка 9): Эта команда выполняет переход к метке finished, если флаг нуля установлен (то есть если байт, на который указывает eax, равен 0). Это условный переход, и если cmp не установит флаг, выполнение продолжится.
* inc eax (строка 10): Если предыдущая команда не привела к переходу, эта команда увеличивает значение в регистре eax на 1.

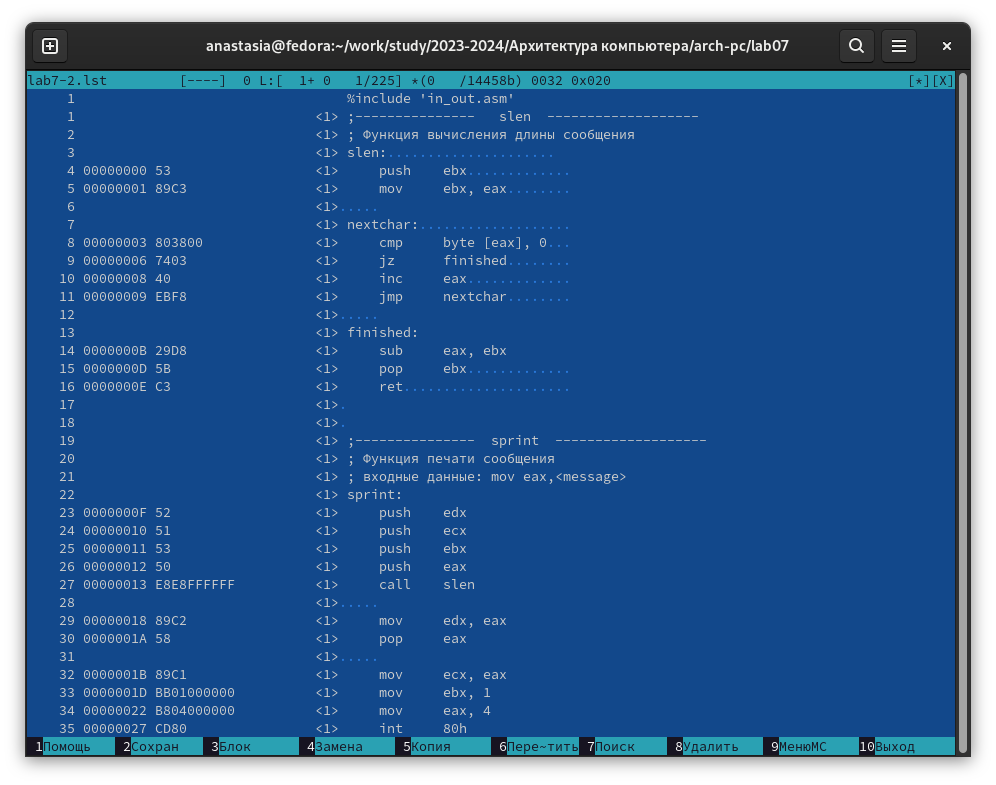


Рис. 12: Текст файла листинга

Открываю файл с программой lab7-2.asm и в инструкции с двумя операндами удаляю один из них. (рис. 13).



Рис. 13: Удаление одного операнда

Выполняю трансляцию с получением файла листинга. (рис. 14). Команда выводит ошибку.

Рис. 14: Выполнение трансляции

Рис. 14: Выполнение трансляции

В новом файле листинга показвает ошибку, возникшуу в результате трансляции файла. Никакие выходные данные при этом не создаются. (рис. 15).

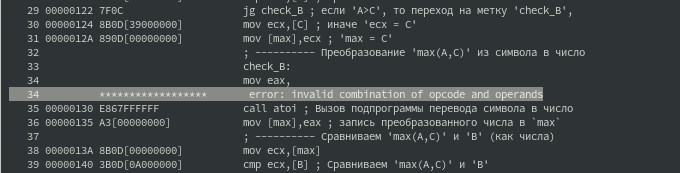


Рис. 15: Ошибка в файле листинга

## 4.3 Выполнение заданий для самостоятельной работы

Создаю файл lab7-3.asm для выолнения первого задания самостоятельной работы и пишу программу нахождения наименьшей из трех целочисленных переменных a, b и c. (рис. 16).

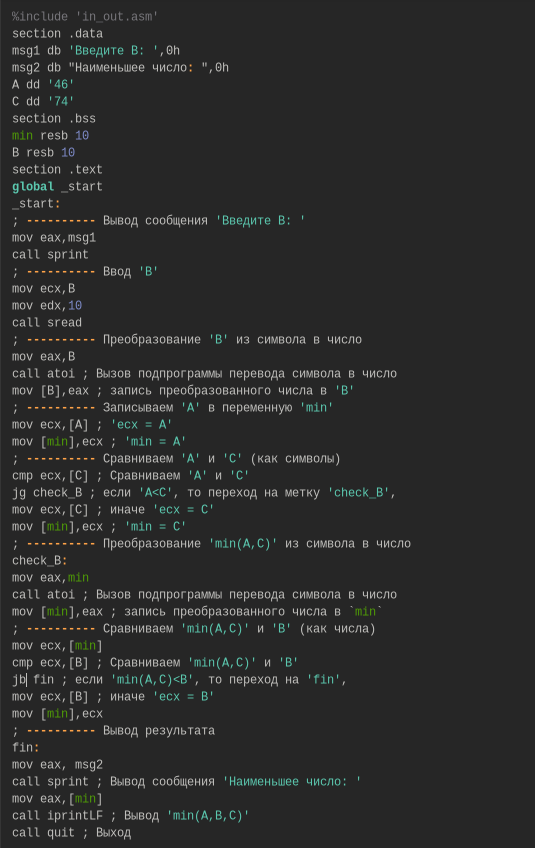


Рис. 16: Текст программы файла lab7-3.asm

Создаю исполняемый файл и запускаю его. Проверяю его работу, введя переменные из таблицы в соответствии со своим вариантом, полученным в результате выполнения предыдущей лабораторной работы (вариант 19). Ввожу значение 32, программа выводит результат, наименьшую переменную. Программа работает исправно, 32 и правда наименьшая из трех переменных. (рис. 17).

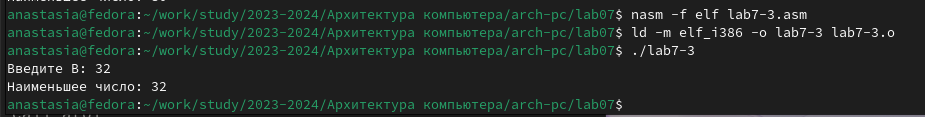


Рис. 17: Результат работы программы

Код программы:

%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 ; edi = x  
  
mov eax, msg\_a  
call sprint  
mov ecx, a  
mov edx, 80  
call sread  
mov eax, a  
call atoi  
mov esi, eax ; esi = a  
  
cmp edi,esi ; сравниваю x и a  
jg add\_values ; если х > a, перехожу в add\_values  
mov eax, edi ; если x <= a, вывожу х  
jmp print\_result   
  
add\_values: ; x > a, вывожу а + х  
mov eax, edi  
add eax,esi ; еах = а + х  
  
print\_result:  
mov edi, eax  
mov eax, res  
call sprint  
  
mov eax, edi  
call iprintLF  
call quit

Для второго задания самостоятельной работы создаю файл lab7-4.asm и пишу программу, которая для веденных с клавиатуры значений x и a вычисляет значение заданной функции f(x) и выводит результат вычислений. Выбираю вид функции f(x) в соответствии с моим вариантом 19. (рис. 18).

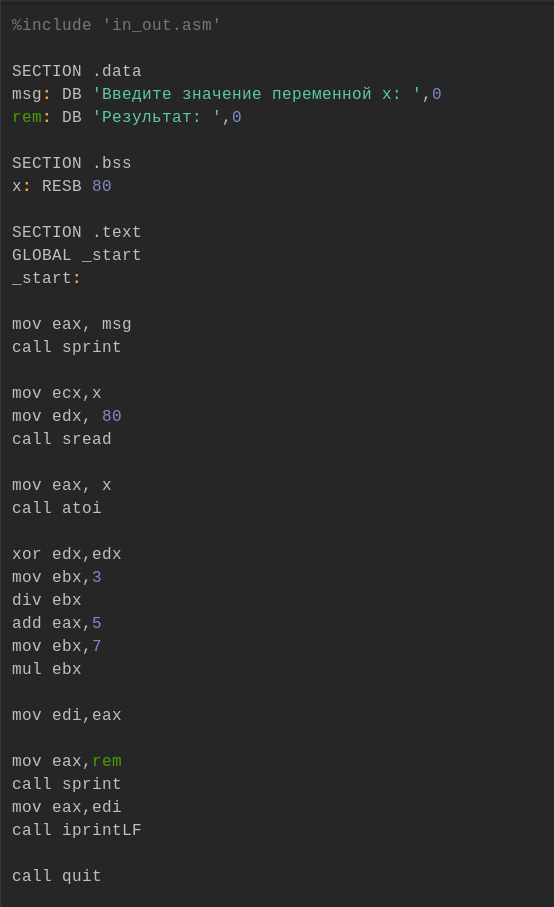


Рис. 18: Текст программы файла lab7-4.asm

Создаю исполняемый файл и запускаю его. Проверяю его работу, введя значения переменных из таблицы 7.6. Сначала ввожу первую пару переменных (4 и 5), затем вторую (3 и 2). Программа работает исправно. (рис. 19).

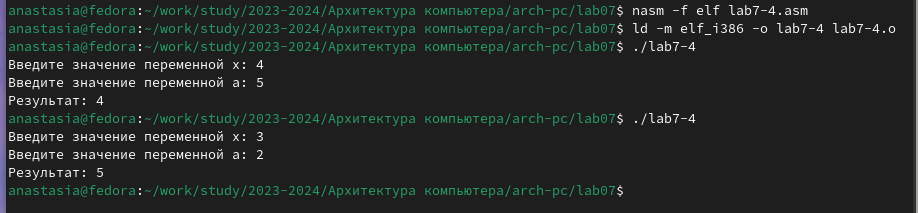


Рис. 19: Результат работы программы

Код программы:

%include 'in\_out.asm'  
section .data  
msg1 db 'Введите B: ',0h  
msg2 db "Наименьшее число: ",0h  
A dd '46'  
C dd '74'  
section .bss  
min resb 10  
B resb 10  
section .text  
global \_start  
\_start:  
; ---------- Вывод сообщения 'Введите B: '  
mov eax,msg1  
call sprint  
; ---------- Ввод 'B'  
mov ecx,B  
mov edx,10  
call sread  
; ---------- Преобразование 'B' из символа в число  
mov eax,B  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [B],eax ; запись преобразованного числа в 'B'  
; ---------- Записываем 'A' в переменную 'min'  
mov ecx,[A] ; 'ecx = A'  
mov [min],ecx ; 'min = A'  
; ---------- Сравниваем 'A' и 'С' (как символы)  
cmp ecx,[C] ; Сравниваем 'A' и 'С'  
jg check\_B ; если 'A<C', то переход на метку 'check\_B',  
mov ecx,[C] ; иначе 'ecx = C'  
mov [min],ecx ; 'min = C'  
; ---------- Преобразование 'min(A,C)' из символа в число  
check\_B:  
mov eax,min  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [min],eax ; запись преобразованного числа в `min`  
; ---------- Сравниваем 'min(A,C)' и 'B' (как числа)  
mov ecx,[min]  
cmp ecx,[B] ; Сравниваем 'min(A,C)' и 'B'  
jb fin ; если 'min(A,C)<B', то переход на 'fin',  
mov ecx,[B] ; иначе 'ecx = B'  
mov [min],ecx  
; ---------- Вывод результата  
fin:  
mov eax, msg2  
call sprint ; Вывод сообщения 'Наименьшее число: '  
mov eax,[min]  
call iprintLF ; Вывод 'min(A,B,C)'  
call quit ; Выход

# 5 Выводы

# 6 Список литературы