Отчёт по лабораторной работе №7

Дисциплина: архитектура компьютера.

Наговицын Арсений Владимирович

Содержание

# 1 Цель работы

Целью данной лабораторной работы является изучение команд условного и безусловного переходов, приобретение навыков написания программ с использованием переходов и знакомство с назначением и структурой файла листинга.

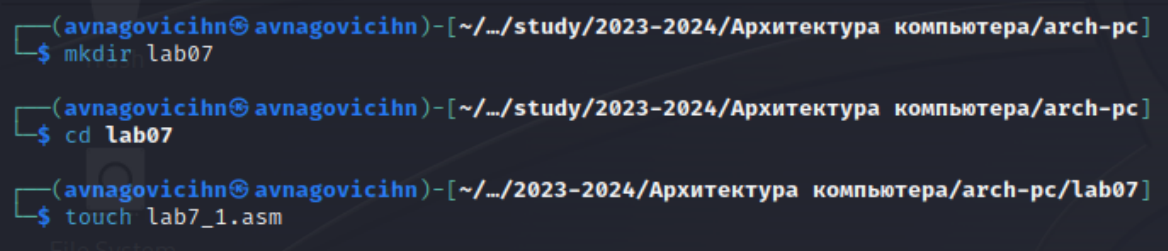
# 2 Задание

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

# 3 Выполнение лабораторной работы

## 3.1 Реализация переходов в NASM

Создаю каталог для программ и перехожу в него. Создаю файл (рис. ??).



Создание каталога и файла

Ввожу в файл текст программы из листинга 7.1(рис. ??).

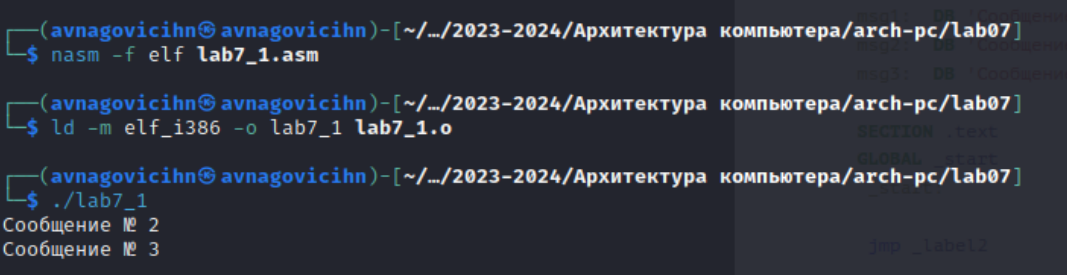


Редактирование файла

**Листинг 7.1 Программа с использованием инструкции jmp**

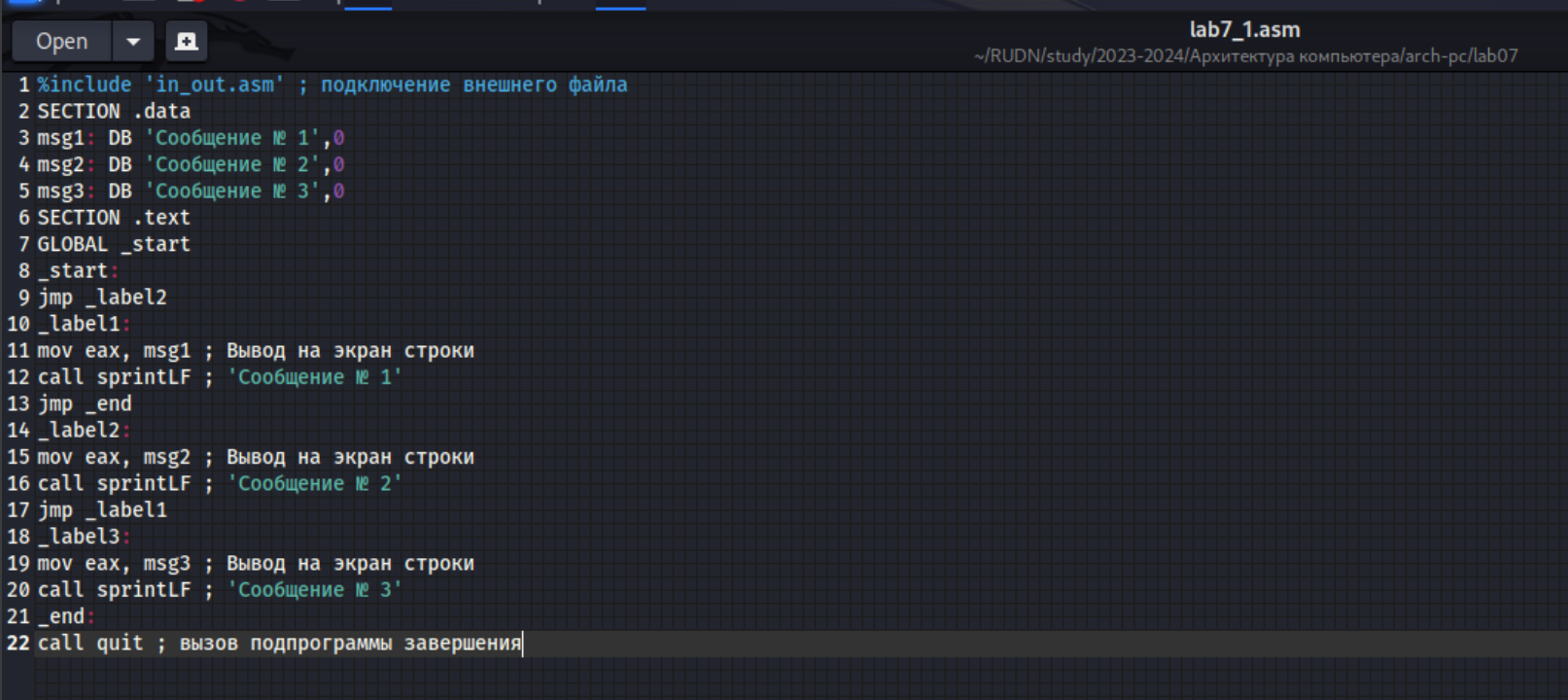
%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data  
msg1: DB 'Сообщение № 1',0  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL \_start  
\_start:  
jmp \_label2  
\_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 1'  
\_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 2'  
\_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 3'  
\_end:  
call quit ; вызов подпрограммы завершения

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



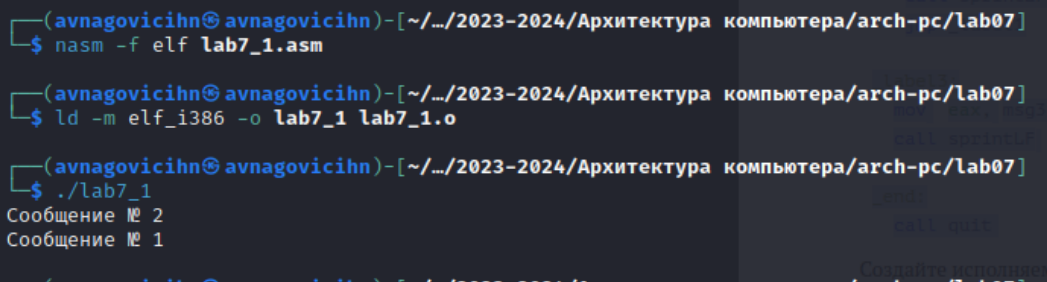
Компиляция и запуск файла

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



Редактирование файла

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

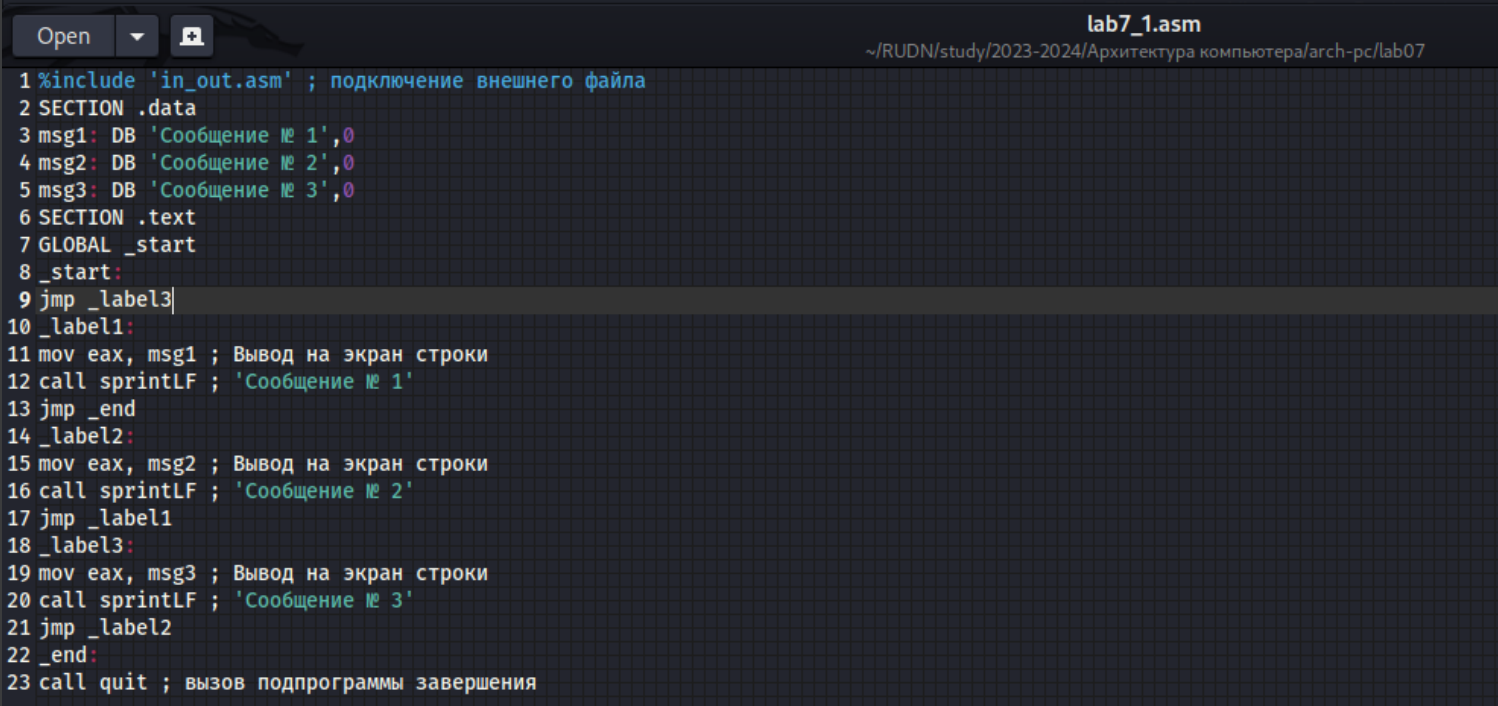


Компиляция и запуск файла

**Листинг 7.2 Программа с использованием инструкции jmp**

%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data  
msg1: DB 'Сообщение № 1',0  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL \_start  
\_start:  
jmp \_label2  
\_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 1'  
jmp \_end  
\_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 2'  
jmp \_label1  
\_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 3'  
\_end:  
call quit ; вызов подпрограммы завершения

Изменяю текст программы так, чтобы программа сначала выводила ‘Сообщение № 3’, затем ‘Сообщение № 2’, а затем ‘Сообщение № 1’ (рис. ??)

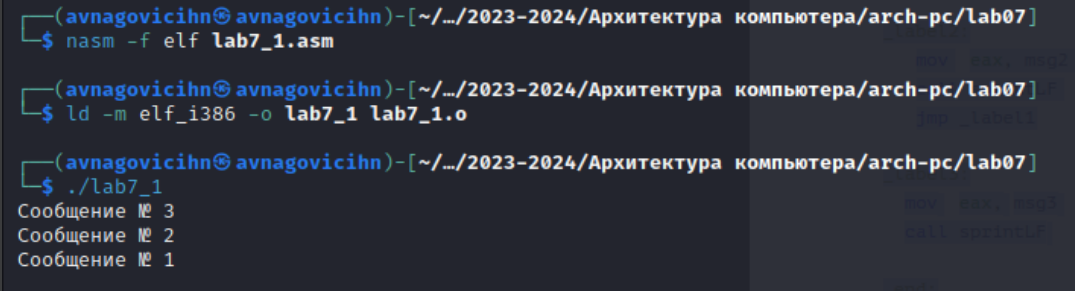


Редактирование файла

**Листинг 7.3 Программа с использованием инструкции jmp**

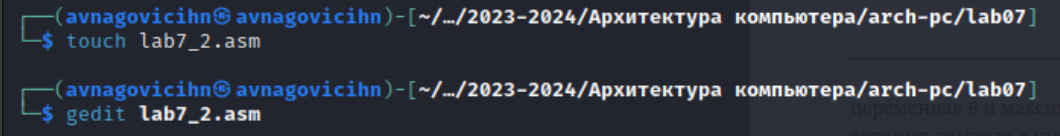
%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data  
msg1: DB 'Сообщение № 1',0  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL \_start  
\_start:  
jmp \_label3  
\_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 1'  
jmp \_end  
\_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 2'  
jmp \_label1  
\_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 3'  
jmp \_label2  
\_end:  
call quit ; вызов подпрограммы завершения

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



Компиляция и запуск файла

Создаю новый файл lab7\_2.asm (рис. ??).



Создание файла

Изменяю текст программы в соответствии с листингом 7.4 (рис. ??)

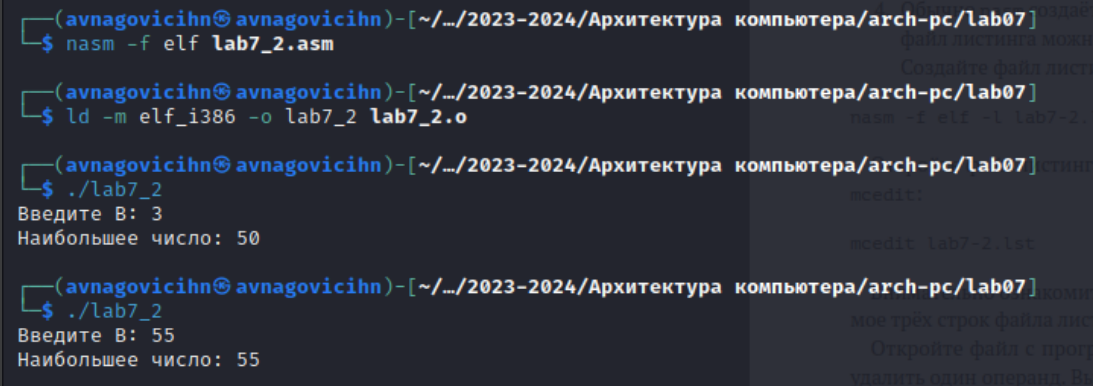


Редактирование файла

**Листинг 7.4 Программа с использованием инструкции jmp**

%include 'in\_out.asm'  
section .data  
msg1 db 'Введите B: ',0h  
msg2 db "Наибольшее число: ",0h  
A dd '20'  
C dd '50'  
section .bss  
max 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 'max'  
mov ecx,[A] ; 'ecx = A'  
mov [max],ecx ; 'max = 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 [max],ecx ; 'max = C'  
; ---------- Преобразование 'max(A,C)' из символа в число  
check\_B:  
mov eax,max  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [max],eax ; запись преобразованного числа в max  
; ---------- Сравниваем 'max(A,C)' и 'B' (как числа)  
mov ecx,[max]  
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'  
jg fin ; если 'max(A,C)>B', то переход на 'fin',  
mov ecx,[B] ; иначе 'ecx = B'  
mov [max],ecx  
; ---------- Вывод результата  
fin:  
mov eax, msg2  
call sprint ; Вывод сообщения 'Наибольшее число: '  
mov eax,[max]  
call iprintLF ; Вывод 'max(A,B,C)'  
call quit ; Выход

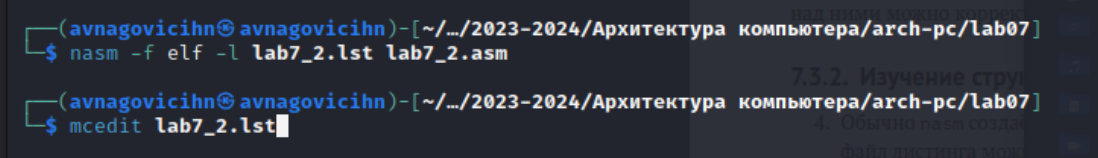
Создаю исполняемый файл, запускаю его и проверяю на нескольких значениях (рис. ??).



Компиляция и запуск файла

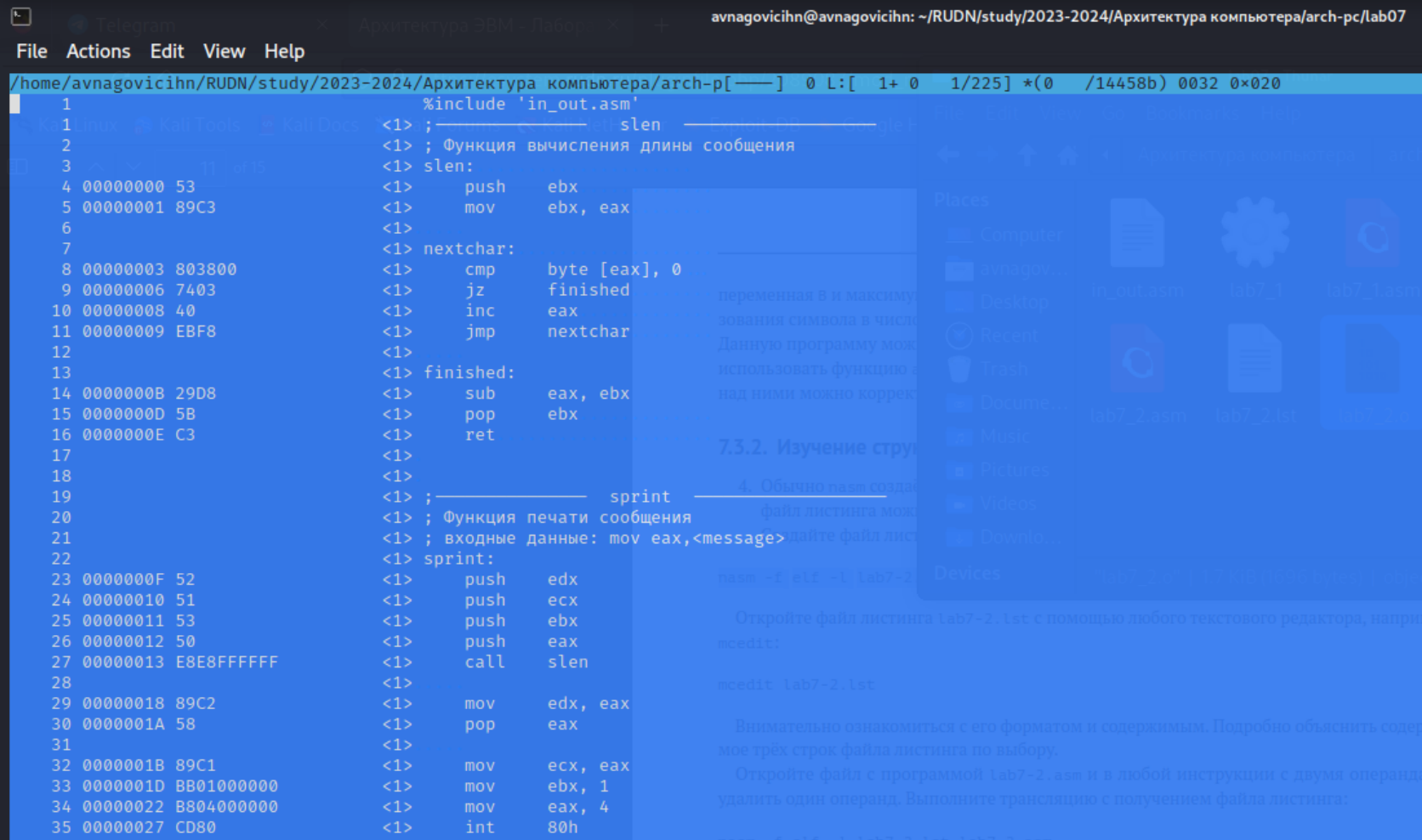
## 3.2 Изучение структуры файла листинга

Создаю файл листинга для программы из файла lab7-2.asm (рис. ??).



Создание файла

Открываю файл листинга (рис. ??).



Просмотр файла

**Объясняю содержимое первой выбранной строки:**

4 00000000 53 push ebx

4 - номер строки; 00000000 - адрес строки (смещение машинного кода от начала текущего сегмента); “53” - машинный код; “push ebx” - исходный текст программы. Инструкция “push” помещает операнд “ebx” в стек.

**Объясняю содержимое второй выбранной строки:**

29 00000018 89C2 mov edx,eax

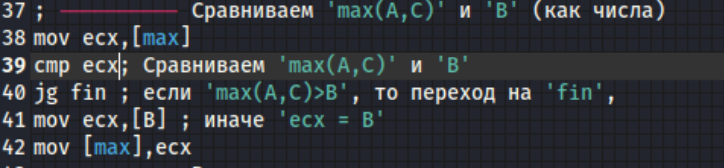
29 - номер строки; 00000018 - адрес строки (смещение машинного кода от начала текущего сегмента); “89C2” - машинный код; mov edx,eax - исходный текст программы. Инструкция на машинном языке, записывающая значение переменной eax в регистр edx

**Объясняю содержимое третьей выбранной строки:**

20 Функция печати сообщения

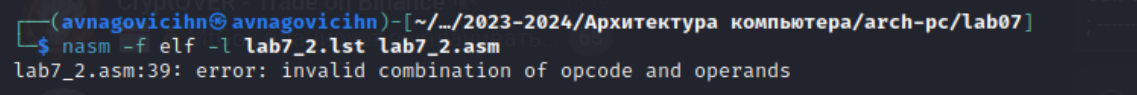
20 - номер строки; Функция печати сообщения - комментарий к коду, не имеет адреса, машинного кода.

Открываю файл с программой и удаляю операнд (рис. ??).



Редактирование файла

Выполняю трансляцию файла (рис. ??).

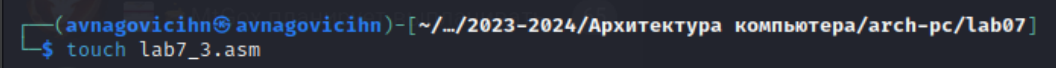


Трансляция файла

На выходе я не получаю никаких файлов, так как инструкция cmp подразумевает сравнение двух операндов.

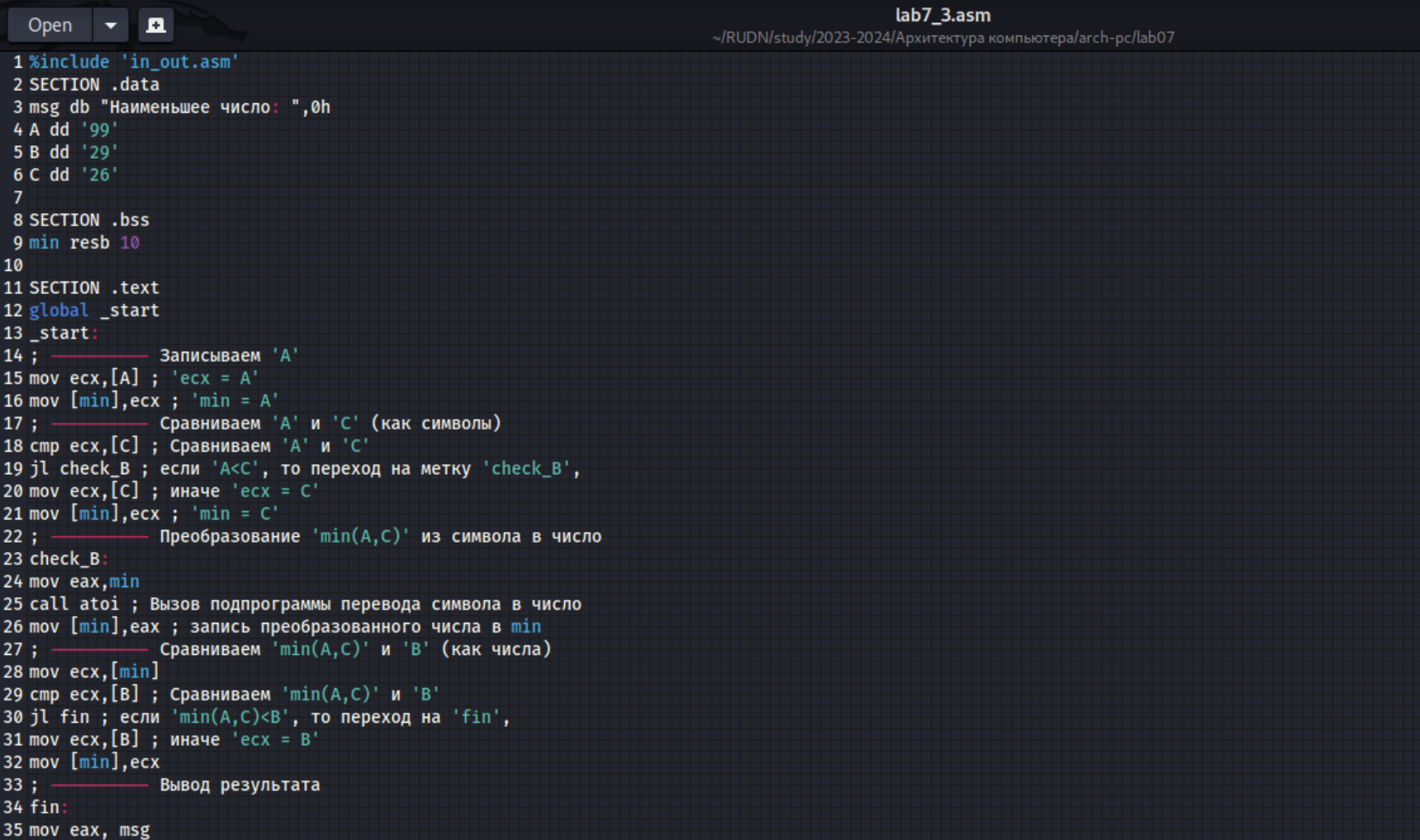
## 3.3 Выполнение заданий для самостоятельной работы

1. Создаю новый файл lab7\_3.asm (рис. ??).



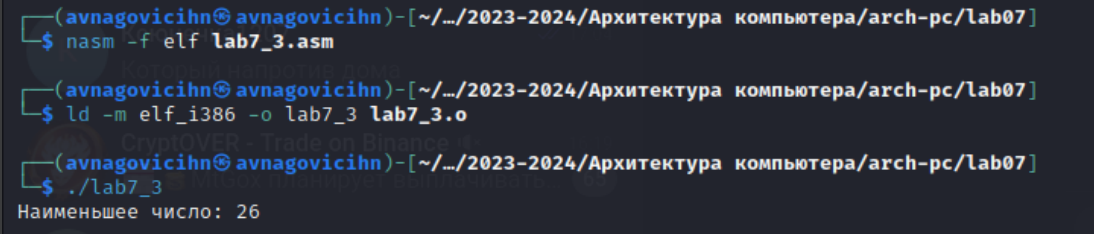
Создание файла

Открываю файл и пишу программу. Присваиваю переменным значения, указанные в 12 варианте (99,29,26) (рис. ??).



Редактирование файла

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

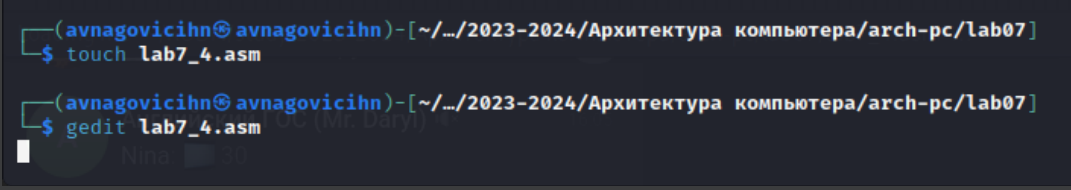


Компиляция и запуск файла

**Листинг 7.5. Программа для нахождения наименьшего числа**

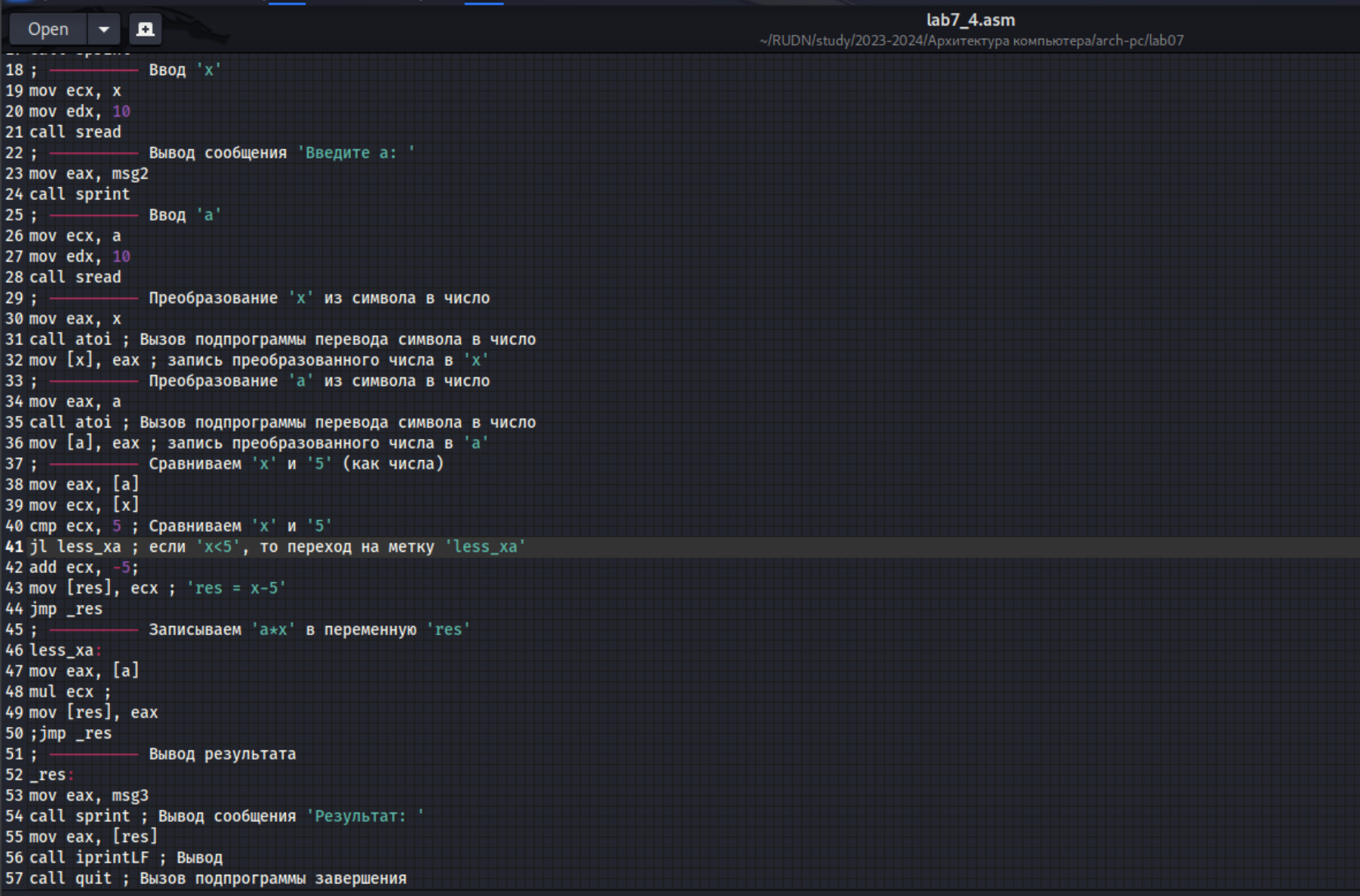
%include 'in\_out.asm'  
SECTION .data  
msg db "Наименьшее число: ",0h  
A dd '99'  
B dd '29'  
C dd '26'  
  
SECTION .bss  
min resb 10  
  
SECTION .text  
global \_start  
\_start:  
; ---------- Записываем 'A'  
mov ecx,[A] ; 'ecx = A'  
mov [min],ecx ; 'min = A'  
; ---------- Сравниваем 'A' и 'С' (как символы)  
cmp ecx,[C] ; Сравниваем 'A' и 'С'  
jl 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'  
jl fin ; если 'min(A,C)<B', то переход на 'fin',  
mov ecx,[B] ; иначе 'ecx = B'  
mov [min],ecx  
; ---------- 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

1. Создаю новый файл lab7\_4.asm (рис. ??).



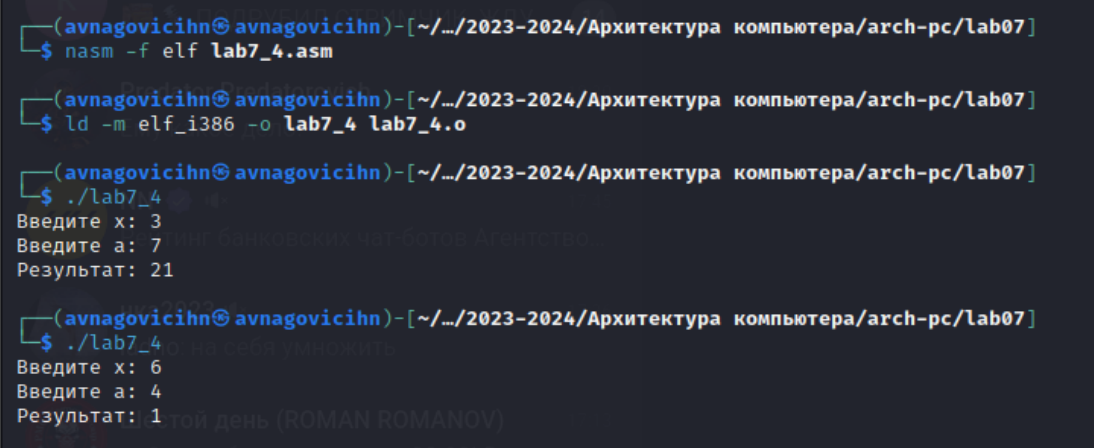
Компиляция и запуск файла

Открываю файл и пишу программу (рис. ??).



Редактирование файла

Создаю исполняемый файл и запускаю его. Ввожу значения в соответствии со своим вариантом ((3;7),(6;4)) (рис. ??).



Компиляция и запуск файла

**Листинг 7.6 Программа для вычисления заданной функции**

%include 'in\_out.asm'  
SECTION .data  
msg1 db 'Введите x: ', 0h  
msg2 db 'Введите a: ', 0h  
msg3 db 'Результат: ', 0h  
  
SECTION .bss  
x resb 11  
a resb 11  
res resb 12  
  
SECTION .text  
global \_start  
\_start:  
; ---------- Вывод сообщения 'Введите x: '  
mov eax, msg1  
call sprint  
; ---------- Ввод 'x'  
mov ecx, x  
mov edx, 10  
call sread  
; ---------- Вывод сообщения 'Введите a: '  
mov eax, msg2  
call sprint  
; ---------- Ввод 'a'  
mov ecx, a  
mov edx, 10  
call sread  
; ---------- Преобразование 'x' из символа в число  
mov eax, x  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [x], eax ; запись преобразованного числа в 'x'  
; ---------- Преобразование 'a' из символа в число  
mov eax, a  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [a], eax ; запись преобразованного числа в 'a'  
; ---------- Сравниваем 'x' и '5' (как числа)  
mov eax, [a]  
mov ecx, [x]  
cmp ecx, 5 ; Сравниваем 'x' и '5'  
jl less\_xa ; если 'x<5', то переход на метку 'less\_xa'  
add ecx, -5;  
mov [res], ecx ; 'res = x-5'  
jmp \_res  
; ---------- Записываем 'a\*x' в переменную 'res'  
less\_xa:  
mov eax, [a]  
mul ecx ;  
mov [res], eax   
;jmp \_res  
; ---------- Вывод результата  
\_res:  
mov eax, msg3  
call sprint ; Вывод сообщения 'Результат: '  
mov eax, [res]  
call iprintLF ; Вывод  
call quit ; Вызов подпрограммы завершения

# 4 Выводы

В ходе выполнения данной лабораторной работы я изучил команды условного и безусловного переходов, приобрёл навыки написания программ с использованием переходов и познакомился с назначением и структурой файла листинга.