

Отчёта по лабораторной работе 8

**Команды безусловного и условного переходов в Nasm.
Программирование ветвлений**

Одеджими Олуваколаде Осемудиам

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	20

Список иллюстраций

2.1	Файл lab8-1.asm:	7
2.2	Программа lab8-1.asm:	8
2.3	Файл lab8-1.asm:	9
2.4	Программа lab8-1.asm:	9
2.5	Файл lab8-1.asm	10
2.6	Программа lab8-1.asm	11
2.7	Файл lab8-2.asm	12
2.8	Программа lab8-2.asm	13
2.9	Файл листинга lab8-2	14
2.10	ошибка трансляции lab8-2	15
2.11	файл листинга с ошибкой lab8-2	16
2.12	Файл lab8-3.asm	17
2.13	Программа lab8-3.asm	18
2.14	Файл lab8-4.asm	19
2.15	Программа lab8-4.asm	19

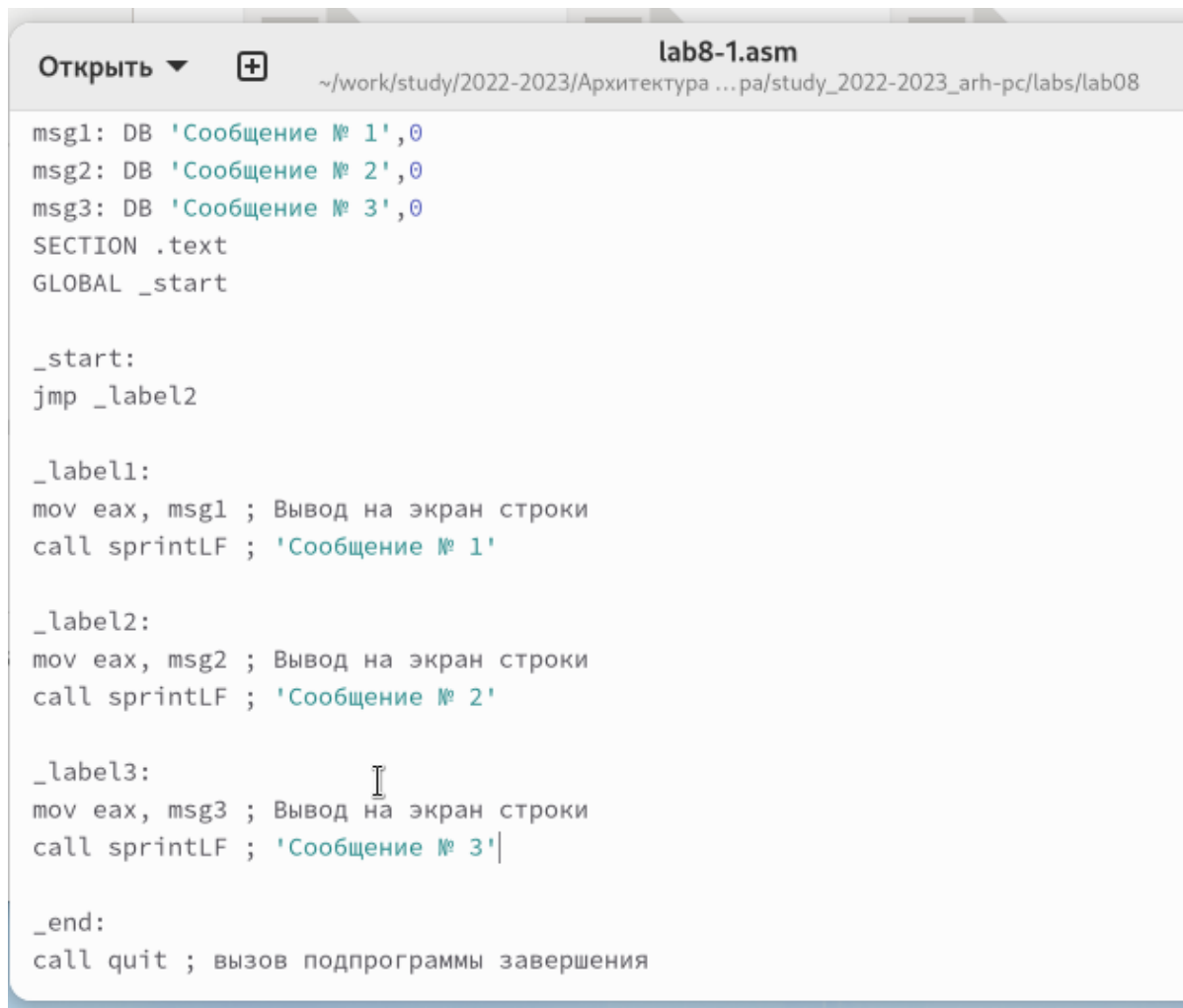
Список таблиц

1 Цель работы

Целью работы является изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

2 Выполнение лабораторной работы

1. Создайте каталог для программ лабораторной работы № 8, перейдите в него и создайте файл lab8-1.asm
2. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции jmp. Введите в файл lab8-1.asm текст программы из листинга 8.1. (рис. [2.1])



```
lab8-1.asm
~/work/study/2022-2023/Архитектура ... pa/study_2022-2023_arh-pc/labs/lab08

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 sprintf ; 'Сообщение № 1'

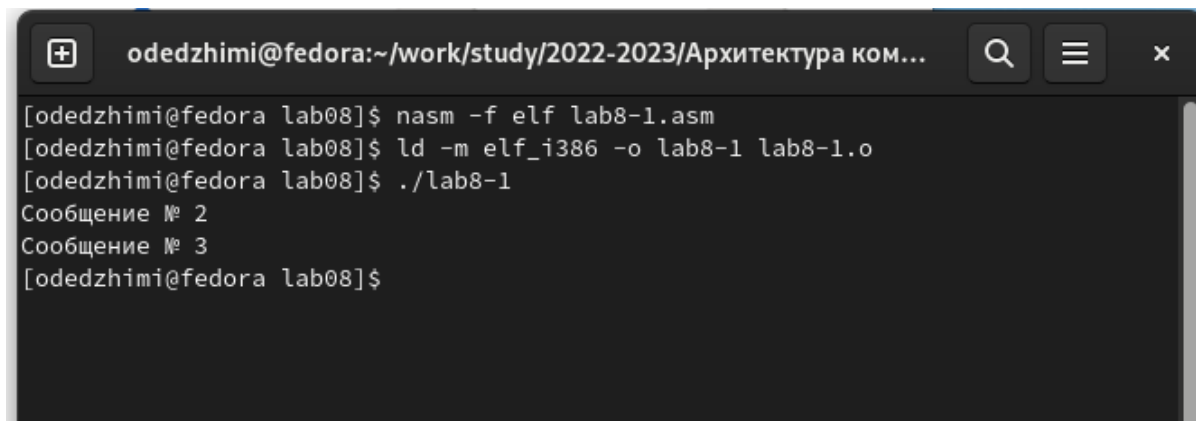
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintf ; 'Сообщение № 2'

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintf ; 'Сообщение № 3'

_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.1: Файл lab8-1.asm:

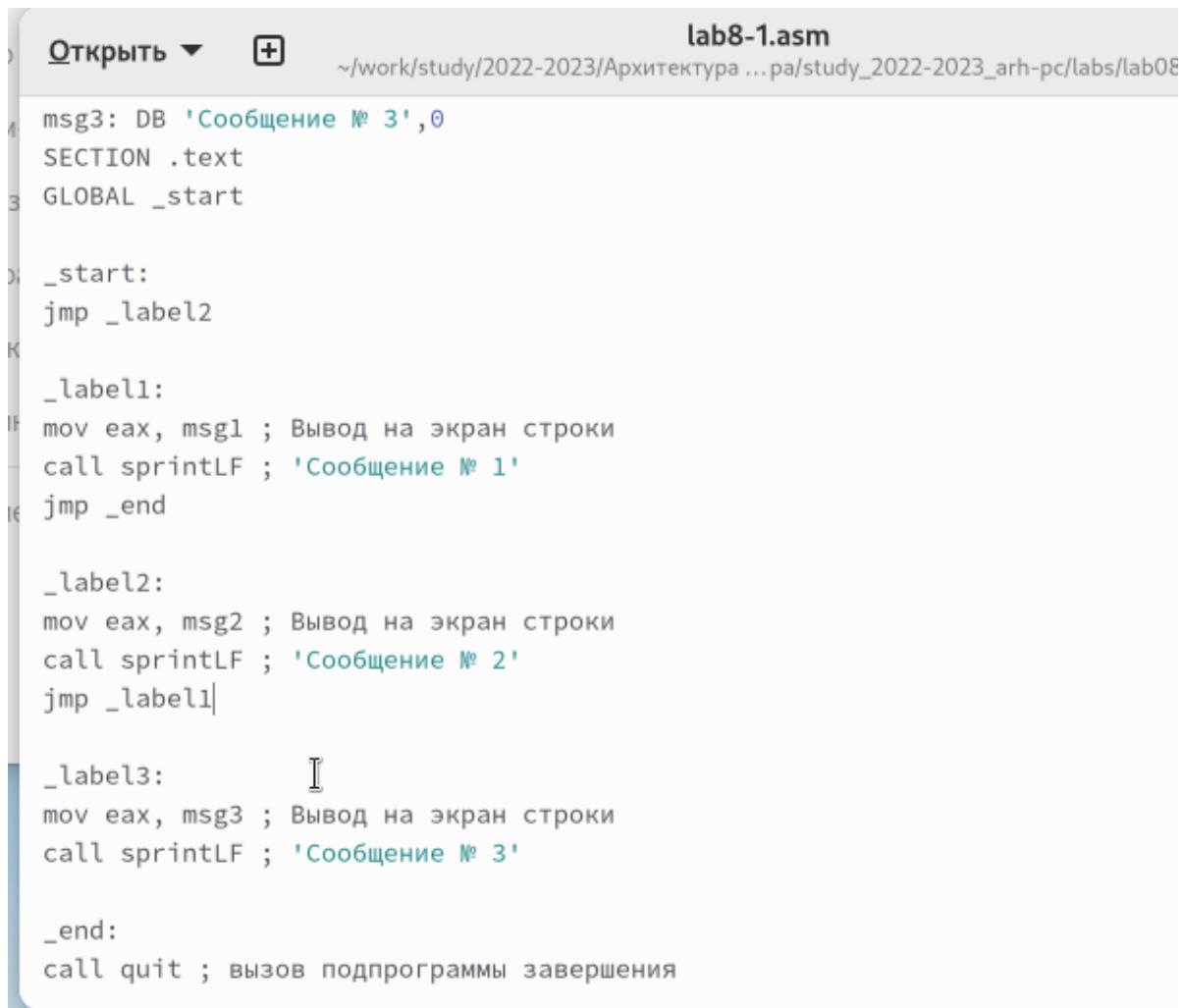
Создайте исполняемый файл и запустите его. (рис. [2.2])

A terminal window with a dark background. The title bar shows the user 'odedzhimi@fedora' and the path '~/work/study/2022-2023/Архитектура ком...'. The terminal contains the following text:

```
[odedzhimi@fedora lab08]$ nasm -f elf lab8-1.asm
[odedzhimi@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[odedzhimi@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 3
[odedzhimi@fedora lab08]$
```

Рис. 2.2: Программа lab8-1.asm:

Инструкция `jmp` позволяет осуществлять переходы не только вперед но и назад. Изменим программу таким образом, чтобы она выводила сначала 'Сообщение № 2', потом 'Сообщение № 1' и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию `jmp` с меткой `_label1` (т.е. переход к инструкциям вывода сообщения № 1) и после вывода сообщения № 1 добавим инструкцию `jmp` с меткой `_end` (т.е. переход к инструкции `call quit`). Измените текст программы в соответствии с листингом 8.2. (рис. [2.3], [2.4])



```
lab8-1.asm
~/work/study/2022-2023/Архитектура ... pa/study_2022-2023_arh-pc/labs/lab08

msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start

_start:
jmp _label2

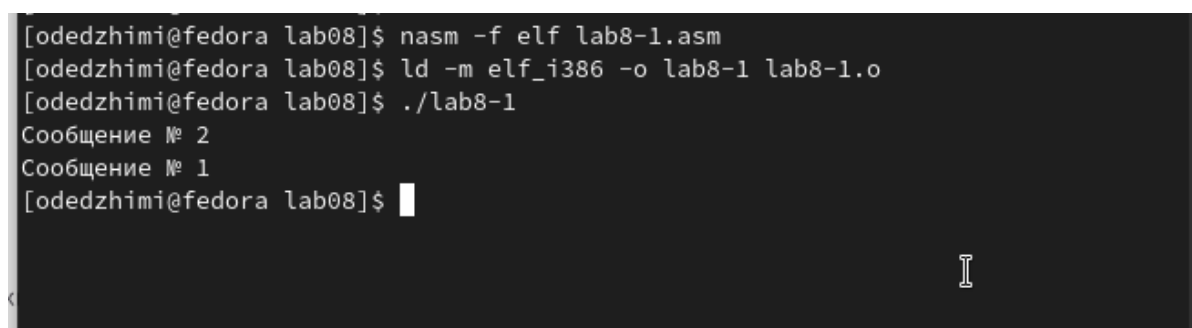
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 1'
jmp _end

_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 2'
jmp _label1

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 3'

_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.3: Файл lab8-1.asm:



```
[odedzhimi@fedora lab08]$ nasm -f elf lab8-1.asm
[odedzhimi@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o
[odedzhimi@fedora lab08]$ ./lab8-1
Сообщение № 2
Сообщение № 1
[odedzhimi@fedora lab08]$
```

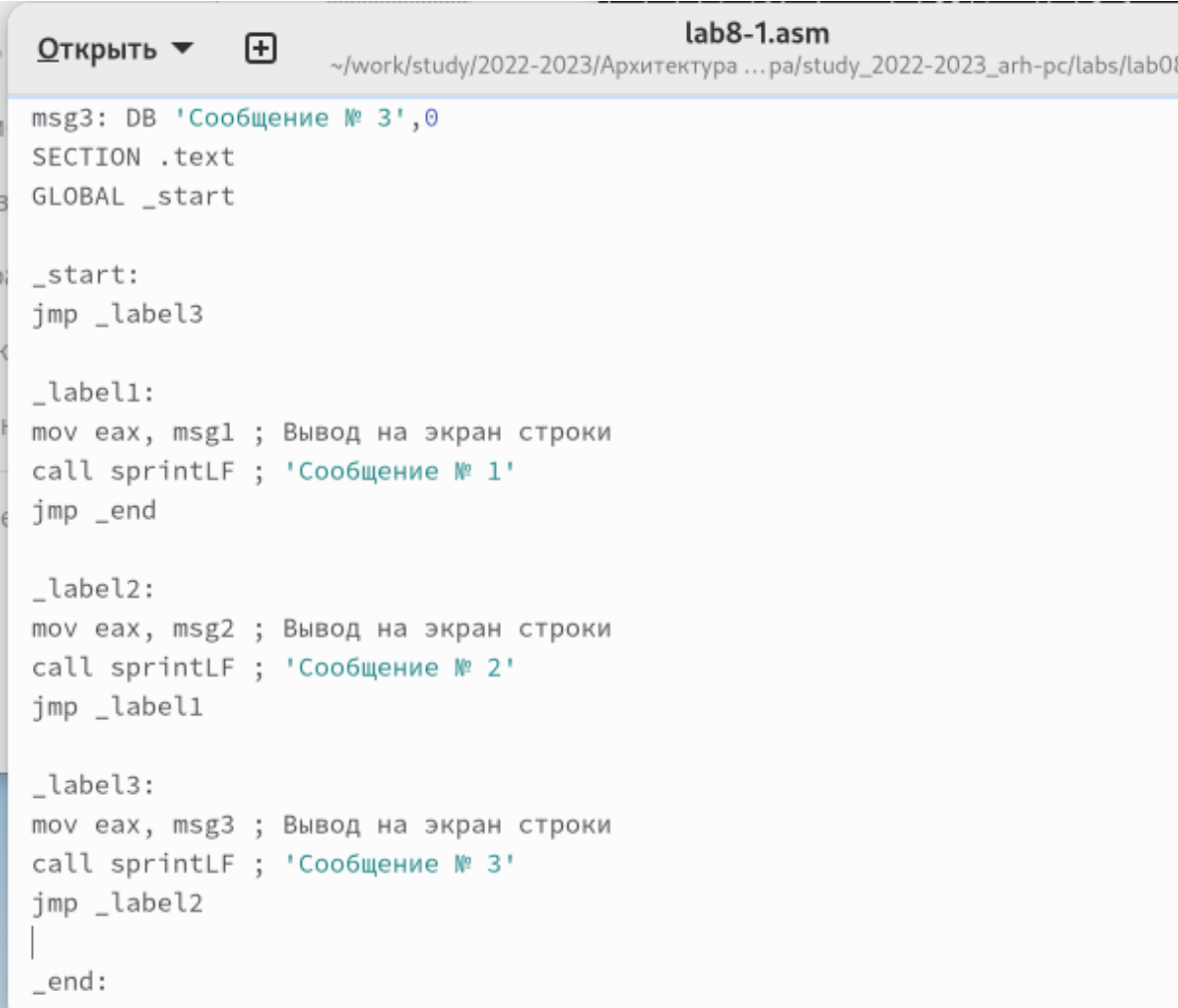
Рис. 2.4: Программа lab8-1.asm:

Измените текст программы добавив или изменив инструкции `jmp`, чтобы вывод программы был следующим (рис. [2.5], [2.6]):

Сообщение № 3

Сообщение № 2

Сообщение № 1



```
lab8-1.asm
~/work/study/2022-2023/Архитектура ...pa/study_2022-2023_arh-pc/labs/lab08

msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start

_start:
jmp _label3

_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 1'
jmp _end

_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 2'
jmp _label1

_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintfLF ; 'Сообщение № 3'
jmp _label2

_end:
call quit ; вызов подпрограммы завершения
```

Рис. 2.5: Файл lab8-1.asm

```
[odedzhimi@fedora lab08]$  
[odedzhimi@fedora lab08]$ nasm -f elf lab8-1.asm  
[odedzhimi@fedora lab08]$ ld -m elf_i386 -o lab8-1 lab8-1.o  
[odedzhimi@fedora lab08]$ ./lab8-1  
Сообщение № 3  
Сообщение № 2  
Сообщение № 1  
[odedzhimi@fedora lab08]$
```

Рис. 2.6: Программа lab8-1.asm

3. Использование инструкции `jmp` приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, т.е. переход должен происходить если выполнено какое-либо условие. В качестве примера рассмотрим программу, которая определяет и выводит на экран наибольшую из 3 целочисленных переменных: А, В и С. Значения для А и С задаются в программе, значение В вводится с клавиатуры. Создайте исполняемый файл и проверьте его работу для разных значений В. (рис. [2.7], [2.8])

Открыть + lab8-2.asm ~\work\study\2022-2023\Архитектура ...pa\study_2022-2023_arh-pc\labs\lab08

```
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 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
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 sprintf ; Вывод сообщения 'Наибольшее число: '
mov eax,[max]
call iprintLF ; Вывод 'max(A,B,C)'
call quit ; Выход
```

Рис. 2.7: Файл lab8-2.asm

```
[odedzhimi@fedora lab08]$  
[odedzhimi@fedora lab08]$ nasm -f elf lab8-2.asm  
[odedzhimi@fedora lab08]$ ld -m elf_i386 -o lab8-2 lab8-2.o  
[odedzhimi@fedora lab08]$ ./lab8-2  
Введите B: 150  
Наибольшее число: 150  
[odedzhimi@fedora lab08]$ ./lab8-2  
Введите B: 20  
Наибольшее число: 50  
[odedzhimi@fedora lab08]$
```

Рис. 2.8: Программа lab8-2.asm

4. Обычно `nasm` создаёт в результате ассемблирования только объектный файл. Получить файл листинга можно, указав ключ `-l` и задав имя файла листинга в командной строке. Создайте файл листинга для программы из файла `lab8-2.asm` (рис. [2.9])

Рис. 2.9: Файл листинга lab8-2

Внимательно ознакомиться с его форматом и содержимым. Подробно объяснить содержимое трёх строк файла листинга по выбору.

строка 51

- 51 - номер строки
- 00000033 - адрес
- B80A000000 - машинный код

- `mov eax, 0AH` - код программы

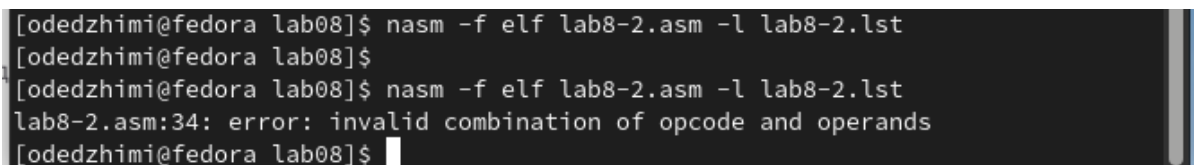
строка 52

- 52 - номер строки
- 00000038 - адрес
- 50 - машинный код
- `push eax` - код программы

строка 53

- 53 - номер строки
- 00000039 - адрес
- 89E0 - машинный код
- `mov eax, esp` - код программы

Откройте файл с программой `lab8-2.asm` и в любой инструкции с двумя операндами удалить один операнд. Выполните трансляцию с получением файла листинга (рис. [2.10],[2.11])

A screenshot of a terminal window with a dark background. The text shows a user at the prompt [odedzhimi@fedora lab08]\$ running the command `nasm -f elf lab8-2.asm -l lab8-2.lst`. The command is repeated twice. The second time, an error message is displayed: `lab8-2.asm:34: error: invalid combination of opcode and operands`. The prompt returns to [odedzhimi@fedora lab08]\$.

```
[odedzhimi@fedora lab08]$ nasm -f elf lab8-2.asm -l lab8-2.lst
[odedzhimi@fedora lab08]$
[odedzhimi@fedora lab08]$ nasm -f elf lab8-2.asm -l lab8-2.lst
lab8-2.asm:34: error: invalid combination of opcode and operands
[odedzhimi@fedora lab08]$
```

Рис. 2.10: ошибка трансляции `lab8-2`

```

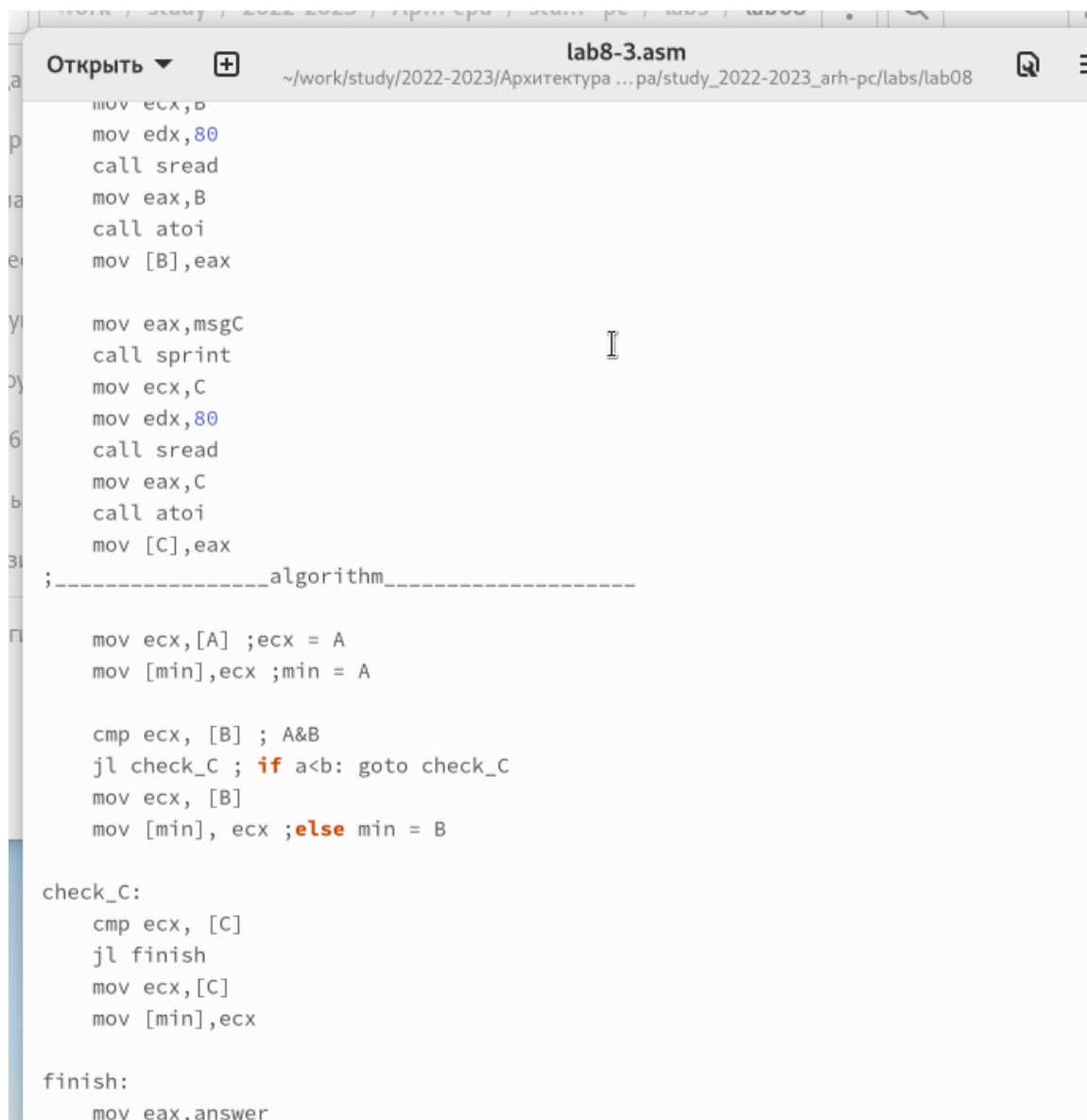
lab8-2.asm                                lab8-2.lst
29 00000122 7F0C                          jg check_B ; если 'A>C', то переход на метку
'check_B',
30 00000124 8B0D[39000000]                      mov ecx,[C] ; иначе 'ecx = C'
31 0000012A 890D[00000000]                      mov [max],ecx ; 'max = C'
32                                     ; ----- Преобразование 'max(A,C)' из символа
в число
33                                     check_B:
34                                     mov eax,
34                                     *****
error: invalid combination of opcode and
operands
35 00000130 E867FFFFFF                          call atoi ; Вызов подпрограммы перевода символа в
число
36 00000135 A3[00000000]                      mov [max],eax ; запись преобразованного числа в
'max'
37                                     ; ----- Сравниваем 'max(A,C)' и 'B' (как
числа)
38 0000013A 8B0D[00000000]                      mov ecx,[max]
39 00000140 3B0D[0A000000]                      cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
40 00000146 7F0C                          jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 00000148 8B0D[0A000000]                      mov ecx,[B] ; иначе 'ecx = B'
42 0000014E 890D[00000000]                      mov [max],ecx
43                                     ; ----- Вывод результата
44                                     fin:
45 00000154 B8[13000000]                      mov eax, msg2
46 00000159 E8B1FFFFFF                          call sprint ; Вывод сообщения 'Наибольшее число:
'
47 0000015E A1[00000000]                      mov eax,[max]
48 00000163 E81EFFFFFF                          call iprintLF ; Вывод 'max(A,B,C)'
49 00000168 E86EFFFFFF                          call quit ; Выход
50

```

Рис. 2.11: файл листинга с ошибкой lab8-2

5. Напишите программу нахождения наименьшей из 3 целочисленных переменных a, b и c. Значения переменных выбрать из табл. 8.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу (рис. [2.12],[2.13])

для варианта 1 - 17, 23, 45



```
lab8-3.asm
~/work/study/2022-2023/Архитектура ...pa/study_2022-2023_arh-pc/labs/lab08

mov ecx,B
mov edx,80
call sread
mov eax,B
call atoi
mov [B],eax

mov eax,msgC
call sprint
mov ecx,C
mov edx,80
call sread
mov eax,C
call atoi
mov [C],eax

;-----algorithm-----

mov ecx,[A] ;ecx = A
mov [min],ecx ;min = A

cmp ecx, [B] ; A&B
jnl check_C ; if a<b: goto check_C
mov ecx, [B]
mov [min], ecx ;else min = B

check_C:
cmp ecx, [C]
jnl finish
mov ecx,[C]
mov [min],ecx

finish:
mov eax,answer
```

Рис. 2.12: Файл lab8-3.asm

```
[odedzhimi@fedora lab08]$  
[odedzhimi@fedora lab08]$ nasm -f elf lab8-3.asm  
[odedzhimi@fedora lab08]$ ld -m elf_i386 -o lab8-3 lab8-3.o  
[odedzhimi@fedora lab08]$ ./lab8-3  
Input A: 17  
Input B: 23  
Input C: 45  
Smallest: 17  
[odedzhimi@fedora lab08]$  
[odedzhimi@fedora lab08]$
```

Рис. 2.13: Программа lab8-3.asm

6. Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ выбрать из таблицы 8.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений x и a из 8.6. (рис. [2.14],[2.15])

для варианта 1

$$\begin{cases} 2a - x, x < a \\ 8, x \geq a \end{cases}$$

```

mov [X],eax
;-----algorithm-----

mov ebx, [X]
mov edx, [A]
cmp ebx, edx
jl first
jmp second

first:
mov eax,[A]
mov ebx,2
mul ebx
sub eax,[X]
call iprintLF
call quit
second:
mov eax,8
call iprintLF |
call quit

```

Рис. 2.14: Файл lab8-4.asm

```

[odedzhimi@fedora lab08]$
[odedzhimi@fedora lab08]$ nasm -f elf lab8-4.asm
[odedzhimi@fedora lab08]$ ld -m elf_i386 -o lab8-4 lab8-4.o
[odedzhimi@fedora lab08]$ ./lab8-4
Input A: 2
Input X: 1
3
[odedzhimi@fedora lab08]$ ./lab8-4
Input A: 1
Input X: 2
8
[odedzhimi@fedora lab08]$

```

Рис. 2.15: Программа lab8-4.asm

3 Выводы

Изучили команды условного и безусловного переходов, познакомились с фалом листинга.