

# **Отчёта по лабораторной работе 8**

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

Джозеф Кервенс

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	22

## 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: . . . . .	10
2.5	Файл lab8-1.asm . . . . .	11
2.6	Программа lab8-1.asm . . . . .	12
2.7	Файл lab8-2.asm . . . . .	13
2.8	Программа lab8-2.asm . . . . .	14
2.9	Файл листинга lab8-2 . . . . .	15
2.10	ошибка трансляции lab8-2 . . . . .	16
2.11	файл листинга с ошибкой lab8-2 . . . . .	17
2.12	Файл lab8-3.asm . . . . .	18
2.13	Программа lab8-3.asm . . . . .	19
2.14	Файл lab8-4.asm . . . . .	20
2.15	Программа lab8-4.asm . . . . .	21

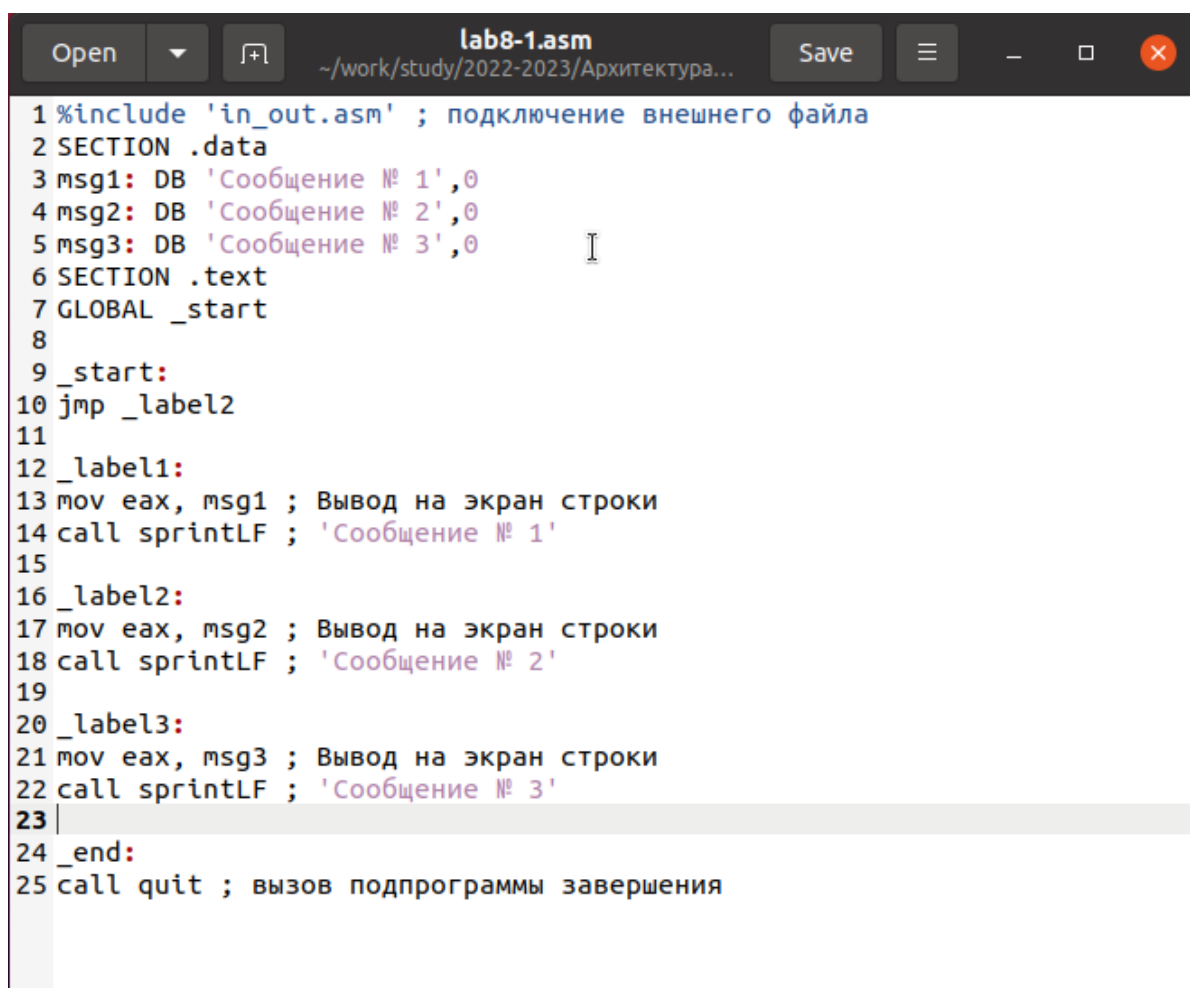
## **Список таблиц**

# 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

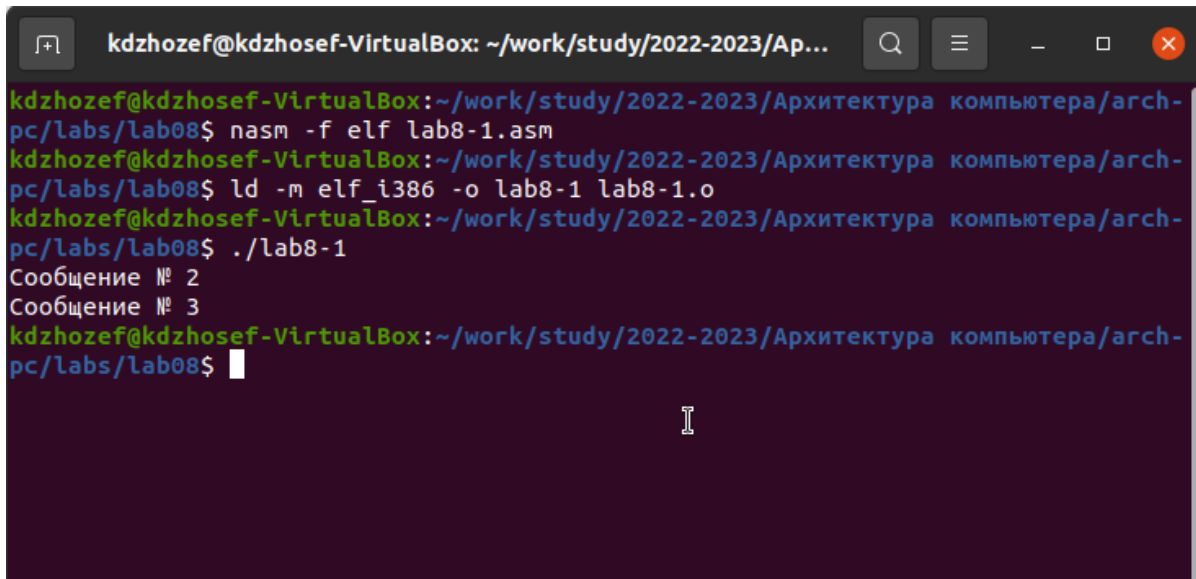
1. Создайте каталог для программ лабораторной работы № 8, перейдите в него и создайте файл lab8-1.asm
2. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции jmp. Введите в файл lab8-1.asm текст программы из листинга 8.1. (рис.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



```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8
9 _start:
10 jmp _label2
11
12 _label1:
13 mov eax, msg1 ; Вывод на экран строки
14 call sprintLF ; 'Сообщение № 1'
15
16 _label2:
17 mov eax, msg2 ; Вывод на экран строки
18 call sprintLF ; 'Сообщение № 2'
19
20 _label3:
21 mov eax, msg3 ; Вывод на экран строки
22 call sprintLF ; 'Сообщение № 3'
23
24 _end:
25 call quit ; вызов подпрограммы завершения
```

Рис. 2.1: Файл lab8-1.asm:

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

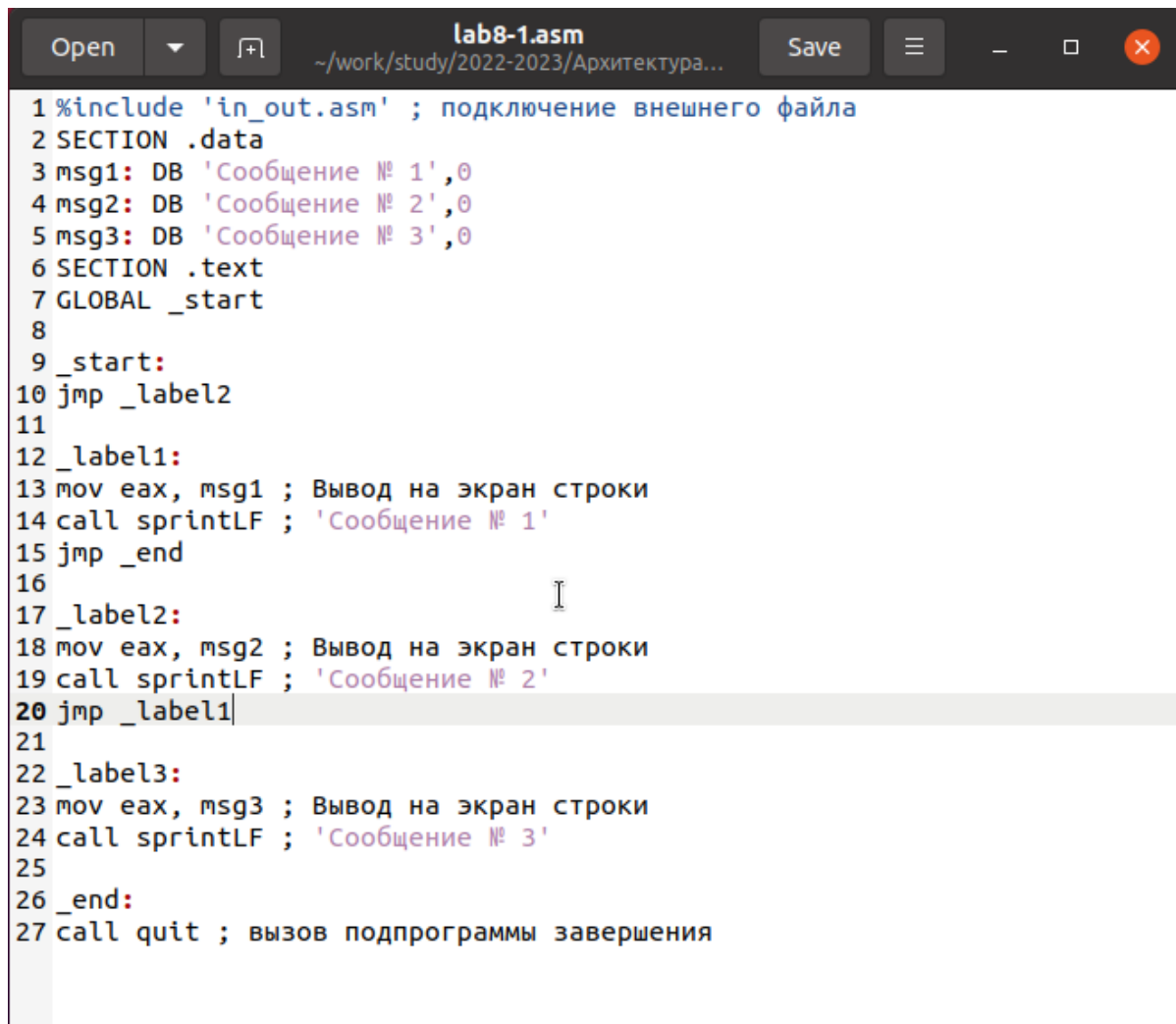


```
kdzhosef@kdzhosef-VirtualBox: ~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-1.asm  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-1  
Сообщение № 2  
Сообщение № 3  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$
```

Рис. 2.2: Программа lab8-1.asm:

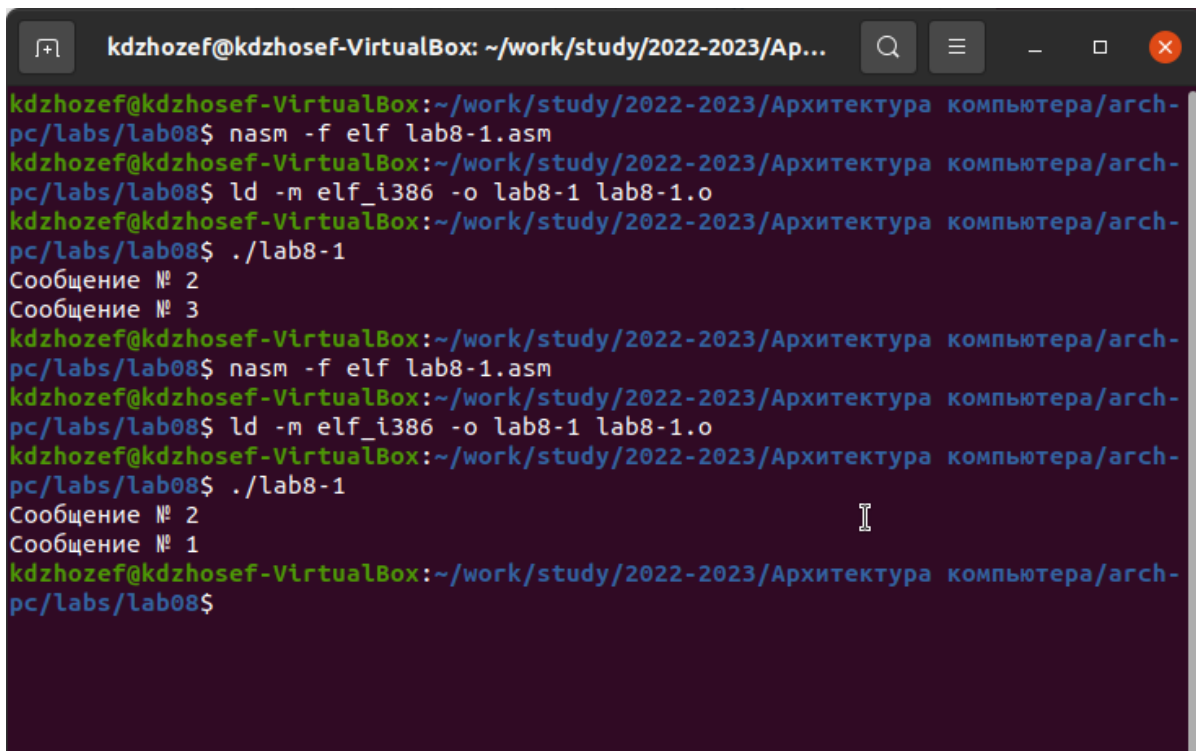
Инструкция `jmp` позволяет осуществлять переходы не только вперед но и назад. Изменим программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию `jmp` с меткой `_label1` (т.е. 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)





```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8
9 _start:
10 jmp _label2
11
12 _label1:
13 mov eax, msg1 ; Вывод на экран строки
14 call sprintf ; 'Сообщение № 1'
15 jmp _end
16
17 _label2:
18 mov eax, msg2 ; Вывод на экран строки
19 call sprintf ; 'Сообщение № 2'
20 jmp _label1
21
22 _label3:
23 mov eax, msg3 ; Вывод на экран строки
24 call sprintf ; 'Сообщение № 3'
25
26 _end:
27 call quit ; вызов подпрограммы завершения
```

Рис. 2.3: Файл lab8-1.asm:



```
kdzhosef@kdzhosef-VirtualBox: ~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-1.asm  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-1  
Сообщение № 2  
Сообщение № 3  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-1.asm  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-1  
Сообщение № 2  
Сообщение № 1  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$
```

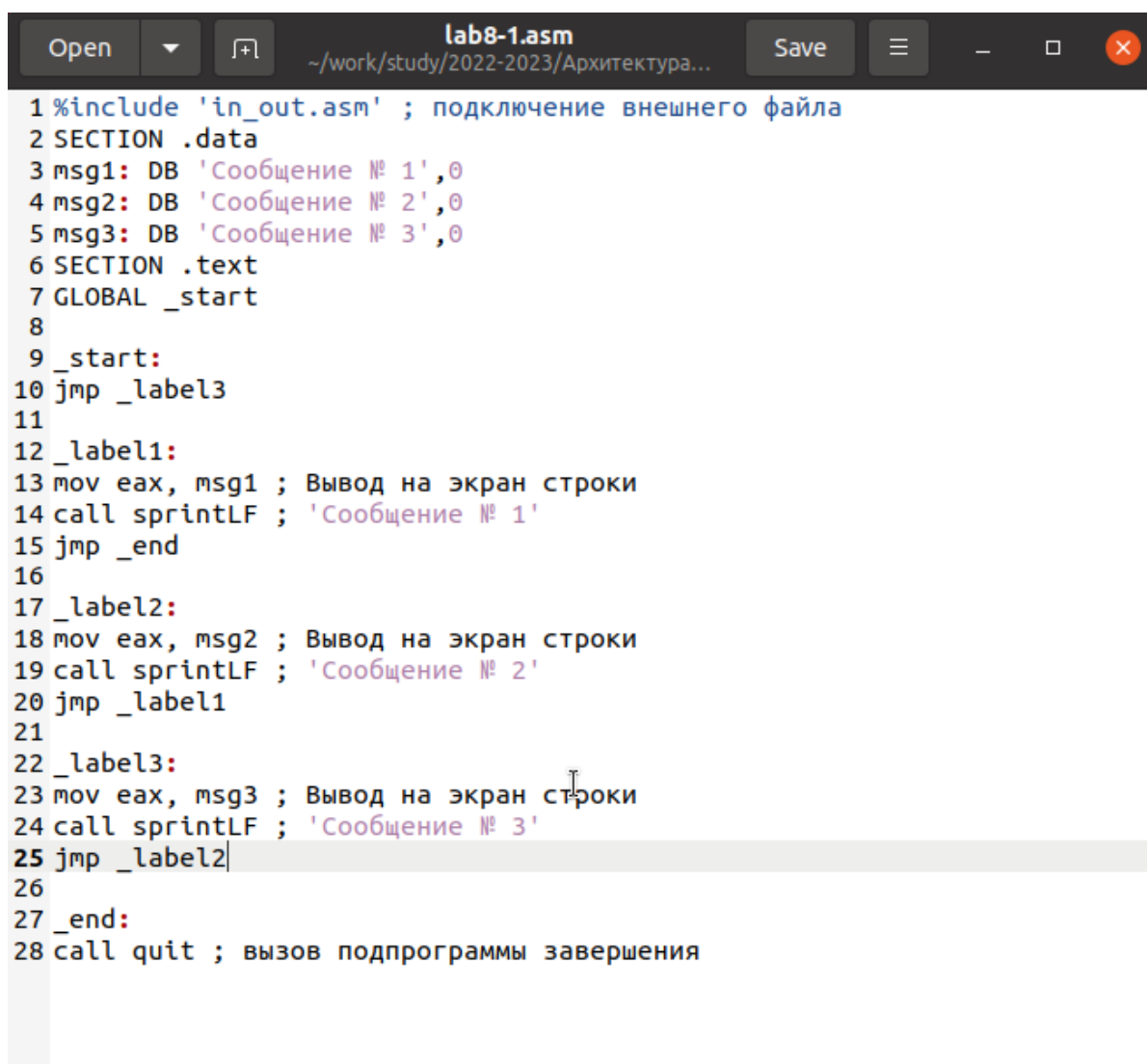
Рис. 2.4: Программа lab8-1.asm:

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

Сообщение № 3

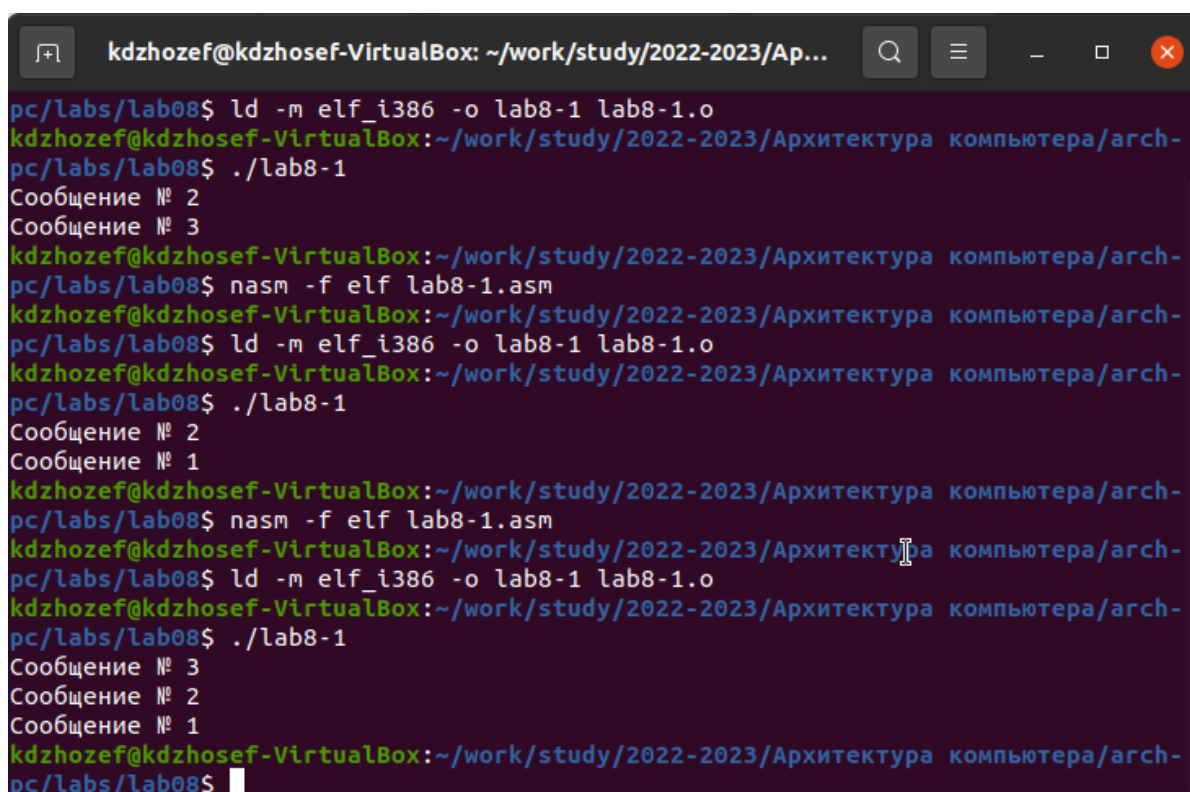
Сообщение № 2

Сообщение № 1



```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 msg1: DB 'Сообщение № 1',0
4 msg2: DB 'Сообщение № 2',0
5 msg3: DB 'Сообщение № 3',0
6 SECTION .text
7 GLOBAL _start
8
9 _start:
10 jmp _label3
11
12 _label1:
13 mov eax, msg1 ; Вывод на экран строки
14 call sprintf ; 'Сообщение № 1'
15 jmp _end
16
17 _label2:
18 mov eax, msg2 ; Вывод на экран строки
19 call sprintf ; 'Сообщение № 2'
20 jmp _label1
21
22 _label3:
23 mov eax, msg3 ; Вывод на экран строки
24 call sprintf ; 'Сообщение № 3'
25 jmp _label2
26
27 _end:
28 call quit ; вызов подпрограммы завершения
```

Рис. 2.5: Файл lab8-1.asm



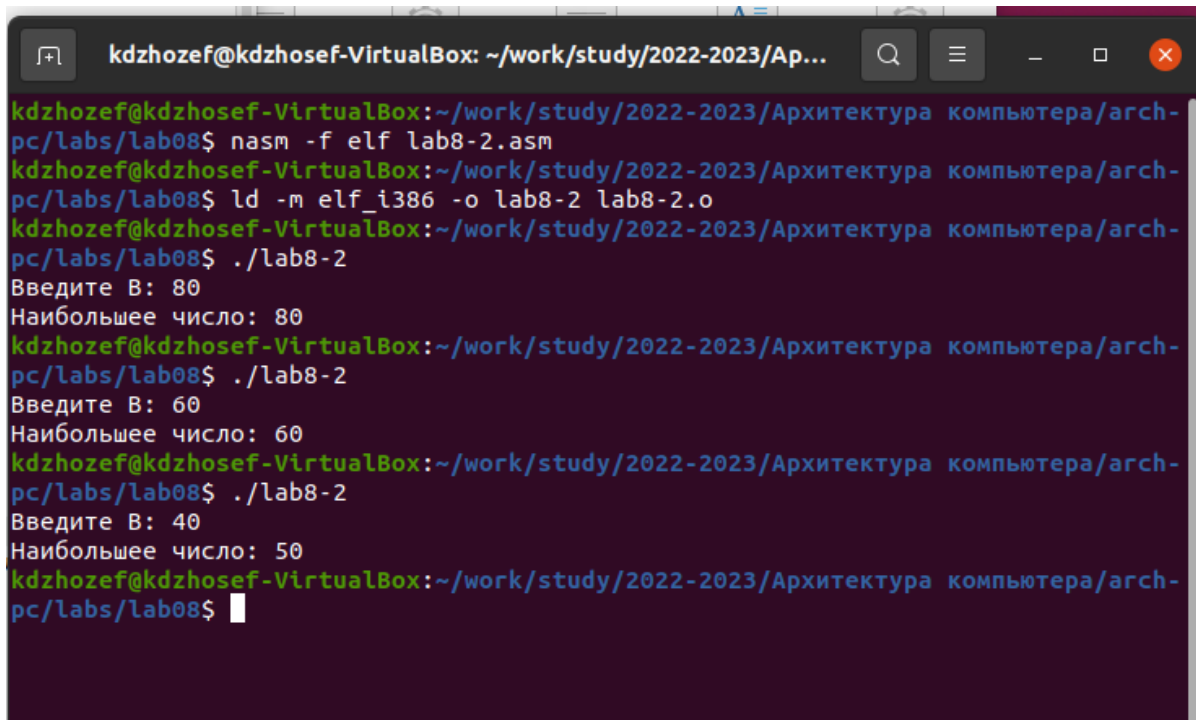
```
kdzhosef@kdzhosef-VirtualBox: ~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-1  
Сообщение № 2  
Сообщение № 3  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-1.asm  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-1  
Сообщение № 2  
Сообщение № 1  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-1.asm  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-1  
Сообщение № 3  
Сообщение № 2  
Сообщение № 1  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/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)

```
1 %include 'in_out.asm'
2 section .data
3 msg1 db 'Введите B: ',0h
4 msg2 db "Наибольшее число: ",0h
5 A dd '20'
6 C dd '50'
7 section .bss
8 max resb 10
9 B resb 10
10 section .text
11 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 mov eax,msg1
15 call sprint
16 ; ----- Ввод 'B'
17 mov ecx,B
18 mov edx,10
19 call sread
20 ; ----- Преобразование 'B' из символа в число
21 mov eax,B
22 call atoi ; Вызов подпрограммы перевода символа в число
23 mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
25 mov ecx,[A] ; 'ecx = A'
26 mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C] ; Сравниваем 'A' и 'C'
29 jg check_B ; если 'A>C', то переход на метку 'check_B',
30 mov ecx,[C] ; иначе 'ecx = C'
31 mov [max],ecx ; 'max = C'
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 mov eax,max
35 call atoi ; Вызов подпрограммы перевода символа в число
36 mov [max],eax ; запись преобразованного числа в 'max'
```

Рис. 2.7: Файл lab8-2.asm



```
kdzhofef@kdzhofef-VirtualBox: ~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-2.asm  
kdzhofef@kdzhofef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-2 lab8-2.o  
kdzhofef@kdzhofef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-2  
Введите B: 80  
Наибольшее число: 80  
kdzhofef@kdzhofef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-2  
Введите B: 60  
Наибольшее число: 60  
kdzhofef@kdzhofef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-2  
Введите B: 40  
Наибольшее число: 50  
kdzhofef@kdzhofef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/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)

```

192 17 000000F2 B9[0A000000]    mov ecx,B
193 18 000000F7 BA0A000000    mov edx,10
194 19 000000FC E842FFFFFF    call sread
195 20                                ; ----- Преобразование 'B' из символа в число
196 21 00000101 B8[0A000000]    mov eax,B
197 22 00000106 E891FFFFFF    call atoi ; Вызов подпрограммы перевода символа в число
198 23 0000010B A3[0A000000]    mov [B],eax ; запись преобразованного числа в 'B'
199 24                                ; ----- Записываем 'A' в переменную 'max'
200 25 00000110 8B0D[35000000]    mov ecx,[A] ; 'ecx = A'
201 26 00000116 890D[00000000]    mov [max],ecx ; 'max = A'
202 27                                ; ----- Сравниваем 'A' и 'C' (как символы)
203 28 0000011C 3B0D[39000000]    cmp ecx,[C] ; Сравниваем 'A' и 'C'
204 29 00000122 7F0C                                jg check_B ; если 'A>C', то переход на метку 'check_B',
205 30 00000124 8B0D[39000000]    mov ecx,[C] ; иначе 'ecx = C'
206 31 0000012A 890D[00000000]    mov [max],ecx ; 'max = C'
207 32                                ; ----- Преобразование 'max(A,C)' из символа в число
208 33                                check_B:
209 34 00000130 B8[00000000]    mov eax,max
210 35 00000135 E862FFFFFF    call atoi ; Вызов подпрограммы перевода символа в число
211 36 0000013A A3[00000000]    mov [max],eax ; запись преобразованного числа в 'max'
212 37                                ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
213 38 0000013F 8B0D[00000000]    mov ecx,[max]
214 39 00000145 3B0D[0A000000]    cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
215 40 0000014B 7F0C                                jg fin ; если 'max(A,C)>B', то переход на 'fin',
216 41 0000014D 8B0D[0A000000]    mov ecx,[B] ; иначе 'ecx = B'
217 42 00000153 890D[00000000]    mov [max],ecx
218 43                                ; ----- Вывод результата
219 44                                fin:
220 45 00000159 B8[13000000]    mov eax,msg2
221 46 0000015E E8ACFFFFFF    call sprintf ; Вывод сообщения 'Наибольшее число: '
222 47 00000163 A1[00000000]    mov eax,[max]
223 48 00000168 E819FFFFFF    call iprintLF ; Вывод 'max(A,B,C)'
224 49 0000016D E869FFFFFF    call quit ; Выход
225 50

```

Рис. 2.9: Файл листинга lab8-2

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

строка 10

- 10 - номер строки
- 00000006 - адрес
- 7403 - машинный код
- jz finished - код программы

строка 11

- 11 - номер строки
- 00000008 - адрес

- 40 - машинный код
- inc eax - код программы

строка 12

- 12 - номер строки
- 00000009 - адрес
- EBF8 - машинный код
- jmp nextchar - 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)

```
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-
pc/labs/lab08$ nasm -f elf lab8-2.asm -l lab8-2.lst
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-
pc/labs/lab08$ nasm -f elf lab8-2.asm -l lab8-2.lst
lab8-2.asm:23: error: invalid combination of opcode and operands
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-
pc/labs/lab08$
```

Рис. 2.10: ошибка трансляции lab8-2



```

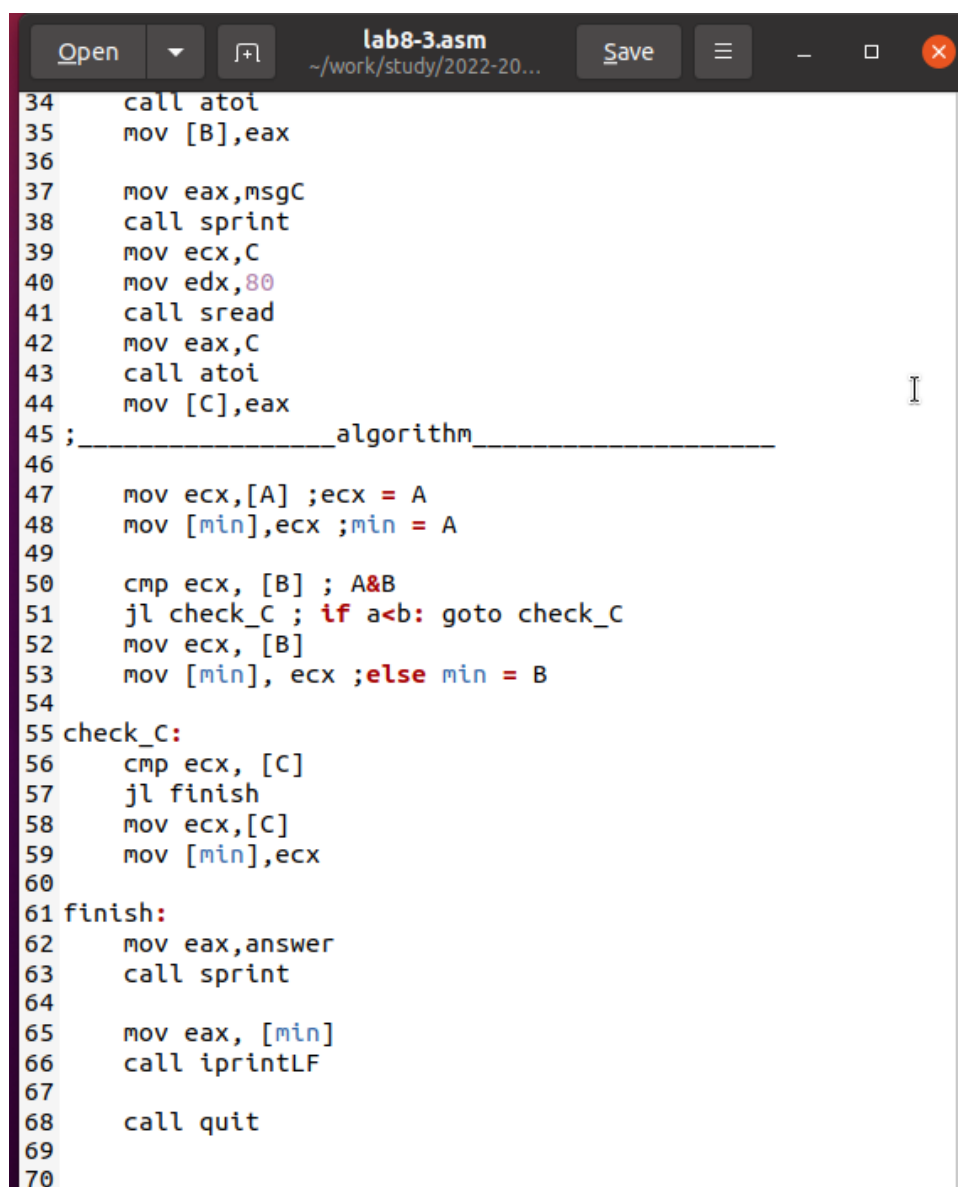
lab8-2.asm
193 18 000000F7 BA0A000000    mov edx,10
194 19 000000FC E842FFFFFF    call sread
195 20                                ; ----- Преобразование 'B' из символа в число
196 21 00000101 B8[0A000000]    mov eax,B
197 22 00000106 E891FFFFFF    call atoi ; Вызов подпрограммы перевода символа в число
198 23                                mov [B], ; запись преобразованного числа в 'B'
199 23                                error: invalid combination of opcode and operands
200 24                                ; ----- Записываем 'A' в переменную 'max'
201 25 0000010B 8B0D[35000000]    mov ecx,[A] ; 'ecx = A'
202 26 00000111 890D[00000000]    mov [max],ecx ; 'max = A'
203 27                                ; ----- Сравниваем 'A' и 'C' (как символы)
204 28 00000117 3B0D[39000000]    cmp ecx,[C] ; Сравниваем 'A' и 'C'
205 29 0000011D 7F0C                                jg check_B ; если 'A>C', то переход на метку 'check_B',
206 30 0000011F 8B0D[39000000]    mov ecx,[C] ; иначе 'ecx = C'
207 31 00000125 890D[00000000]    mov [max],ecx ; 'max = C'
208 32                                ; ----- Преобразование 'max(A,C)' из символа в число
209 33                                check_B:
210 34 0000012B B8[00000000]    mov eax,max
211 35 00000130 E867FFFFFF    call atoi ; Вызов подпрограммы перевода символа в число
212 36 00000135 A3[00000000]    mov [max],eax ; запись преобразованного числа в 'max'
213 37                                ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
214 38 0000013A 8B0D[00000000]    mov ecx,[max]
215 39 00000140 3B0D[0A000000]    cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
216 40 00000146 7F0C                                jg fin ; если 'max(A,C)>B', то переход на 'fin',
217 41 00000148 8B0D[0A000000]    mov ecx,[B] ; иначе 'ecx = B'
218 42 0000014E 890D[00000000]    mov [max],ecx
219 43                                ; ----- Вывод результата
220 44                                fin:
221 45 00000154 B8[13000000]    mov eax,msg2
222 46 00000159 E8B1FEFFFF    call sprint ; Вывод сообщения 'Наибольшее число: '
223 47 0000015E A1[00000000]    mov eax,[max]
224 48 00000163 E81EFFFFFF    call iprintLF ; Вывод 'max(A,B,C)'
225 49 00000168 E86EFFFFFF    call quit ; Выход
226 50

```

Рис. 2.11: файл листинга с ошибкой lab8-2

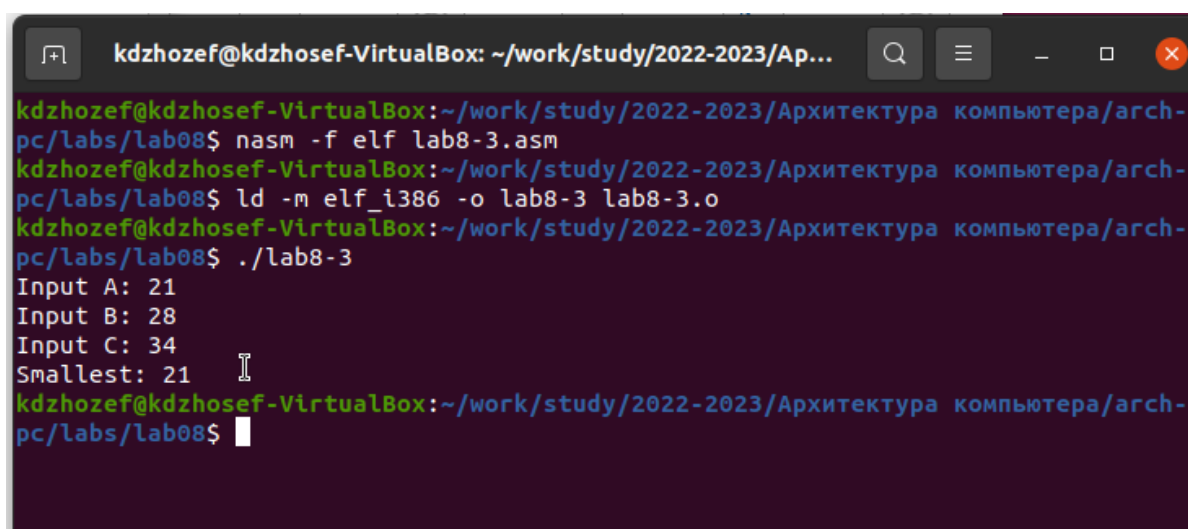
5. Напишите программу нахождения наименьшей из 3 целочисленных переменных a,b и c. Значения переменных выбрать из табл. 8.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу (рис. 2.12,2.13)

для варианта 11 - 21, 28, 34



```
34    call atoi
35    mov [B],eax
36
37    mov eax,msgC
38    call sprint
39    mov ecx,C
40    mov edx,80
41    call sread
42    mov eax,C
43    call atoi
44    mov [C],eax
45 ; _____algorithm_____
46
47    mov ecx,[A] ;ecx = A
48    mov [min],ecx ;min = A
49
50    cmp ecx, [B] ; A&B
51    jl check_C ; if a<b: goto check_C
52    mov ecx, [B]
53    mov [min], ecx ;else min = B
54
55 check_C:
56    cmp ecx, [C]
57    jl finish
58    mov ecx,[C]
59    mov [min],ecx
60
61 finish:
62    mov eax,answer
63    call sprint
64
65    mov eax, [min]
66    call iprintLF
67
68    call quit
69
70
```

Рис. 2.12: Файл lab8-3.asm



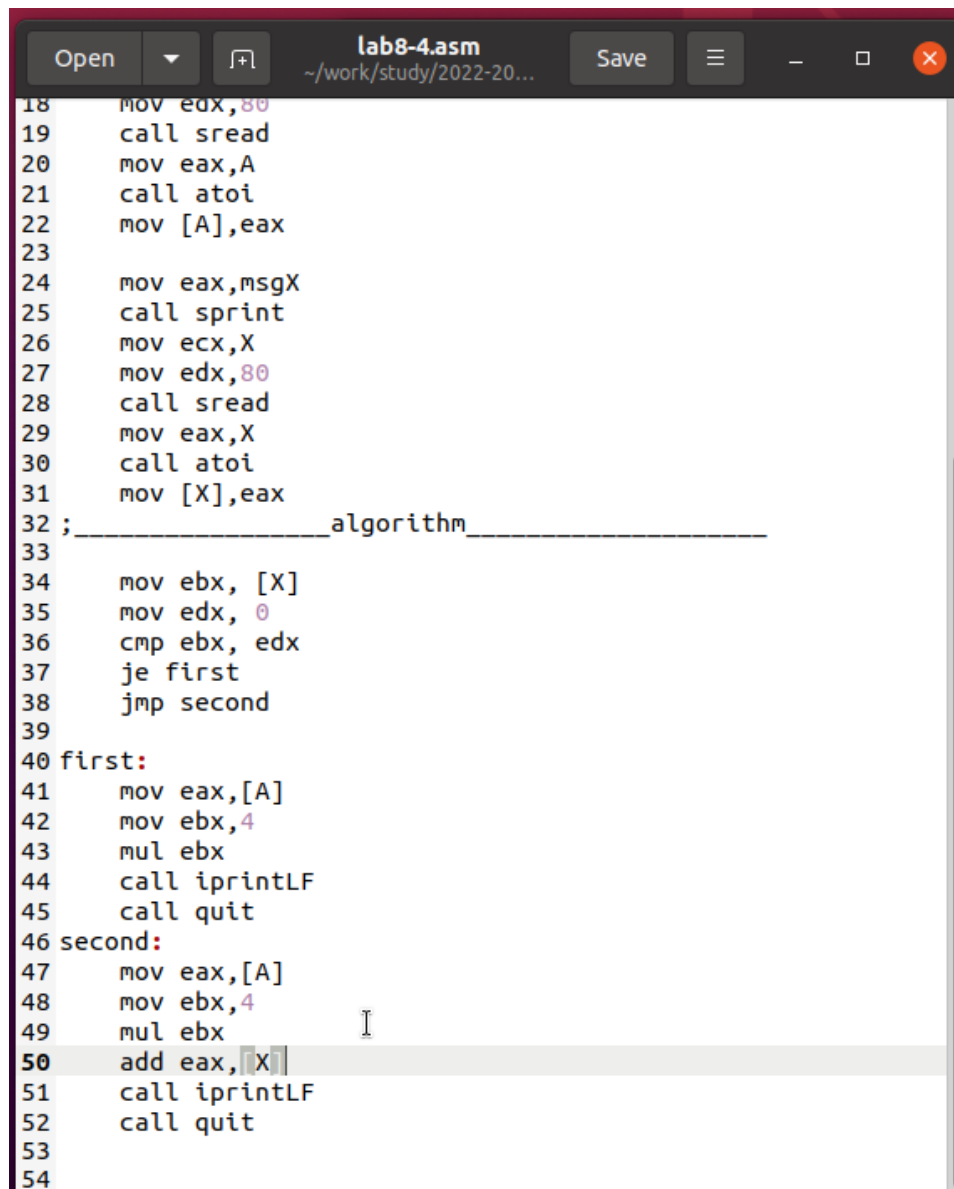
```
kdzhozef@kdzhosef-VirtualBox: ~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-3.asm  
kdzhhozef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o  
kdzhhozef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-3  
Input A: 21  
Input B: 28  
Input C: 34  
Smallest: 21  
kdzhhozef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/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 11

$$\begin{cases} 4a, x = a \\ 4a + x, x \neq a \end{cases}$$



```
lab8-4.asm
~/work/study/2022-20...
Open Save

18  mov edx,80
19  call sread
20  mov eax,A
21  call atoi
22  mov [A],eax
23
24  mov eax,msgX
25  call sprint
26  mov ecx,X
27  mov edx,80
28  call sread
29  mov eax,X
30  call atoi
31  mov [X],eax
32 ; ----- algorithm -----
33
34  mov ebx, [X]
35  mov edx, 0
36  cmp ebx, edx
37  je first
38  jmp second
39
40 first:
41  mov eax,[A]
42  mov ebx,4
43  mul ebx
44  call iprintLF
45  call quit
46 second:
47  mov eax,[A]
48  mov ebx,4
49  mul ebx
50  add eax,[X]
51  call iprintLF
52  call quit
53
54
```

Рис. 2.14: Файл lab8-4.asm

```
kdzhosef@kdzhosef-VirtualBox: ~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ nasm -f elf lab8-4.asm  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-4  
Input A: 3  
Input X: 0  
12  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$ ./lab8-4  
Input A: 2  
Input X: 1  
9  
kdzhosef@kdzhosef-VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-  
pc/labs/lab08$
```

Рис. 2.15: Программа lab8-4.asm

## 3 Выводы

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