

# **Отчёта по лабораторной работе 8**

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

Касканте Родригес Альберто

# 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

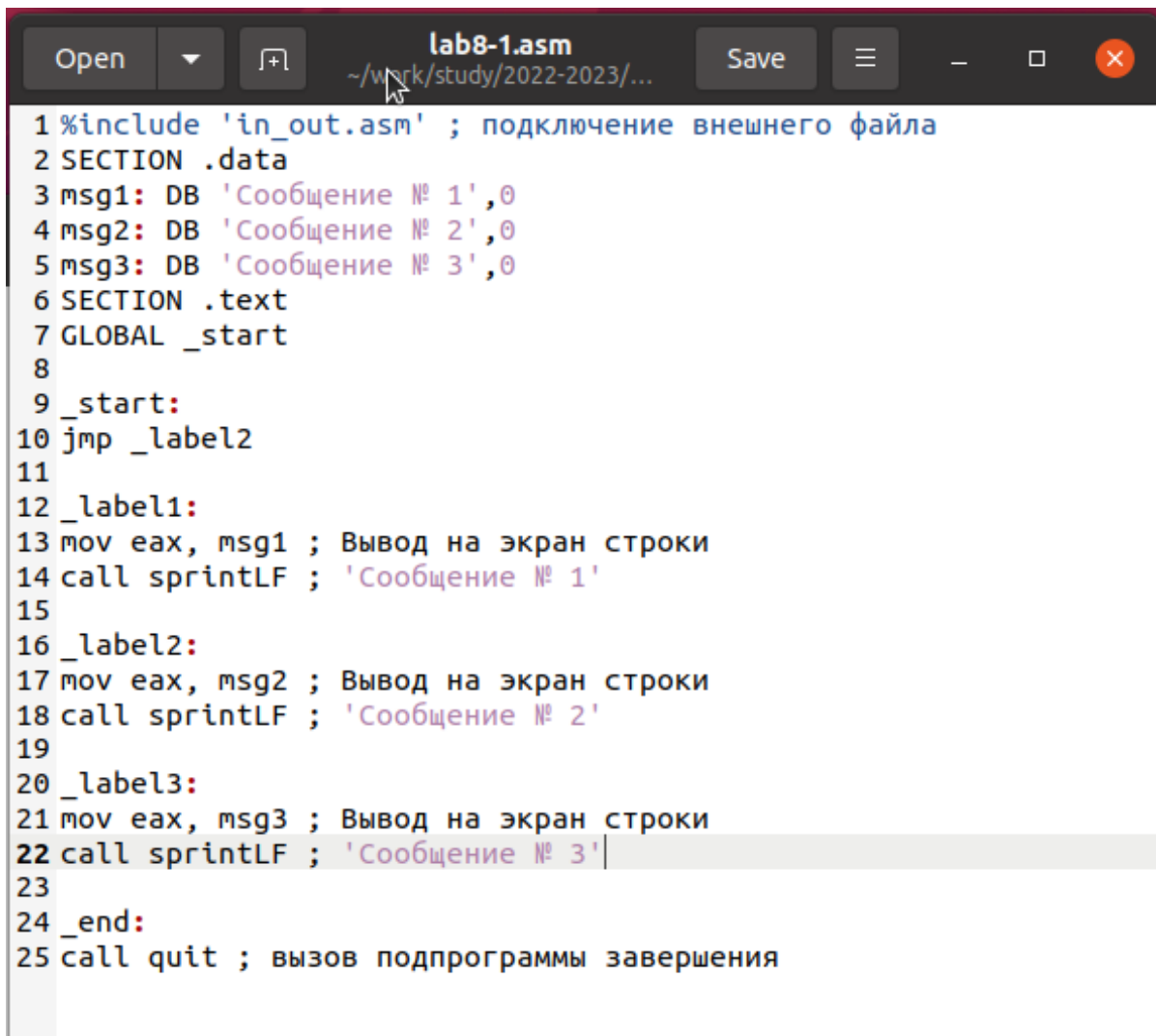
## Список таблиц

# 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

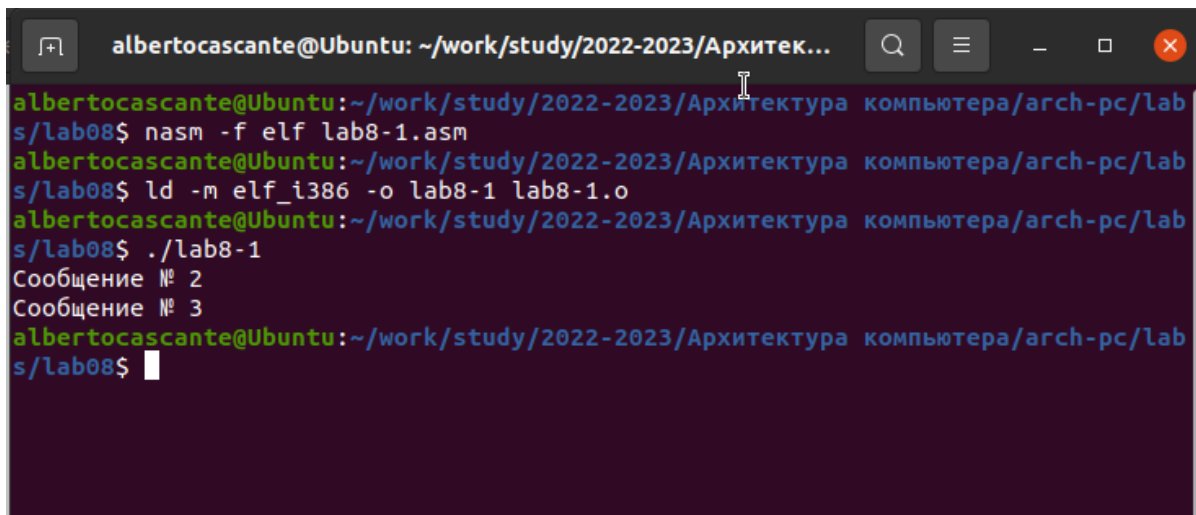
1. Создайте каталог для программ лабораторной работы № 8, перейдите в него и создайте файл lab8-1.asm
2. Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции jmp. Введите в файл lab8-1.asm текст программы из листинга 8.1. (рис.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

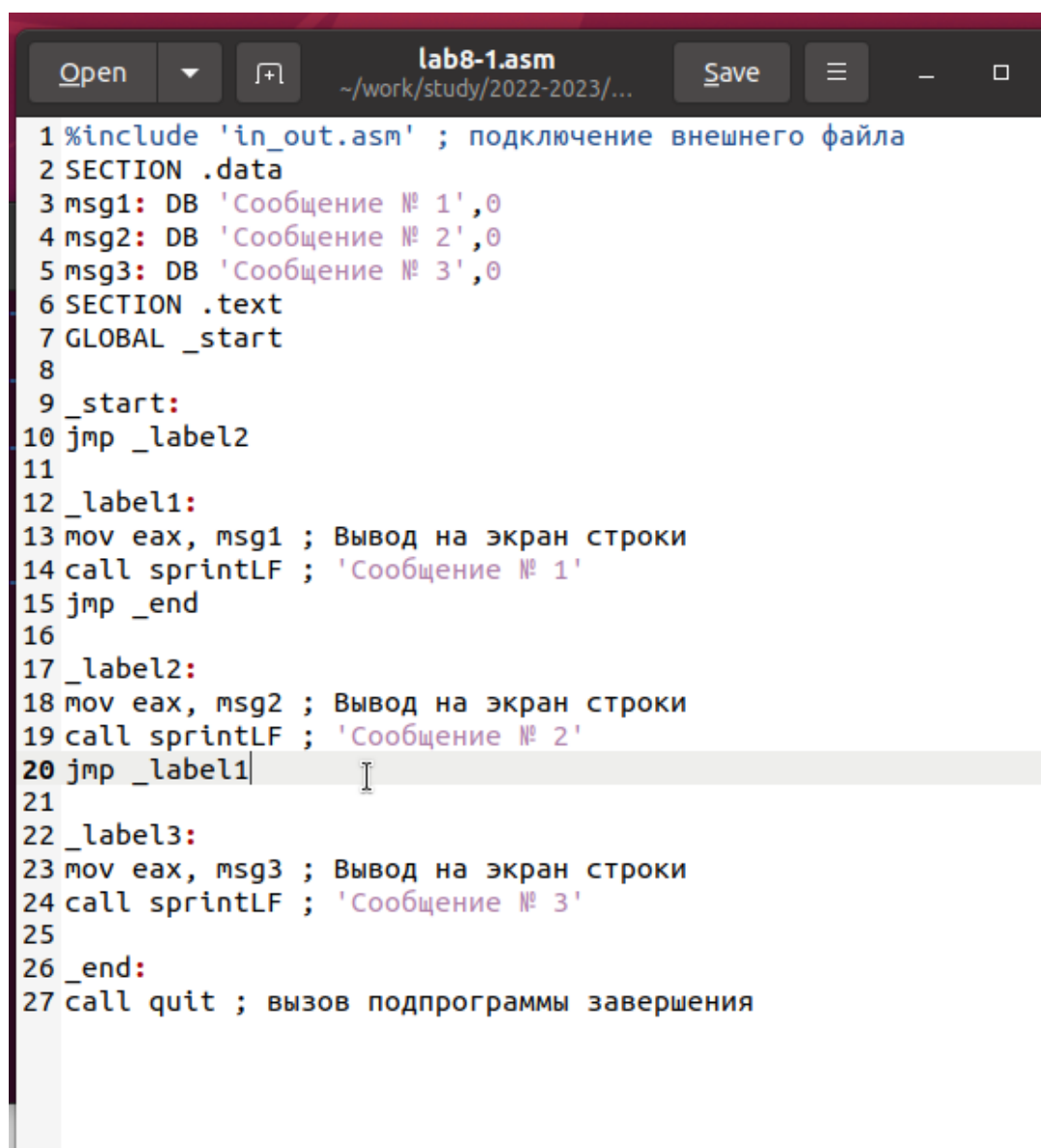
A terminal window with a dark background and light-colored text. The window title is 'albertocascante@Ubuntu: ~/work/study/2022-2023/Архитек...'. The terminal shows the following commands and output:

```
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08$ nasm -f elf lab8-1.asm
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08$ ./lab8-1
Сообщение № 2
Сообщение № 3
albertocascante@Ubuntu:~/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 sprintfLF ; 'Сообщение № 1'
15 jmp _end
16
17 _label2:
18 mov eax, msg2 ; Вывод на экран строки
19 call sprintfLF ; 'Сообщение № 2'
20 jmp _label1
21
22 _label3:
23 mov eax, msg3 ; Вывод на экран строки
24 call sprintfLF ; 'Сообщение № 3'
25
26 _end:
27 call quit ; вызов подпрограммы завершения
```

Рис. 2.3: Файл lab8-1.asm:

```
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/а  
s/lab08$ nasm -f elf lab8-1.asm  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/а  
s/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/а  
s/lab08$ ./lab8-1  
Сообщение № 2  
Сообщение № 1  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/а  
s/lab08$
```

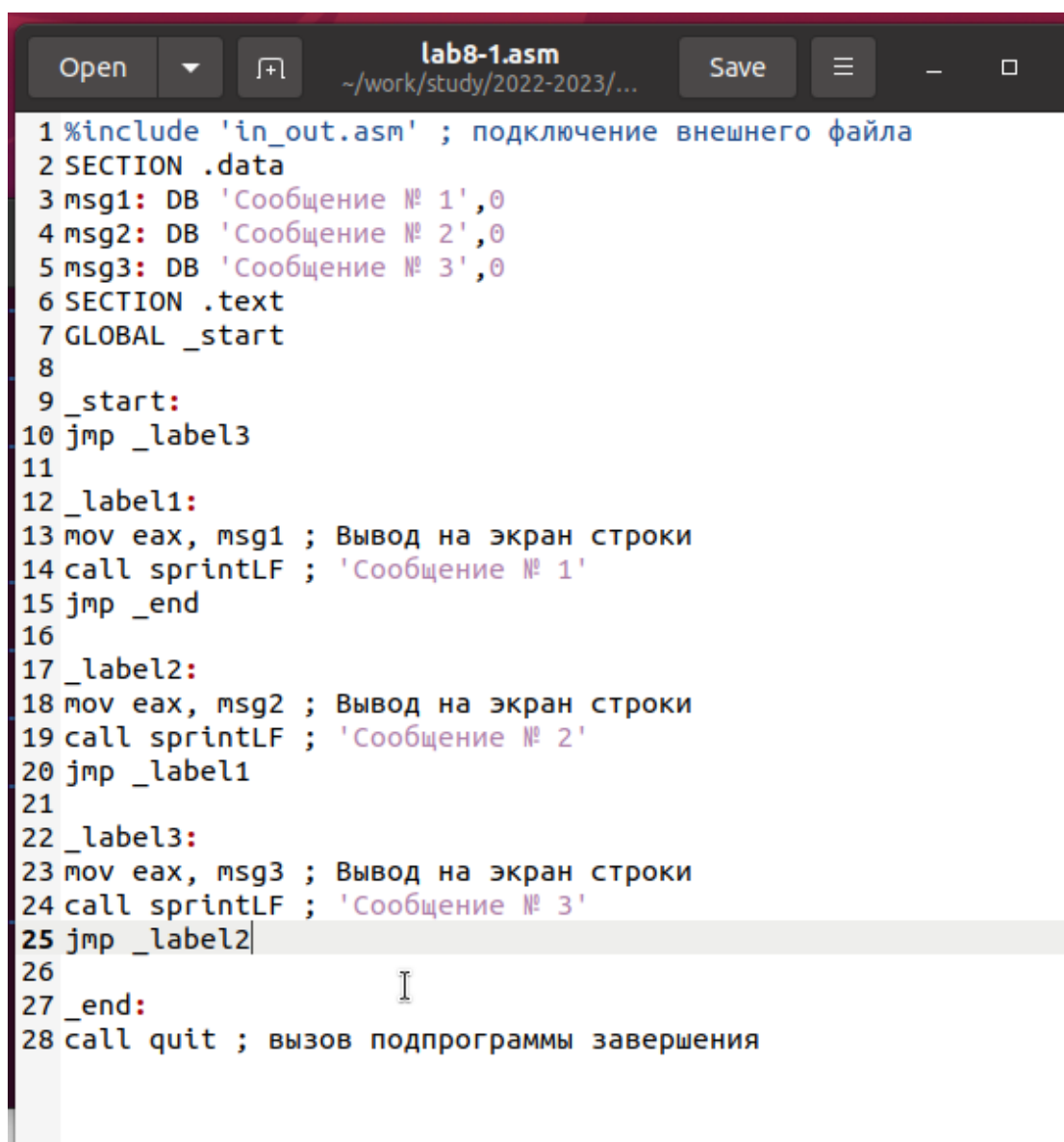
Рис. 2.4: Программа lab8-1.asm:

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 sprintfLF ; 'Сообщение № 1'
15 jmp _end
16
17 _label2:
18 mov eax, msg2 ; Вывод на экран строки
19 call sprintfLF ; 'Сообщение № 2'
20 jmp _label1
21
22 _label3:
23 mov eax, msg3 ; Вывод на экран строки
24 call sprintfLF ; 'Сообщение № 3'
25 jmp _label2
26
27 _end:
28 call quit ; вызов подпрограммы завершения
```

Рис. 2.5: Файл lab8-1.asm

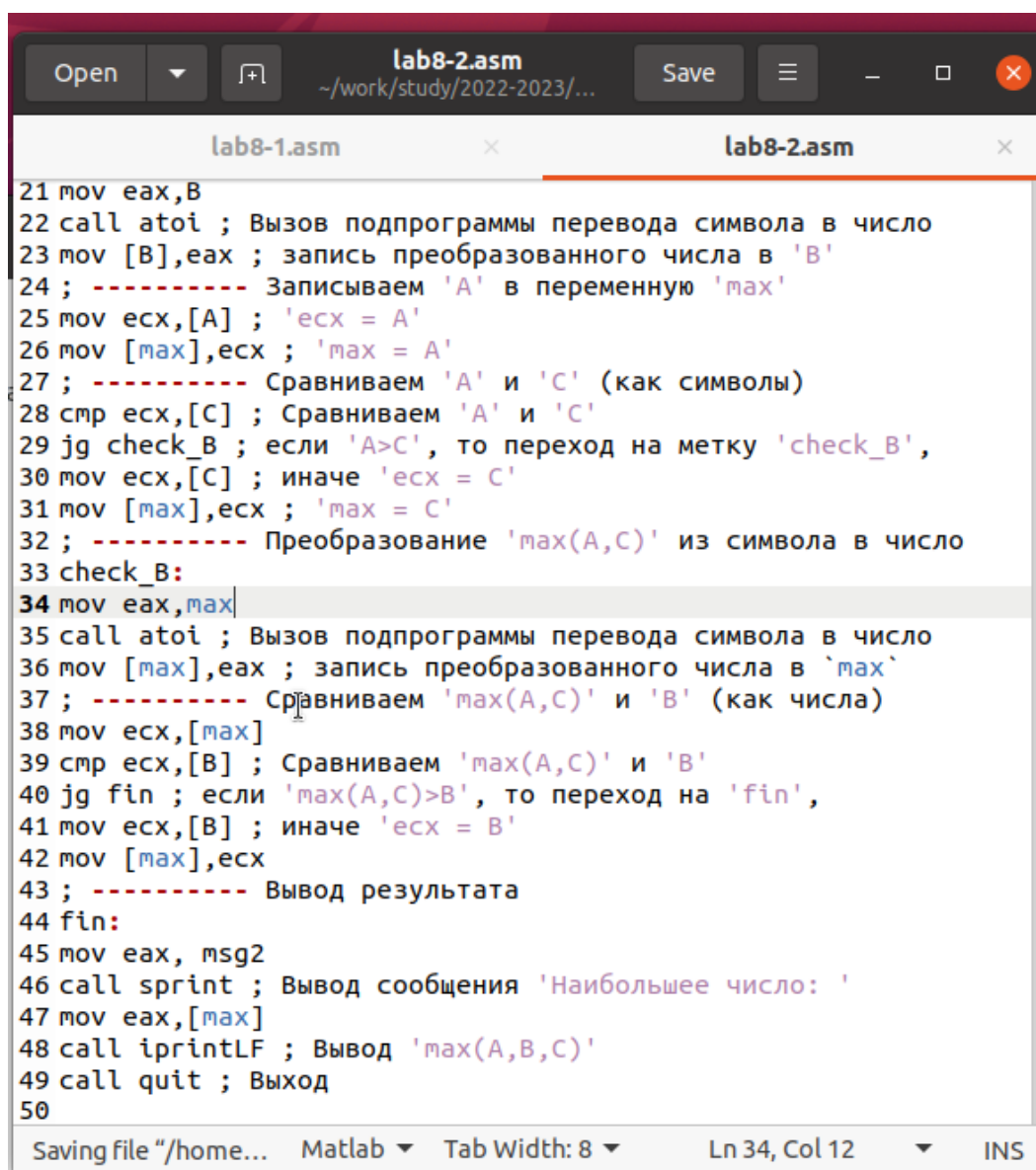
```

albertocascante@Ubuntu:~/work/study/2022-2023/Архитекту
s/lab08$ nasm -f elf lab8-1.asm
albertocascante@Ubuntu:~/work/study/2022-2023/Архитекту
s/lab08$ ld -m elf_i386 -o lab8-1 lab8-1.o
albertocascante@Ubuntu:~/work/study/2022-2023/Архитекту
s/lab08$ ./lab8-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
albertocascante@Ubuntu:~/work/study/2022-2023/Архитекту
s/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])



```
lab8-2.asm
~/work/study/2022-2023/...
Save

lab8-1.asm x lab8-2.asm x
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'
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 mov ecx,[max]
39 cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'
40 jg fin ; если 'max(A,C)>B', то переход на 'fin',
41 mov ecx,[B] ; иначе 'ecx = B'
42 mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 mov eax, msg2
46 call sprintf ; Вывод сообщения 'Наибольшее число: '
47 mov eax,[max]
48 call sprintf ; Вывод 'max(A,B,C)'
49 call quit ; Выход
50

Saving file "/home... Matlab Tab Width: 8 Ln 34, Col 12 INS
```

Рис. 2.7: Файл lab8-2.asm

```
albertocascante@Ubuntu:~/work/study/2022-2023/Архитек
s/lab08$ ./lab8-2
Введите В: 200
Наибольшее число: 200
albertocascante@Ubuntu:~/work/study/2022-2023/Архитек
s/lab08$ ./lab8-2
Введите В: 100
Наибольшее число: 100
albertocascante@Ubuntu:~/work/study/2022-2023/Архитек
s/lab08$ ./lab8-2
Введите В: 20
Наибольшее число: 50
albertocascante@Ubuntu:~/work/study/2022-2023/Архитек
s/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])

```
lab8-2.lst
~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab08

lab8-1.asm lab8-2.asm lab8-2.lst
15 14 00000000 2000 <1> sub     ecx, ecx
16 15 0000000D 5B <1> pop     ebx
17 16 0000000E C3 <1> ret
18 17 <1>
19 18 <1>
20 19 <1> ;----- sprint -----
21 20 <1> ; функция печати сообщения
22 21 <1> ; входные данные: mov eax,<message>
23 22 <1> sprint:
24 23 0000000F 52 <1> push    edx
25 24 00000010 51 <1> push    ecx
26 25 00000011 53 <1> push    ebx
27 26 00000012 50 <1> push    eax
28 27 00000013 E8E8FFFFFF <1> call    slen
29 28 <1>
30 29 00000018 89C2 <1> mov     edx, eax
31 30 0000001A 58 <1> pop     eax
32 31 <1>
33 32 0000001B 89C1 <1> mov     ecx, eax
34 33 0000001D B801000000 <1> mov     ebx, 1
35 34 00000022 B804000000 <1> mov     eax, 4
36 35 00000027 CD80 <1> int     80h
37 36 <1>
38 37 00000029 5B <1> pop     ebx
39 38 0000002A 59 <1> pop     ecx
40 39 0000002B 5A <1> pop     edx
41 40 0000002C C3 <1> ret
42 41 <1>
43 42 <1>
44 43 <1> ;----- sprintLF -----
Plain Text Tab Width: 8 Ln 1, Col 1 INS
```

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

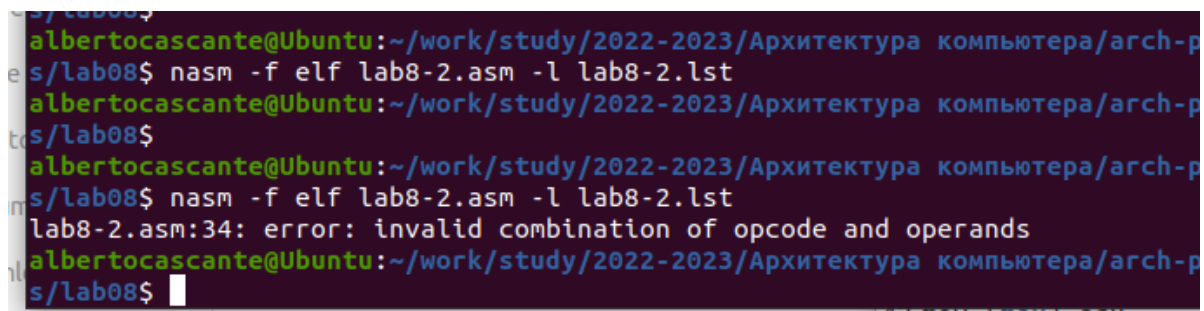
- 52 - номер строки
- 00000038 - адрес

- 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])



```
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/arch-p
s/lab08$ nasm -f elf lab8-2.asm -l lab8-2.lst
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/arch-p
s/lab08$
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/arch-p
s/lab08$ nasm -f elf lab8-2.asm -l lab8-2.lst
lab8-2.asm:34: error: invalid combination of opcode and operands
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура компьютера/arch-p
s/lab08$
```

Рис. 2.10: ошибка трансляции lab8-2



```

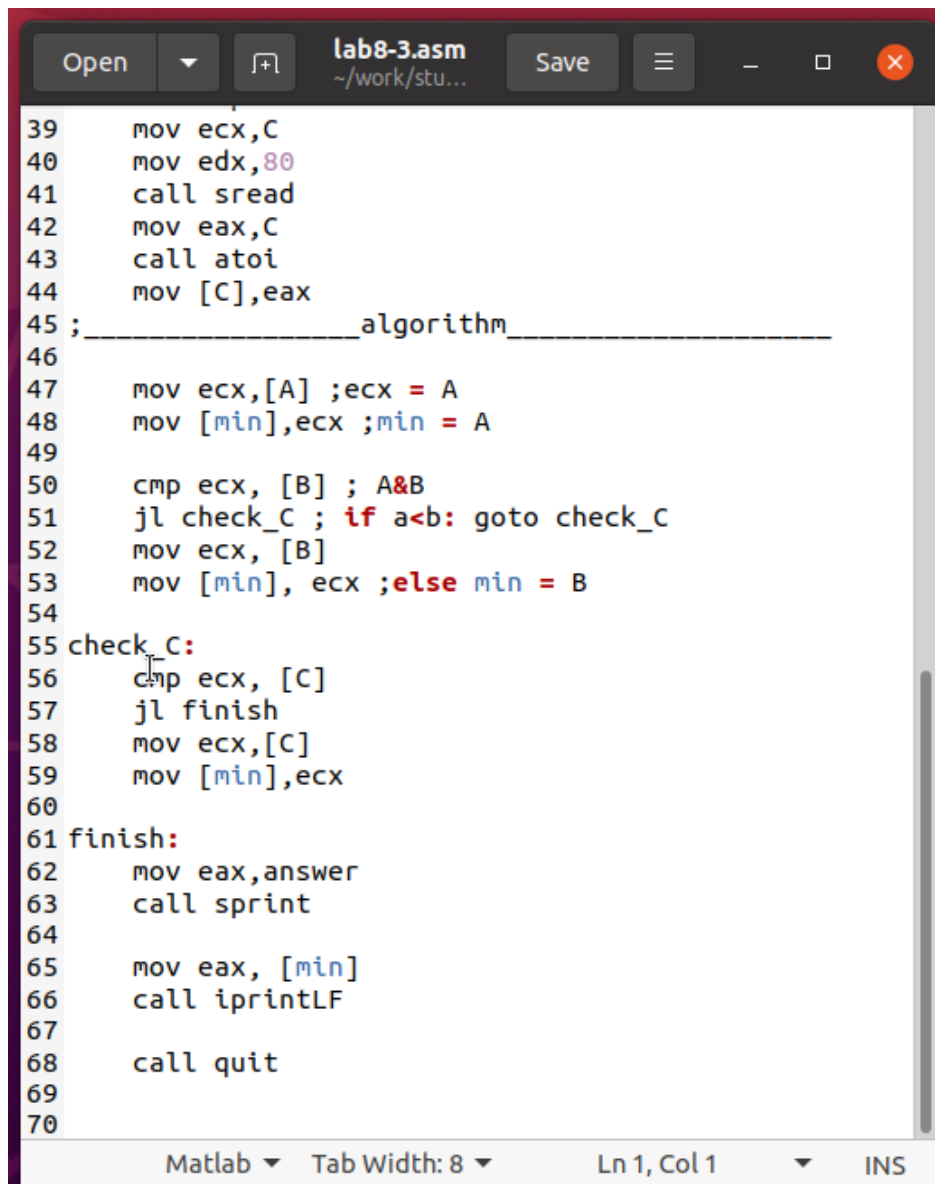
lab8-2.asm
22 00000106 E891FFFFFF call atoi ; Вызов подпрограммы перевода символа в число
23 0000010B A3[0A000000] mov [B],eax ; запись преобразованного числа в 'B'
24 ; ----- Записываем 'A' в переменную 'max'
25 00000110 8B0D[35000000] mov ecx,[A] ; 'ecx = A'
26 00000116 890D[00000000] mov [max],ecx ; 'max = A'
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 0000011C 3B0D[39000000] cmp ecx,[C] ; Сравниваем 'A' и 'C'
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
35 *****
36 00000130 E867FFFFFF call atoi ; Вызов подпрограммы перевода символа в число
37 00000135 A3[00000000] mov [max],eax ; запись преобразованного числа в 'max'
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 sprintf ; Вывод сообщения 'Наибольшее число: '
47 0000015E A1[00000000] mov eax,[max]
48 00000163 E81EFFFFFF call iprintfLF ; 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 20 - 95, 2, 61



```
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
```

Matlab ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS

Рис. 2.12: Файл lab8-3.asm

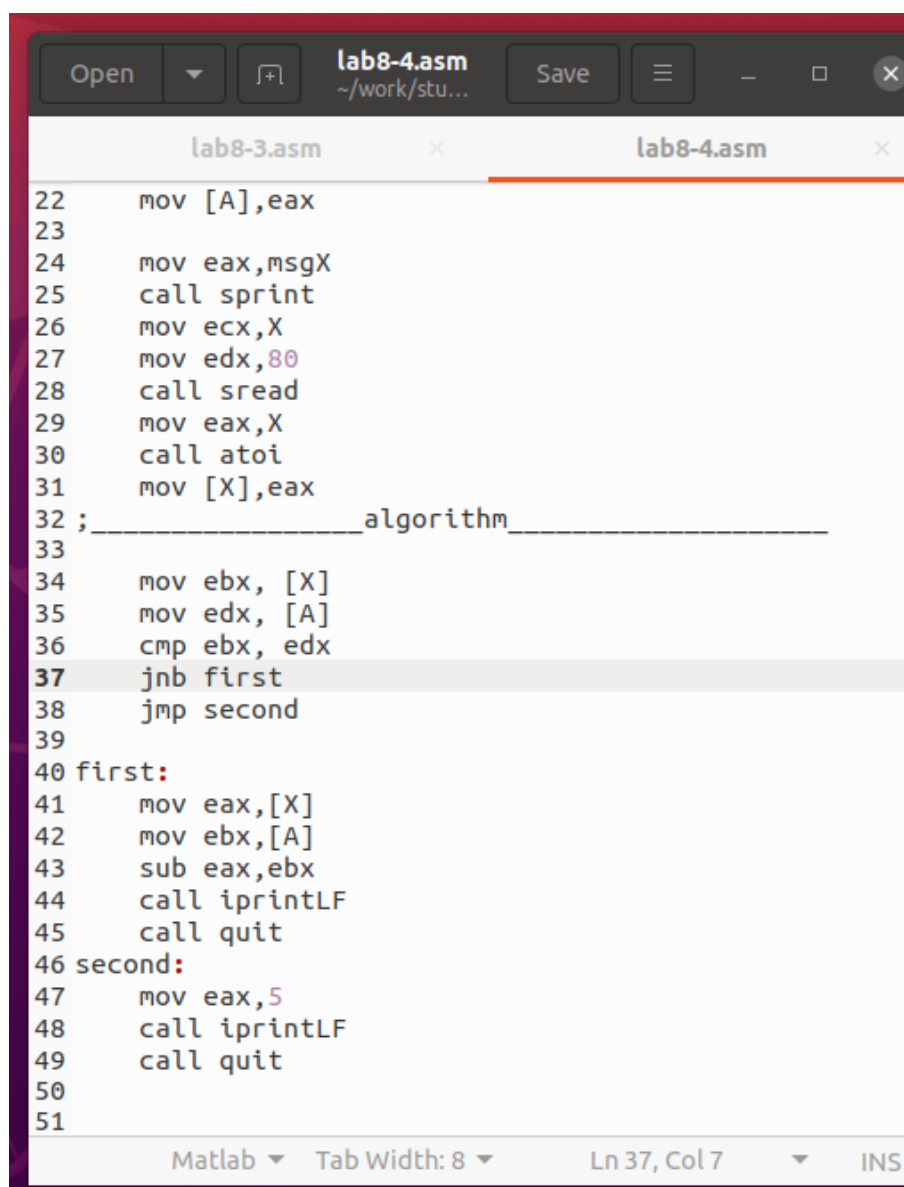
```
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура ко  
s/lab08$ nasm -f elf lab8-3.asm  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура ко  
s/lab08$ ld -m elf_i386 -o lab8-3 lab8-3.o  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура ко  
s/lab08$ ./lab8-3  
Input A: 95  
Input B: 2  
Input C: 61  
Smallest: 2  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура ко  
s/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 20

$$\begin{cases} x - a, & x \geq a \\ 5, & x < a \end{cases}$$



```
lab8-4.asm
~/work/stu...
Open Save
lab8-3.asm lab8-4.asm
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, [A]
36 cmp ebx, edx
37 jnb first
38 jmp second
39
40 first:
41 mov eax,[X]
42 mov ebx,[A]
43 sub eax,ebx
44 call iprintLF
45 call quit
46 second:
47 mov eax,5
48 call iprintLF
49 call quit
50
51
Matlab Tab Width: 8 Ln 37, Col 7 INS
```

Рис. 2.14: Файл lab8-4.asm

```
s/lab08$  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура  
s/lab08$ nasm -f elf lab8-4.asm  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура  
s/lab08$ ld -m elf_i386 -o lab8-4 lab8-4.o  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура  
s/lab08$ ./lab8-4  
Input A: 2  
Input X: 1  
5  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура  
s/lab08$ ./lab8-4  
Input A: 1  
Input X: 2  
1  
albertocascante@Ubuntu:~/work/study/2022-2023/Архитектура  
s/lab08$
```

Рис. 2.15: Программа lab8-4.asm

## 3 Выводы

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