

Отчёт по лабораторной работе 7

Архитектура компьютеров и операционные системы

Кенан Гашимов НКАБд-02-23

Содержание

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

Список иллюстраций

2.1	Редактирование файла lab7-1.asm	7
2.2	Компиляция и проверка программы программы lab7-1.asm	8
2.3	Редактирование файла lab7-1.asm	9
2.4	Компиляция и проверка программы программы lab7-1.asm	10
2.5	Редактирование файла lab7-1.asm	11
2.6	Компиляция и проверка программы программы lab7-1.asm	12
2.7	Редактирование файла lab7-2.asm	13
2.8	Компиляция и проверка программы программы lab7-2.asm	14
2.9	Файл листинга lab7-2	15
2.10	Ошибка трансляции lab7-2	16
2.11	Файл листинга с ошибкой lab7-2	17
2.12	Редактирование файла lab7-3.asm	18
2.13	Компиляция и проверка программы программы lab7-3.asm	19
2.14	Редактирование файла lab7-4.asm	20
2.15	Компиляция и проверка программы программы lab7-4.asm	21

Список таблиц

1 Цель работы

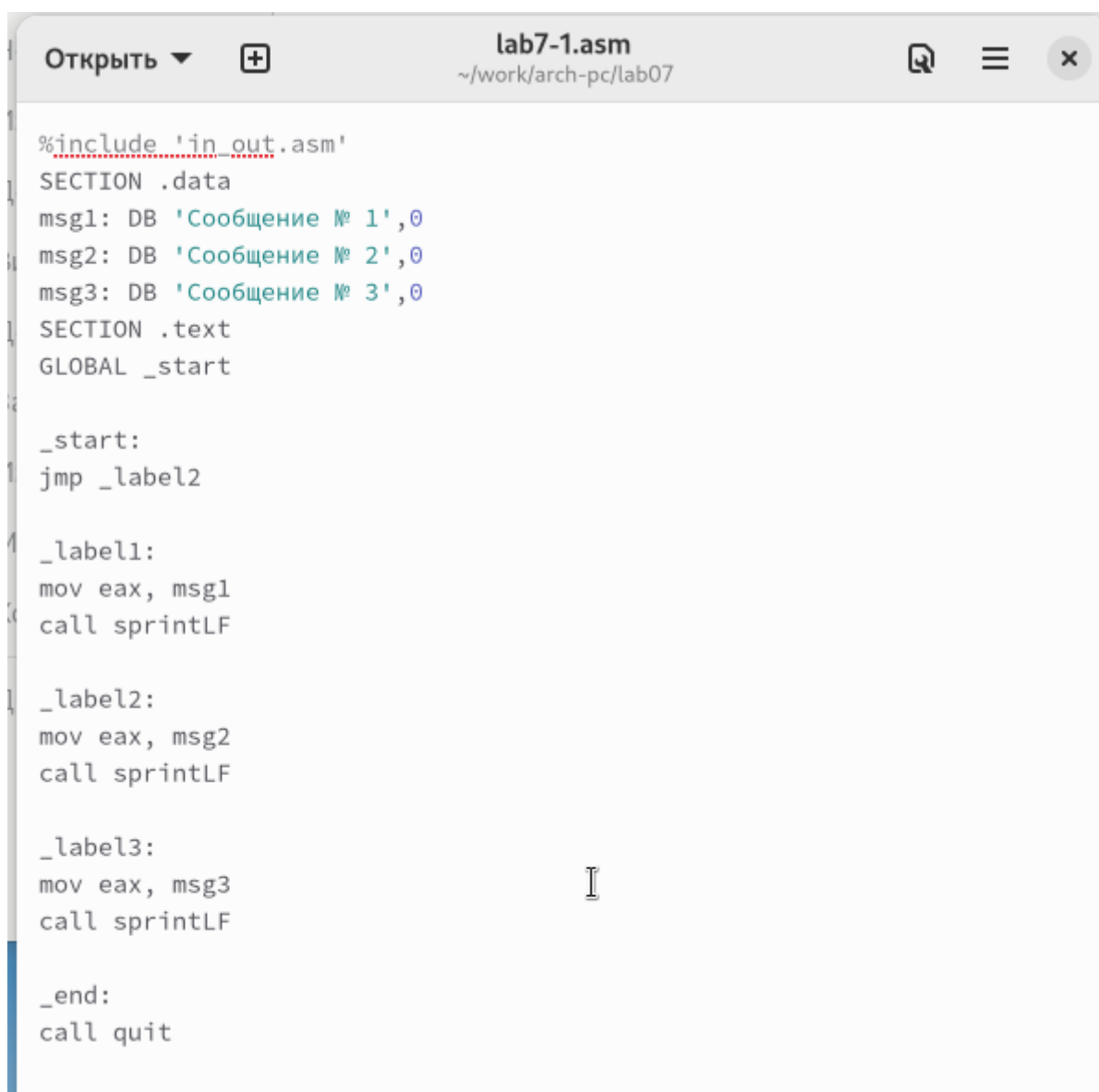
Целью работы является изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

2 Выполнение лабораторной работы

Создал каталог для программ лабораторной работы № 7 и файл lab7-1.asm.

В NASM инструкция `jmp` используется для осуществления безусловных переходов. Давайте рассмотрим пример программы, где используется инструкция `jmp`.

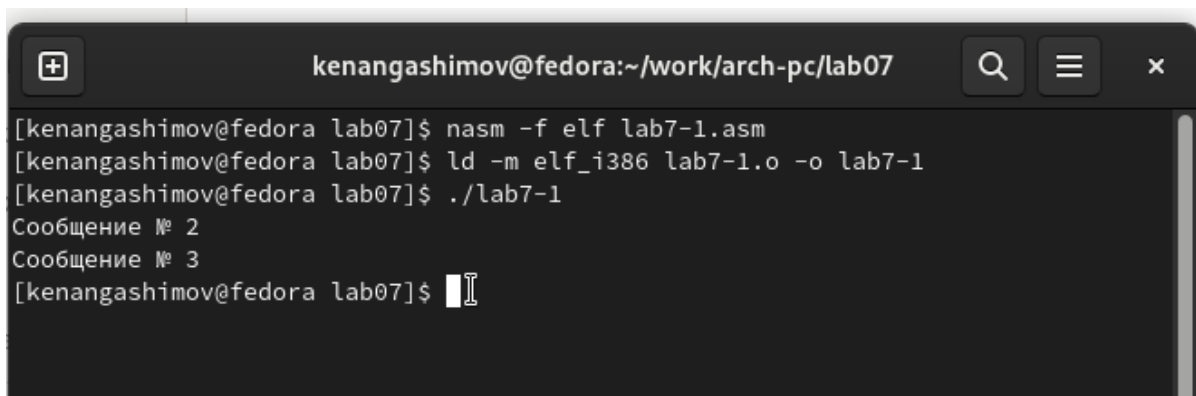
Написал текст программы из листинга 7.1 в файл lab7-1.asm.



```
1 %include 'in_out.asm'
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
14 call sprintLF
15
16 _label2:
17 mov eax, msg2
18 call sprintLF
19
20 _label3:
21 mov eax, msg3
22 call sprintLF
23
24 _end:
25 call quit
```

Рис. 2.1: Редактирование файла lab7-1.asm

Создал исполняемый файл и запустил его.

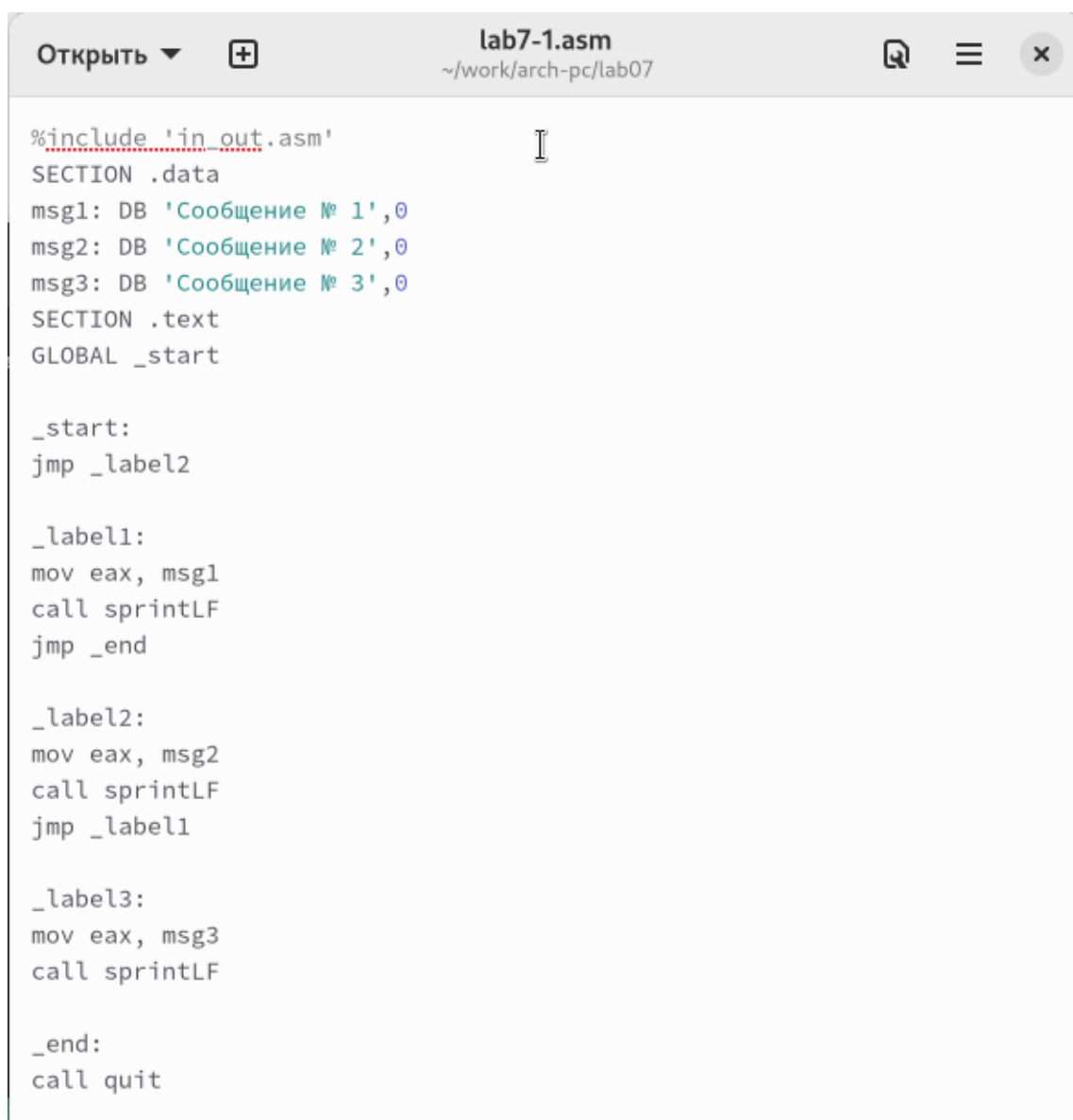
A terminal window with a dark background. The title bar shows the user 'kenangashimov' on a 'fedora' machine, in the directory '~/work/arch-pc/lab07'. The terminal contains the following text:

```
[kenangashimov@fedora lab07]$ nasm -f elf lab7-1.asm
[kenangashimov@fedora lab07]$ ld -m elf_i386 lab7-1.o -o lab7-1
[kenangashimov@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 3
[kenangashimov@fedora lab07]$
```

Рис. 2.2: Компиляция и проверка программы программы lab7-1.asm

Инструкция `jmp` позволяет осуществлять переходы не только вперед, но и назад. Мы изменим программу так, чтобы она сначала выводила “Сообщение № 2”, затем “Сообщение № 1” и завершала работу. Для этого мы добавим в текст программы после вывода “Сообщения № 2” инструкцию `jmp` с меткой `_label1` (переход к инструкциям вывода “Сообщения № 1”) и после вывода “Сообщения № 1” добавим инструкцию `jmp` с меткой `_end` (переход к инструкции `call quit`).

Изменил текст программы в соответствии с листингом 7.2.



```
Открыть ▾ + lab7-1.asm ~/work/arch-pc/lab07
%include 'in_out.asm'
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
call sprintLF
jmp _end

_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF

_end:
call quit
```

Рис. 2.3: Редактирование файла lab7-1.asm

```
[kenangashimov@fedora lab07]$  
[kenangashimov@fedora lab07]$ nasm -f elf lab7-1.asm  
[kenangashimov@fedora lab07]$ ld -m elf_i386 lab7-1.o -o lab7-1  
[kenangashimov@fedora lab07]$ ./lab7-1  
Сообщение № 2  
Сообщение № 1  
[kenangashimov@fedora lab07]$
```


Рис. 2.4: Компиляция и проверка программы программы lab7-1.asm

Изменил текст программы, изменив инструкции jmp, чтобы вывод программы был следующим:

Сообщение № 3

Сообщение № 2

Сообщение № 1



```
Открыть ▾ + lab7-1.asm ~/work/arch-pc/lab07
%include 'in_out.asm'
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
call sprintLF
jmp _end

_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

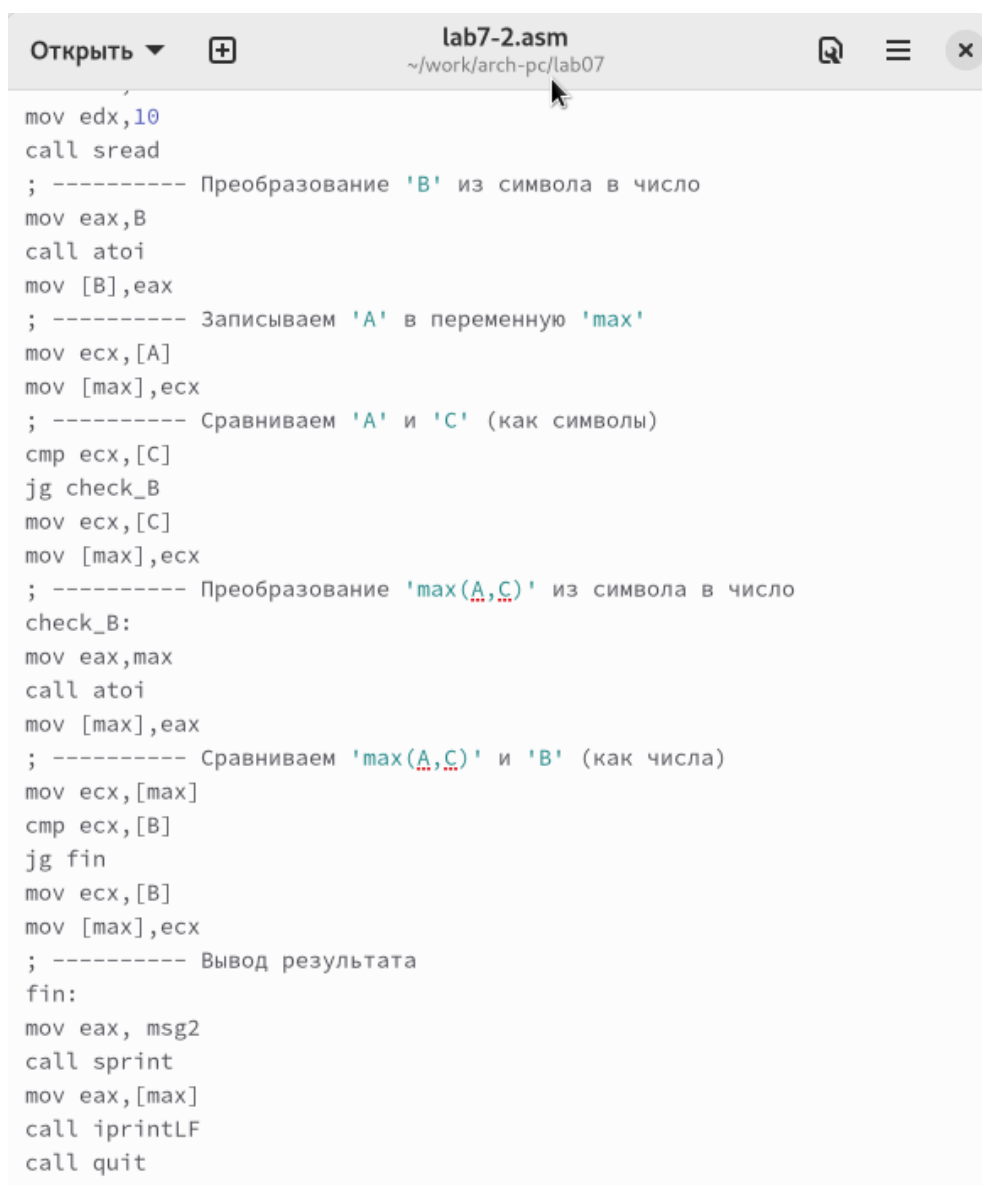
Рис. 2.5: Редактирование файла lab7-1.asm

```
[kenangashimov@fedora lab07]$  
[kenangashimov@fedora lab07]$ nasm -f elf lab7-1.asm  
[kenangashimov@fedora lab07]$ ld -m elf_i386 lab7-1.o -o lab7-1  
[kenangashimov@fedora lab07]$ ./lab7-1  
Сообщение № 3  
Сообщение № 2  
Сообщение № 1  
[kenangashimov@fedora lab07]$
```

Рис. 2.6: Компиляция и проверка программы программы lab7-1.asm

Использование инструкции `jmp` приводит к переходу в любом случае. Однако, при написании программ часто необходимо использовать условные переходы, то есть переход должен происходить, если выполнено определенное условие. Давайте рассмотрим программу, которая определяет и выводит на экран наибольшую из трех целочисленных переменных: А, В и С. Значения для А и С задаются в программе, а значение В вводится с клавиатуры.

Создал исполняемый файл и проверил его работу для разных значений В.



```
mov edx,10
call sread
; ----- Преобразование 'B' из символа в число
mov eax,B
call atoi
mov [B],eax
; ----- Записываем 'A' в переменную 'max'
mov ecx,[A]
mov [max],ecx
; ----- Сравниваем 'A' и 'C' (как символы)
cmp ecx,[C]
jg check_B
mov ecx,[C]
mov [max],ecx
; ----- Преобразование 'max(A,C)' из символа в число
check_B:
mov eax,max
call atoi
mov [max],eax
; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
mov ecx,[max]
cmp ecx,[B]
jg fin
mov ecx,[B]
mov [max],ecx
; ----- Вывод результата
fin:
mov eax, msg2
call sprint
mov eax,[max]
call iprintLF
call quit
```

Рис. 2.7: Редактирование файла lab7-2.asm

```
[kenangashimov@fedora lab07]$ nasm -f elf lab7-2.asm
[kenangashimov@fedora lab07]$ ld -m elf_i386 lab7-2.o -o lab7-2
[kenangashimov@fedora lab07]$ ./lab7-2
Введите B: 20
Наибольшее число: 50
[kenangashimov@fedora lab07]$ ./lab7-2
Введите B: 30
Наибольшее число: 50
[kenangashimov@fedora lab07]$ ./lab7-2
Введите B: 60
Наибольшее число: 60
[kenangashimov@fedora lab07]$
```

Рис. 2.8: Компиляция и проверка программы программы lab7-2.asm

Обычно при ассемблировании с помощью `nasm` создается только объектный файл. Для получения файла листинга можно использовать ключ `-l` и указать имя файла листинга в командной строке.

```
6 00000039 35300000 C dd '50'
7
8 00000000 <res Ah> max resb 10
9 0000000A <res Ah> B resb 10
10
11 section .text
12 global _start
13 _start:
14 ; ----- Вывод сообщения 'Введите B: '
15 000000E8 B8[00000000] mov eax,msg1
16 000000ED E81DFFFFFF call printf
17 ; ----- Ввод 'B'
18 000000F2 B9[0A000000] mov ecx,B
19 000000F7 BA0A000000 mov edx,10
20 000000FC E842FFFFFF call sread
21 ; ----- Преобразование 'B' из символа в число
22 00000101 B8[0A000000] mov eax,B
23 00000106 E891FFFFFF call atoi
24 0000010B A3[0A000000] mov [B],eax
25 ; ----- Записываем 'A' в переменную 'max'
26 00000110 8B0D[35000000] mov ecx,[A]
27 00000116 890D[00000000] mov [max],ecx
28 ; ----- Сравниваем 'A' и 'C' (как символы)
29 0000011C 3B0D[39000000] cmp ecx,[C]
30 00000122 7F0C jg check_B
31 00000124 8B0D[39000000] mov ecx,[C]
32 0000012A 890D[00000000] mov [max],ecx
33 ; ----- Преобразование 'max(A,C)' из символа в число
34 check_B:
35 00000130 B8[00000000] mov eax,max
36 00000135 E862FFFFFF call atoi
37 0000013A A3[00000000] mov [max],eax
```

Рис. 2.9: Файл листинга lab7-2

Я внимательно ознакомился с форматом и содержимым файла листинга. Подробно поясню некоторые элементы листинга.

строка 168

- 168 - номер строки
- 000000DB - адрес
- BB00000000 - машинный код
- mov ebx, 0 - Помещаем 0 в регистр ebx

строка 169

- 169 - номер строки
- 000000E0 - адрес
- B801000000 - машинный код
- mov eax, 1 - Помещаем 1 в регистр eax

строка 170

- 26 - номер строки
- 000000E5 - адрес
- CD80 - машинный код
- int 80h - вызов ядра

Открыл файл программы lab7-2.asm и в инструкции с двумя операндами удалил один из операндов. После этого я выполнил трансляцию программы и получил файл листинга.

```
[kenangashimov@fedora lab07]$  
[kenangashimov@fedora lab07]$ nasm -f elf lab7-2.asm -l lab7-2.lst  
[kenangashimov@fedora lab07]$  
[kenangashimov@fedora lab07]$ nasm -f elf lab7-2.asm -l lab7-2.lst  
lab7-2.asm:34: error: invalid combination of opcode and operands  
[kenangashimov@fedora lab07]$  
[kenangashimov@fedora lab07]$
```

Рис. 2.10: Ошибка трансляции lab7-2


```

lab7-2.asm
lab7-2.lst

196 21 00000101 B8[0A000000] mov eax,B
197 22 00000106 E891FFFFFF call atoi
198 23 0000010B A3[0A000000] mov [B],eax
199 24 ; ----- Записываем 'A' в переменную 'max'
200 25 00000110 8B0D[35000000] mov ecx,[A]
201 26 00000116 890D[00000000] mov [max],ecx
202 27 ; ----- Сравниваем 'A' и 'C' (как символы)
203 28 0000011C 3B0D[39000000] cmp ecx,[C]
204 29 00000122 7F0C jg check_B
205 30 00000124 8B0D[39000000] mov ecx,[C]
206 31 0000012A 890D[00000000] mov [max],ecx
207 32 ; ----- Преобразование 'max(A,C)' из символа в число
208 33 check_B:
209 34 mov eax,
210 34 ***** error: invalid combination of opcode and operands
211 35 00000130 E867FFFFFF call atoi
212 36 00000135 A3[00000000] mov [max],eax
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]
216 40 00000146 7F0C jg fin
217 41 00000148 8B0D[0A000000] mov 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 E8B1FFFFFF call sprint
223 47 0000015E A1[00000000] mov eax,[max]
224 48 00000163 E81EFFFFFF call iprintLF
225 49 00000168 E86EFFFFFF call quit

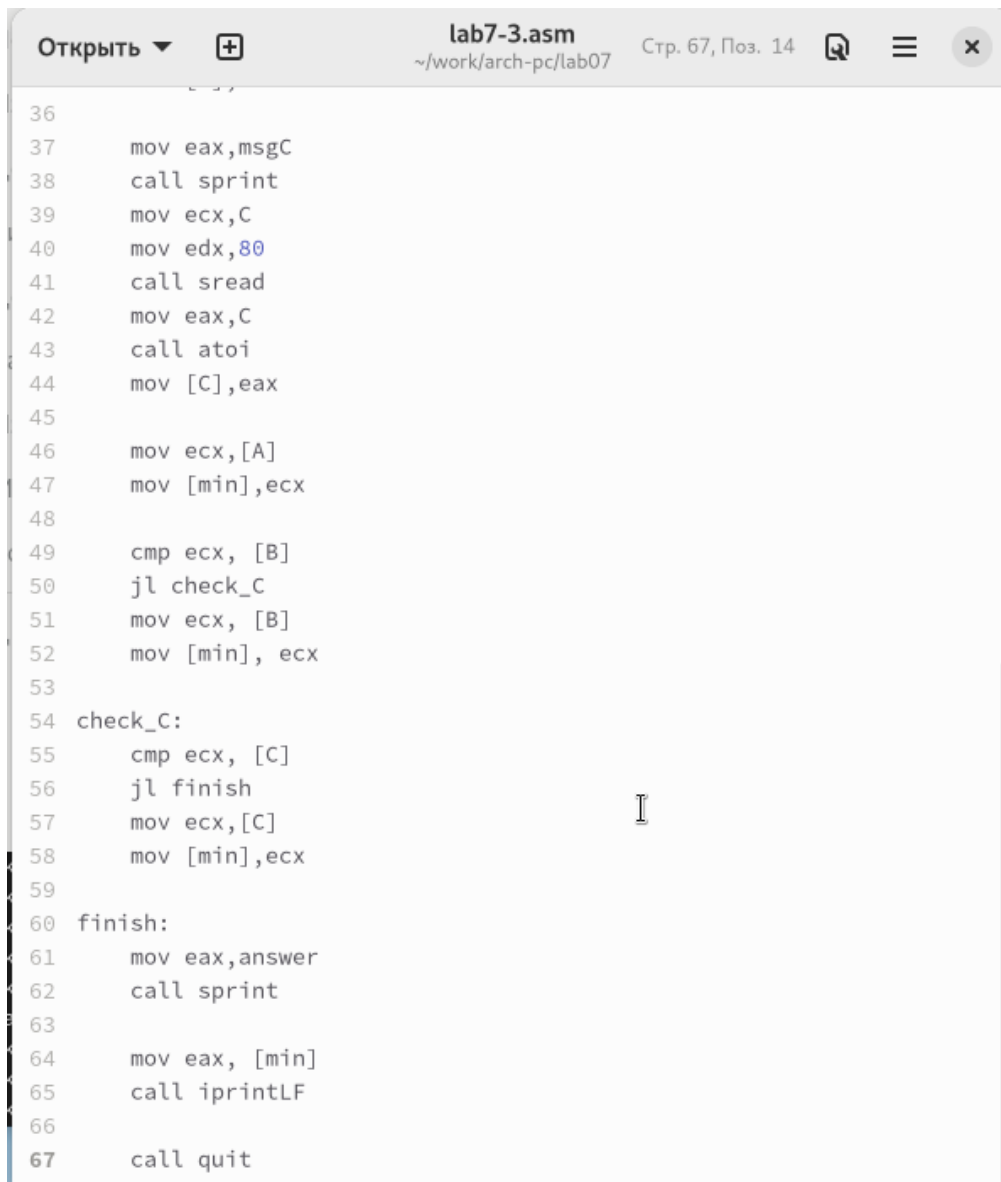
```

Рис. 2.11: Файл листинга с ошибкой lab7-2

Хотя объектный файл не удалось создать из-за ошибки, я все же получил листинг, в котором указано место ошибки.

Напишите программу нахождения наименьшей из 3 целочисленных переменных a,b и c. Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 6. Создайте исполняемый файл и проверьте его работу

для варианта 1 - 17, 23, 45



```
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
46     mov ecx,[A]
47     mov [min],ecx
48
49     cmp ecx, [B]
50     jl check_C
51     mov ecx, [B]
52     mov [min], ecx
53
54 check_C:
55     cmp ecx, [C]
56     jl finish
57     mov ecx,[C]
58     mov [min],ecx
59
60 finish:
61     mov eax,answer
62     call sprint
63
64     mov eax, [min]
65     call iprintLF
66
67     call quit
```

Рис. 2.12: Редактирование файла lab7-3.asm

```

[kenangashimov@fedora lab07]$
[kenangashimov@fedora lab07]$ nasm -f elf lab7-3.asm
[kenangashimov@fedora lab07]$ ld -m elf_i386 lab7-3.o -o lab7-3
[kenangashimov@fedora lab07]$ ./lab7-3
Input A: 17
Input B: 23
Input C: 45
Smallest: 17
[kenangashimov@fedora lab07]$

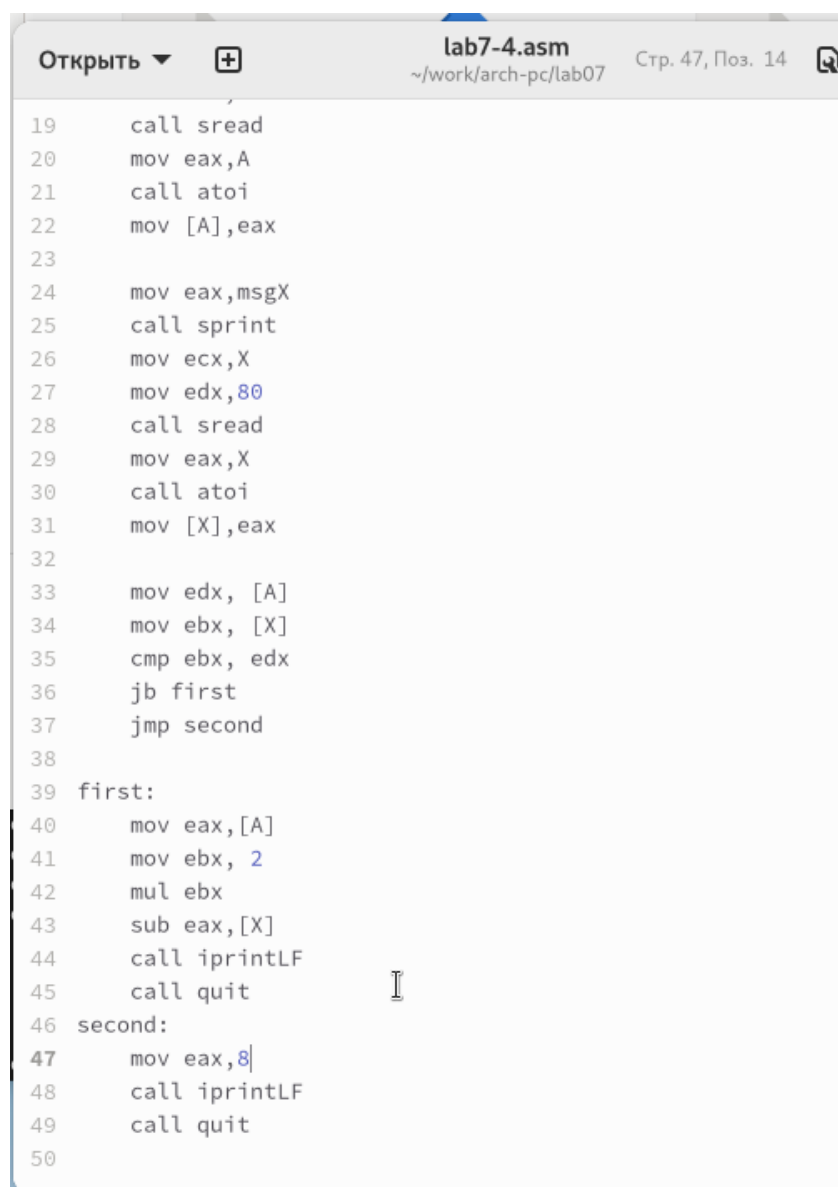
```

Рис. 2.13: Компиляция и проверка программы программы lab7-3.asm

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 7.6 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 7.6.

для варианта 1

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



```
Открыть ▾ + lab7-4.asm Стр. 47, Поз. 14
~/work/arch-pc/lab07

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
33    mov edx, [A]
34    mov ebx, [X]
35    cmp ebx, edx
36    jb first
37    jmp second
38
39 first:
40    mov eax,[A]
41    mov ebx, 2
42    mul ebx
43    sub eax,[X]
44    call iprintLF
45    call quit
46 second:
47    mov eax,8
48    call iprintLF
49    call quit
50
```

Рис. 2.14: Редактирование файла lab7-4.asm

```
[kenangashimov@fedora lab07]$  
[kenangashimov@fedora lab07]$ nasm -f elf lab7-4.asm  
[kenangashimov@fedora lab07]$ ld -m elf_i386 lab7-4.o -o lab7-4  
[kenangashimov@fedora lab07]$ ./lab7-4  
Input A: 2  
Input X: 1  
3  
[kenangashimov@fedora lab07]$ ./lab7-4  
Input A: 1  
Input X: 2  
8  
[kenangashimov@fedora lab07]$
```

Рис. 2.15: Компиляция и проверка программы программы lab7-4.asm

3 Выводы

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