

# **Отчёт по лабораторной работе 7**

**Архитектура компьютеров**

Эйвази Мани НПИбд-03-24

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	21

## Список иллюстраций

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

## 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. Создал каталог для программам лабораторной работы № 7 и файл lab7-1.asm
2. Инструкция `jmp` в NASM используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции `jmp`.

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

```
manieyvazi@VirtualBox: ~/work/arch
GNU nano 4.8 lab7-1.asm
#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

_label2:
mov eax, msg2
call sprintLF

_label3:
mov eax, msg3
call sprintLF

_end:
call quit
```

Рис. 2.1: Программа lab7-1.asm

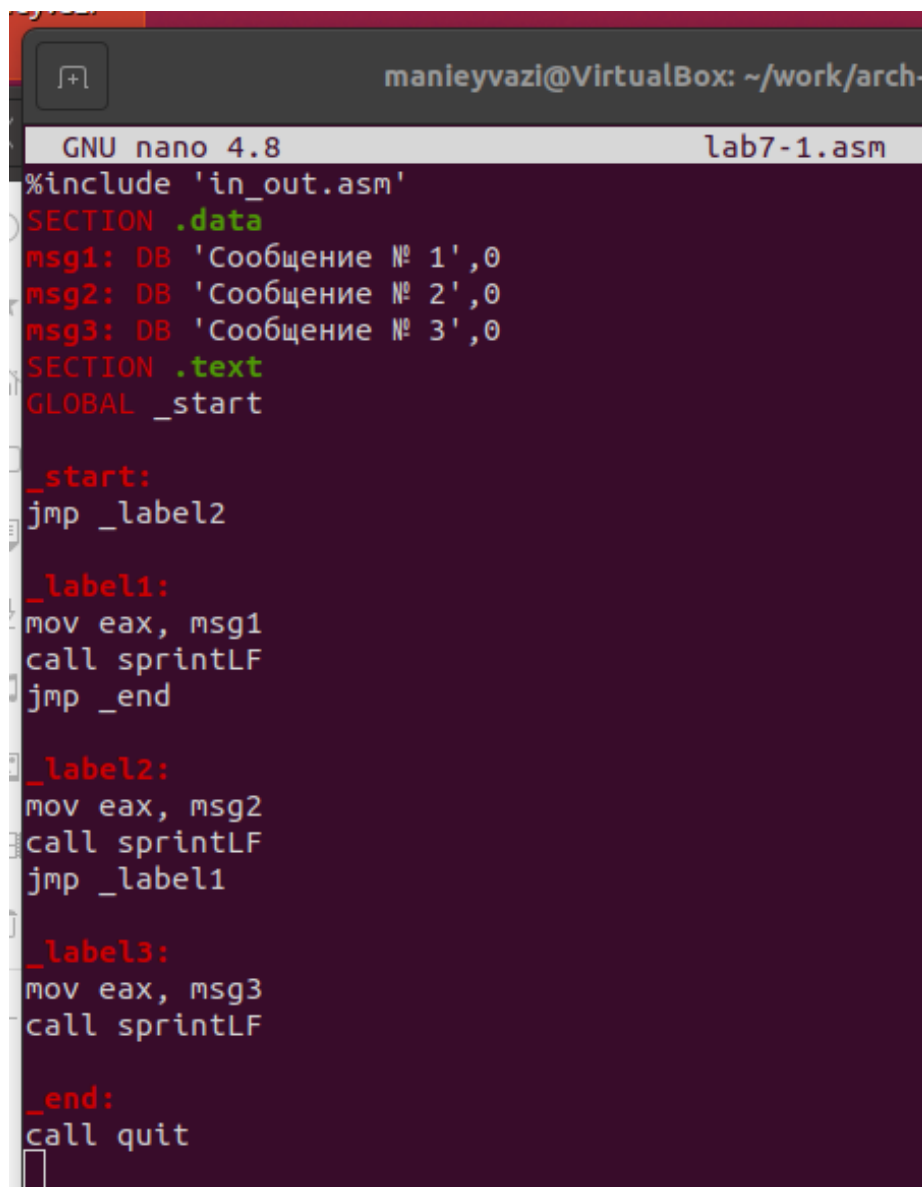
Создал исполняемый файл и запустил его.

```
manieyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
manieyvazi@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
manieyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
manieyvazi@VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.2: Запуск программы lab7-1.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`).

Изменил текст программы в соответствии с листингом 7.2.



```
manieyvazi@VirtualBox: ~/work/arch-  
GNU nano 4.8 lab7-1.asm  
%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 sprintfLF  
jmp _end  
  
_label2:  
mov eax, msg2  
call sprintfLF  
jmp _label1  
  
_label3:  
mov eax, msg3  
call sprintfLF  
  
_end:  
call quit
```

Рис. 2.3: Программа lab7-1.asm



```
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
manleyvazi@VirtualBox:~/work/arch-pc/lab07$
```

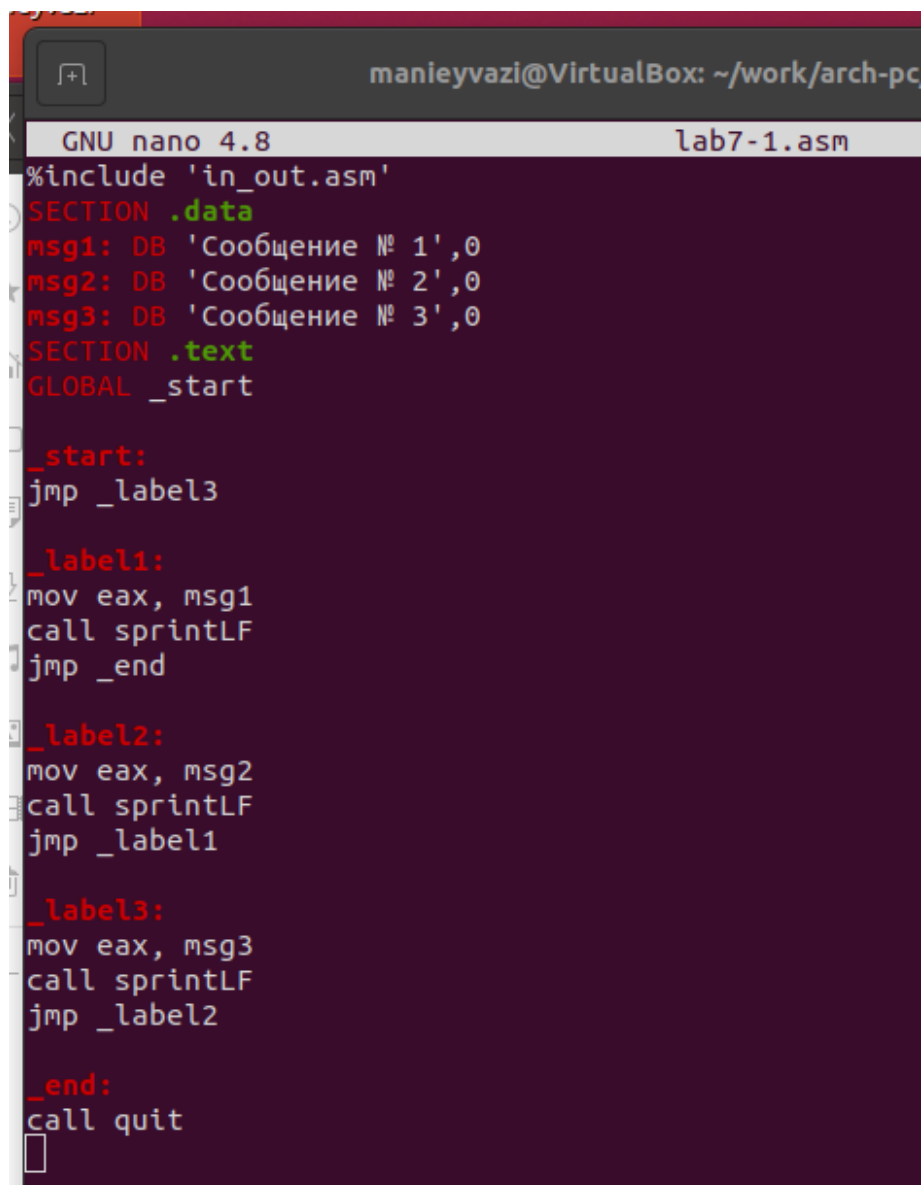
Рис. 2.4: Запуск программы lab7-1.asm

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

Сообщение № 3

Сообщение № 2

Сообщение № 1



```
manieyvazi@VirtualBox: ~/work/arch-pc
GNU nano 4.8 lab7-1.asm
#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

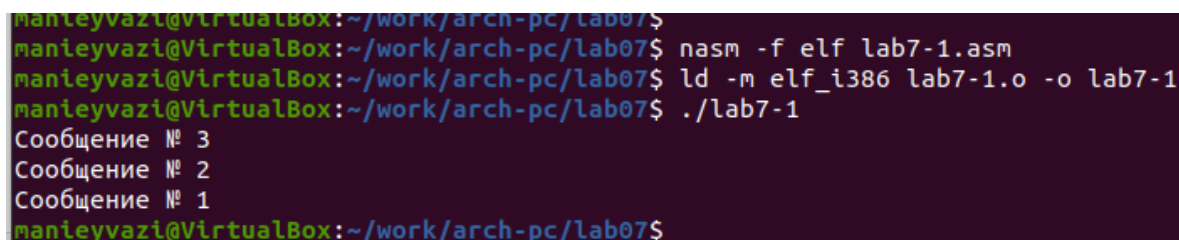
_label1:
mov eax, msg1
call sprintLF
jmp _end

_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

Рис. 2.5: Программа lab7-1.asm

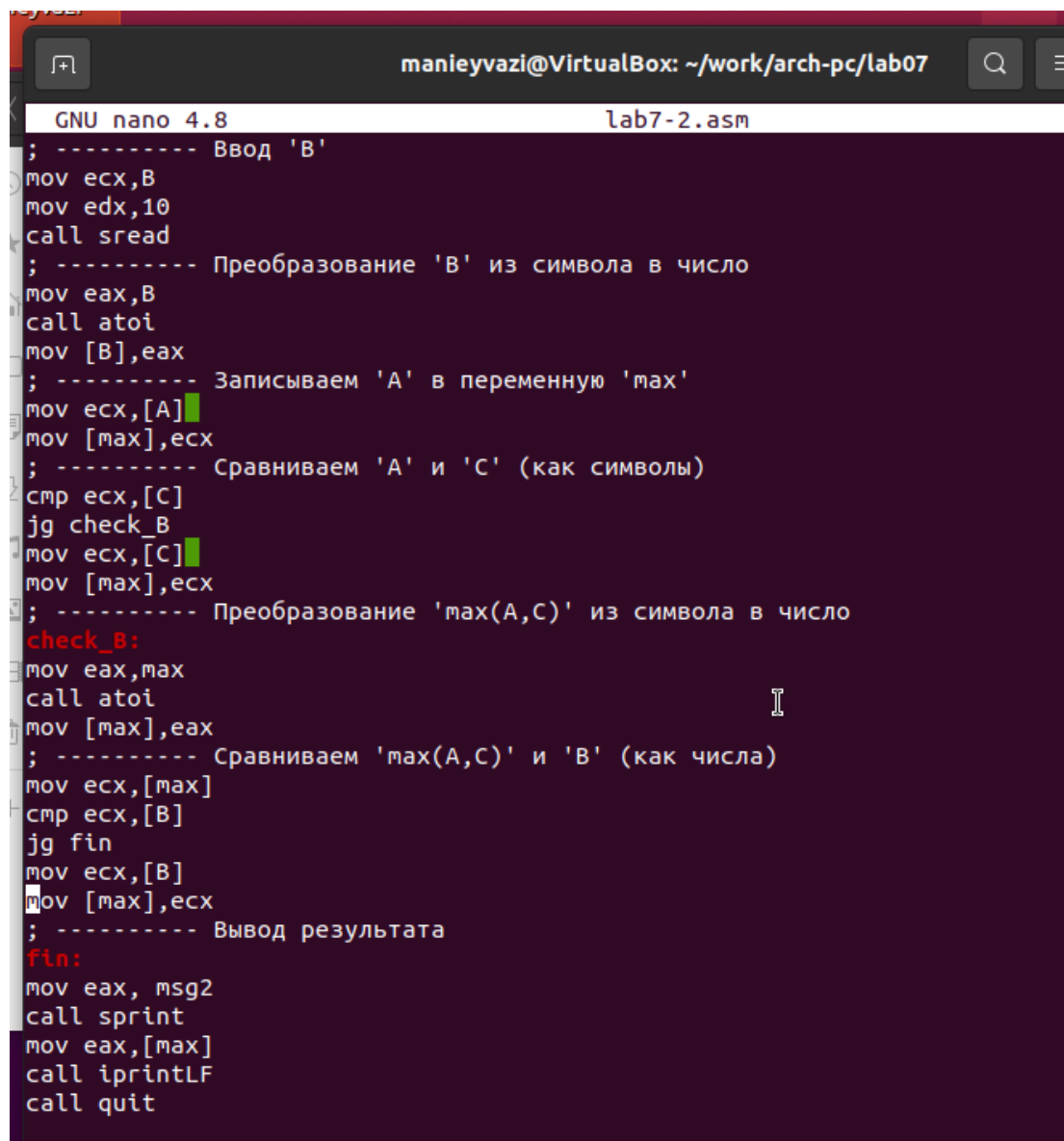


```
manieyvazi@VirtualBox:~/work/arch-pc/lab07$
manieyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
manieyvazi@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
manieyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
manieyvazi@VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.6: Запуск программы lab7-1.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.

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



```
manieyvazi@VirtualBox: ~/work/arch-pc/lab07
GNU nano 4.8 lab7-2.asm
; ----- Ввод 'B'
mov ecx,B
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: Программа lab7-2.asm

```
manleyvazi@VirtualBox:~/work/arch-pc/lab07$  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-2.o -o lab7-2  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 10  
Наибольшее число: 50  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 30  
Наибольшее число: 50  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 60  
Наибольшее число: 60  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.8: Запуск программы lab7-2.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.

Создал файл листинга для программы из файла lab7-2.asm

```

lab7-2.lst
~/work/arch-pc/lab07
Save

187 12 _start:
188 13 ; ----- Вывод сообщения 'Введите B: '
189 14 000000E8 B8[00000000] mov eax,msg1
190 15 000000ED E81DFFFFFF call sprint
191 16 ; ----- Ввод 'B'
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
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 00000130 B8[00000000] mov eax,max
210 35 00000135 E862FFFFFF call atoi
211 36 0000013A A3[00000000] mov [max],eax
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]
215 40 0000014B 7F0C jg fin
216 41 0000014D 8B0D[0A000000] mov 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 E8ACFEFFFF call sprint
222 47 00000163 A1[00000000] mov eax,[max]
223 48 00000168 E819FFFFFF call iprintLF

```

Рис. 2.9: Файл листинга lab7-2

Внимательно ознакомился с его форматом и содержимым. Подробно объясню содержимое трёх строк файла листинга по выбору.

строка 200

- 25 - номер строки
- 00000110 - адрес
- 8B0D[35000000] - машинный код
- mov ecx,[A] - код программы

строка 201

- 26 - номер строки

- 00000116 - адрес
- 890D[00000000] - машинный код
- mov [max],ecx - код программы

строка 203

- 28 - номер строки
- 0000011C - адрес
- 3B0D[39000000] - машинный код
- cmp ecx,[C] - код программы

Открыл файл с программой lab7-2.asm и в инструкции с двумя операндами удалил один операнд. Выполнил трансляцию с получением файла листинга.

```
maneyvazi@VirtualBox:~/work/arch-pc/lab07$
maneyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm -l lab7-2.lst
maneyvazi@VirtualBox:~/work/arch-pc/lab07$
maneyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm -l lab7-2.lst
lab7-2.asm:34: error: invalid combination of opcode and operands
maneyvazi@VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.10: Ошибка трансляции lab7-2

```

lab7-2.lst
~/work/arch-pc/lab07
Save
Open
+
189 14 000000E8 B8[00000000] mov eax,msg1
190 15 000000ED E81DFFFFFF call sprint
191 16 ; ----- Ввод 'B'
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
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 *****
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

```

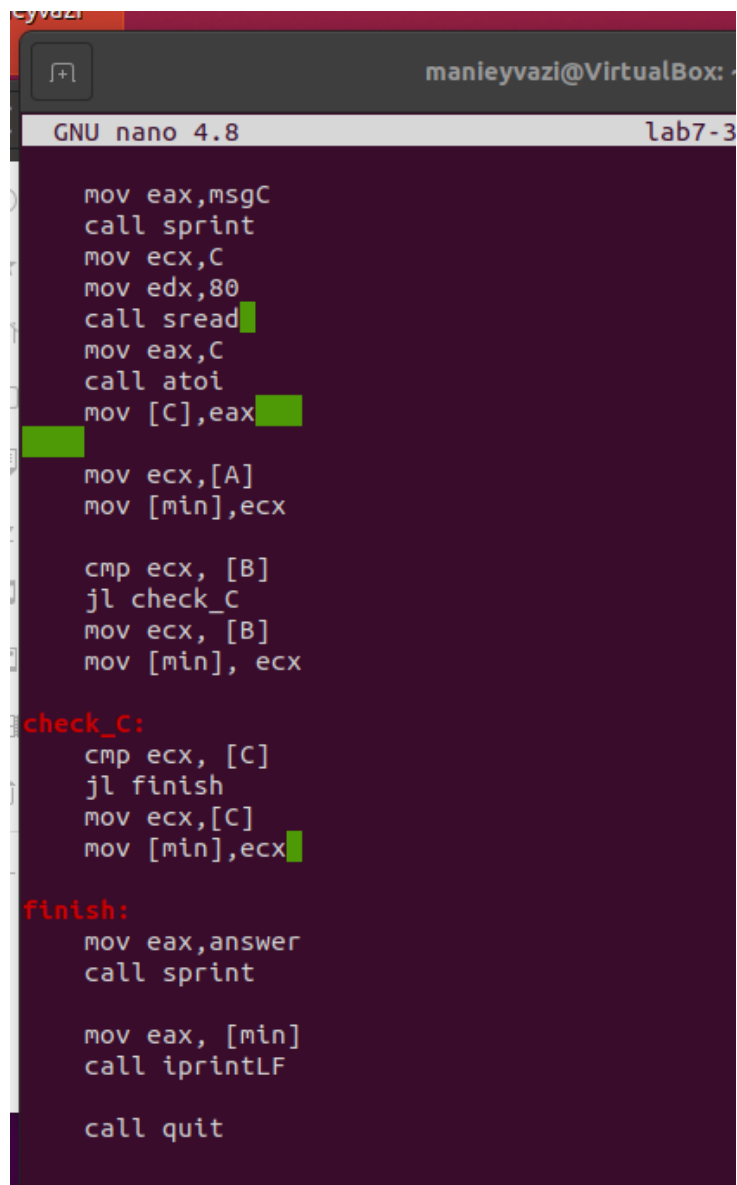
Рис. 2.11: Файл листинга с ошибкой lab7-2

Объектный файл не смог создаться из-за ошибки. Но получился листинг, где выделено место ошибки.

5. 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 8 - 52,33,40





```
manleyvazi@VirtualBox: ~
GNU nano 4.8 lab7-3.asm

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

    mov ecx,[A]
    mov [min],ecx

    cmp ecx, [B]
    jl check_C
    mov ecx, [B]
    mov [min], ecx

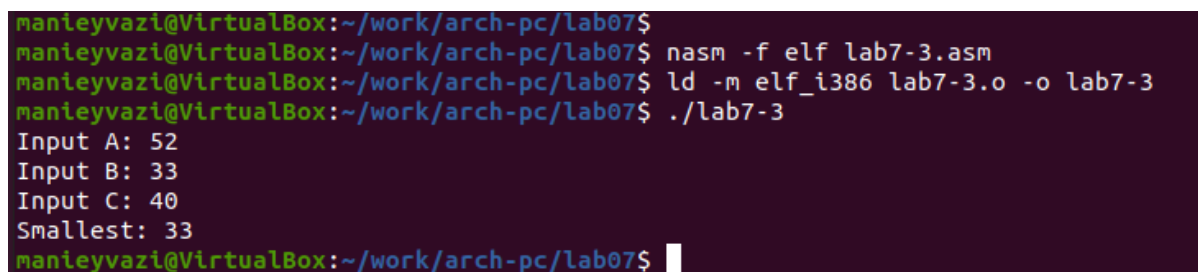
check_C:
    cmp ecx, [C]
    jl finish
    mov ecx,[C]
    mov [min],ecx

finish:
    mov eax,answer
    call sprint

    mov eax, [min]
    call iprintLF

    call quit
```

Рис. 2.12: Программа lab7-3.asm



```
manleyvazi@VirtualBox:~/work/arch-pc/lab07$
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-3.asm
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-3.o -o lab7-3
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-3
Input A: 52
Input B: 33
Input C: 40
Smallest: 33
manleyvazi@VirtualBox:~/work/arch-pc/lab07$
```

Рис. 2.13: Запуск программы lab7-3.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 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 8:

$$\begin{cases} 3a, & a < 3 \\ x + 1, & a \geq 3 \end{cases}$$

При ( $x = 1, a = 4$ ) получается 2

При ( $x = 1, a = 2$ ) получается 6

```
manieyvazi@VirtualBox: ~/w
GNU nano 4.8 lab7-4.asm
mov ecx,A
mov edx,80
call sread
mov eax,A
call atoi
mov [A],eax

mov eax,msgX
call sprint
mov ecx,X
mov edx,80
call sread
mov eax,X
call atoi
mov [X],eax

mov ebx, [A]
mov edx, 3
cmp ebx, edx
jb first
jmp second

first:
mov eax,[A]
mov ebx,3
mul ebx
call iprintLF
call quit

second:
mov eax,[X]
add eax,1
call iprintLF
call quit
```

Рис. 2.14: Программа lab7-4.asm

```
manleyvazi@VirtualBox:~/work/arch-pc/lab07$  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-4.asm  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-4.o -o lab7-4  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-4  
Input A: 4  
Input X: 1  
2  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ ./lab7-4  
Input A: 2  
Input X: 1  
6  
manleyvazi@VirtualBox:~/work/arch-pc/lab07$ █
```

Рис. 2.15: Запуск программы lab7-4.asm

## 3 Выводы

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