

# Отчёт по лабораторной работе 7

Архитектура компьютера

Хулер Александрович Оюн

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Задание для самостоятельной работы . . . . .	16
3	Выводы	21

## Список иллюстраций

2.1	Код программы lab7-1.asm . . . . .	7
2.2	Компиляция и запуск программы lab7-1.asm . . . . .	8
2.3	Код программы lab7-1.asm . . . . .	9
2.4	Компиляция и запуск программы lab7-1.asm . . . . .	9
2.5	Код программы lab7-1.asm . . . . .	10
2.6	Компиляция и запуск программы lab7-1.asm . . . . .	11
2.7	Код программы lab7-2.asm . . . . .	12
2.8	Компиляция и 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	Код программы lab7-3.asm . . . . .	17
2.13	Компиляция и запуск программы lab7-3.asm . . . . .	18
2.14	Код программы lab7-4.asm . . . . .	19
2.15	Компиляция и запуск программы lab7-4.asm . . . . .	20

## Список таблиц

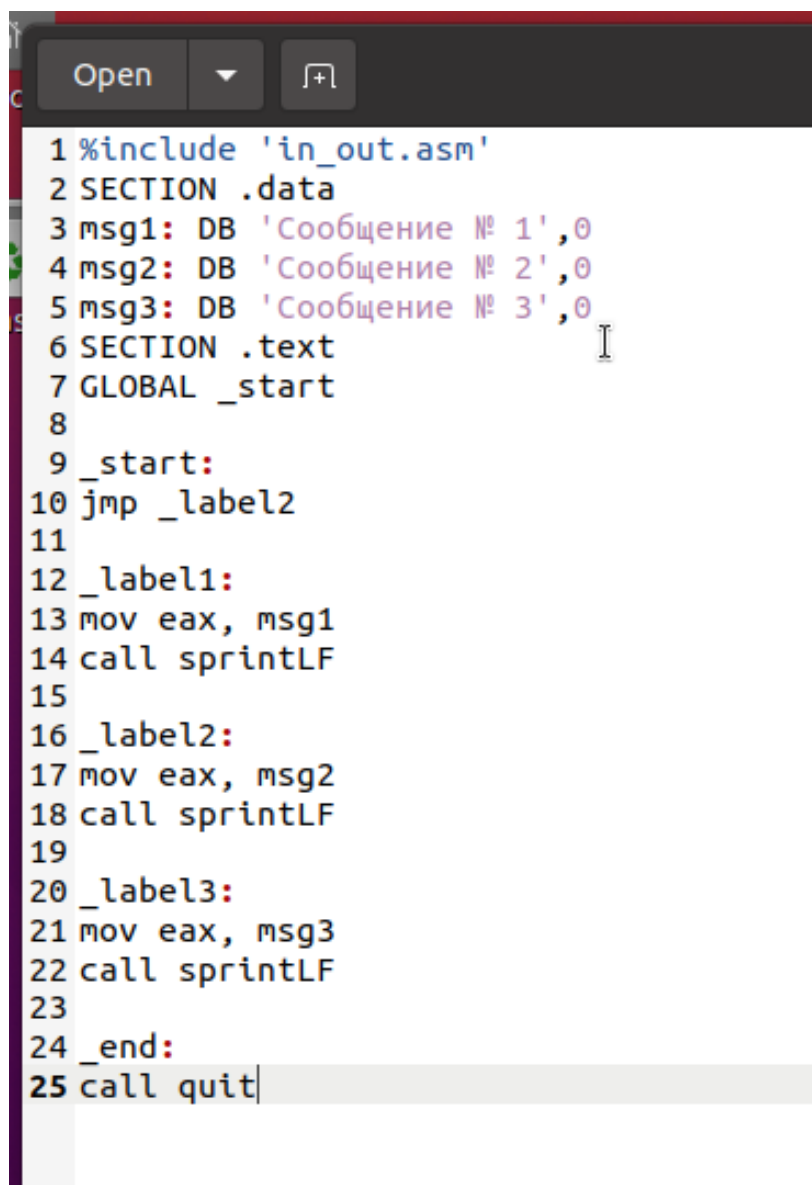
# 1 Цель работы

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

Я организовал папку для работы над седьмой лабораторной и создал файл с исходным кодом lab7-1.asm.

В NASM команда `jmp` применяется для безусловного перехода. Изучил пример кода с этой командой и внёс его 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 sprintf
15
16 _label2:
17 mov eax, msg2
18 call sprintf
19
20 _label3:
21 mov eax, msg3
22 call sprintf
23
24 _end:
25 call quit
```

Рис. 2.1: Код программы lab7-1.asm

Скомпилировал и запустил полученную программу.

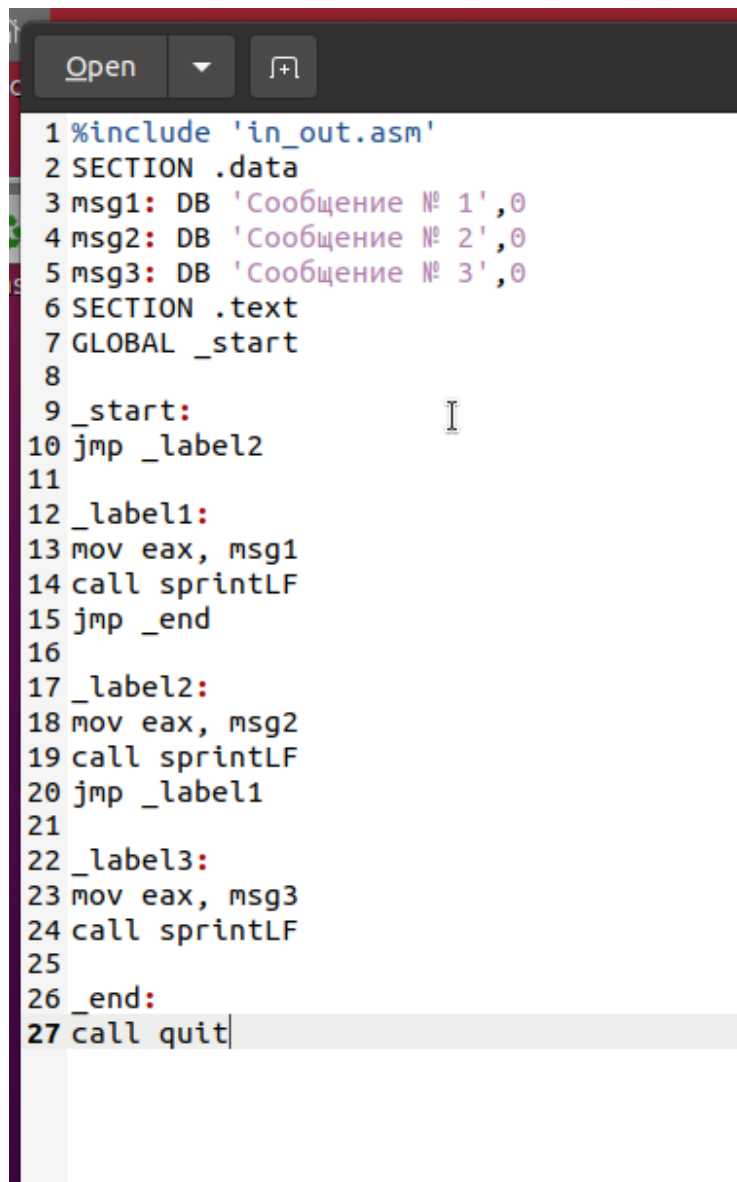
```
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-1.asm
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ld -m elf_i386 lab07-1.o -o lab07-1
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-1
Сообщение № 2
Сообщение № 3
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$
```

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

Команда `jmp` позволяет переходить как вперёд, так и назад в коде. Модифицировал программу так, чтобы она сначала показывала “Сообщение № 2”, а потом “Сообщение № 1”, и после этого завершалась. Это было достигнуто добавлением команды `jmp` с меткой `_label1` после “Сообщения № 2” для перехода к выводу “Сообщения № 1”, и команды `jmp` с меткой `_end` после “Сообщения № 1” для завершения работы через вызов функции `quit`.

Внёс изменения в код, соответствующие листингу 7.2.





```
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 jmp _end
16
17 _label2:
18 mov eax, msg2
19 call sprintLF
20 jmp _label1
21
22 _label3:
23 mov eax, msg3
24 call sprintLF
25
26 _end:
27 call quit
```

Рис. 2.3: Код программы lab7-1.asm



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

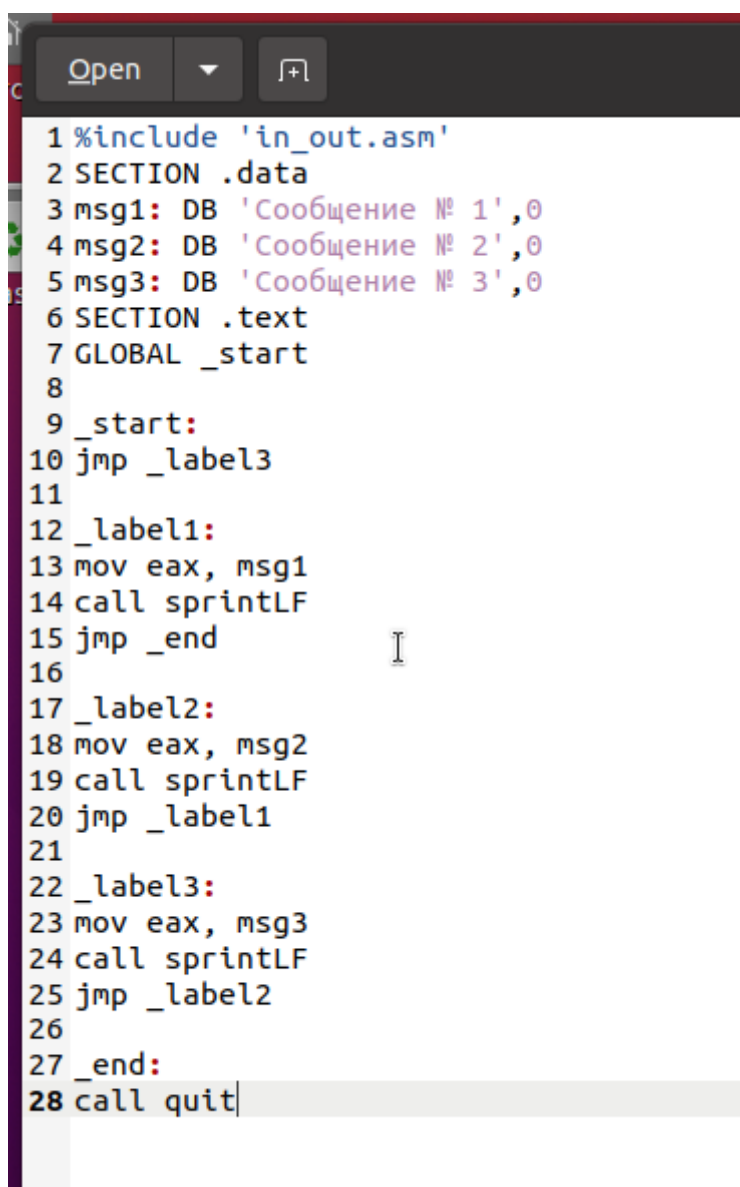
Рис. 2.4: Компиляция и запуск программы lab7-1.asm

Программа была переписана с изменёнными командами jmp для изменения порядка вывода.

Сообщение № 3

Сообщение № 2

Сообщение № 1



```
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 _label3
11
12 _label1:
13 mov eax, msg1
14 call sprintLF
15 jmp _end
16
17 _label2:
18 mov eax, msg2
19 call sprintLF
20 jmp _label1
21
22 _label3:
23 mov eax, msg3
24 call sprintLF
25 jmp _label2
26
27 _end:
28 call quit
```

Рис. 2.5: Код программы lab7-1.asm

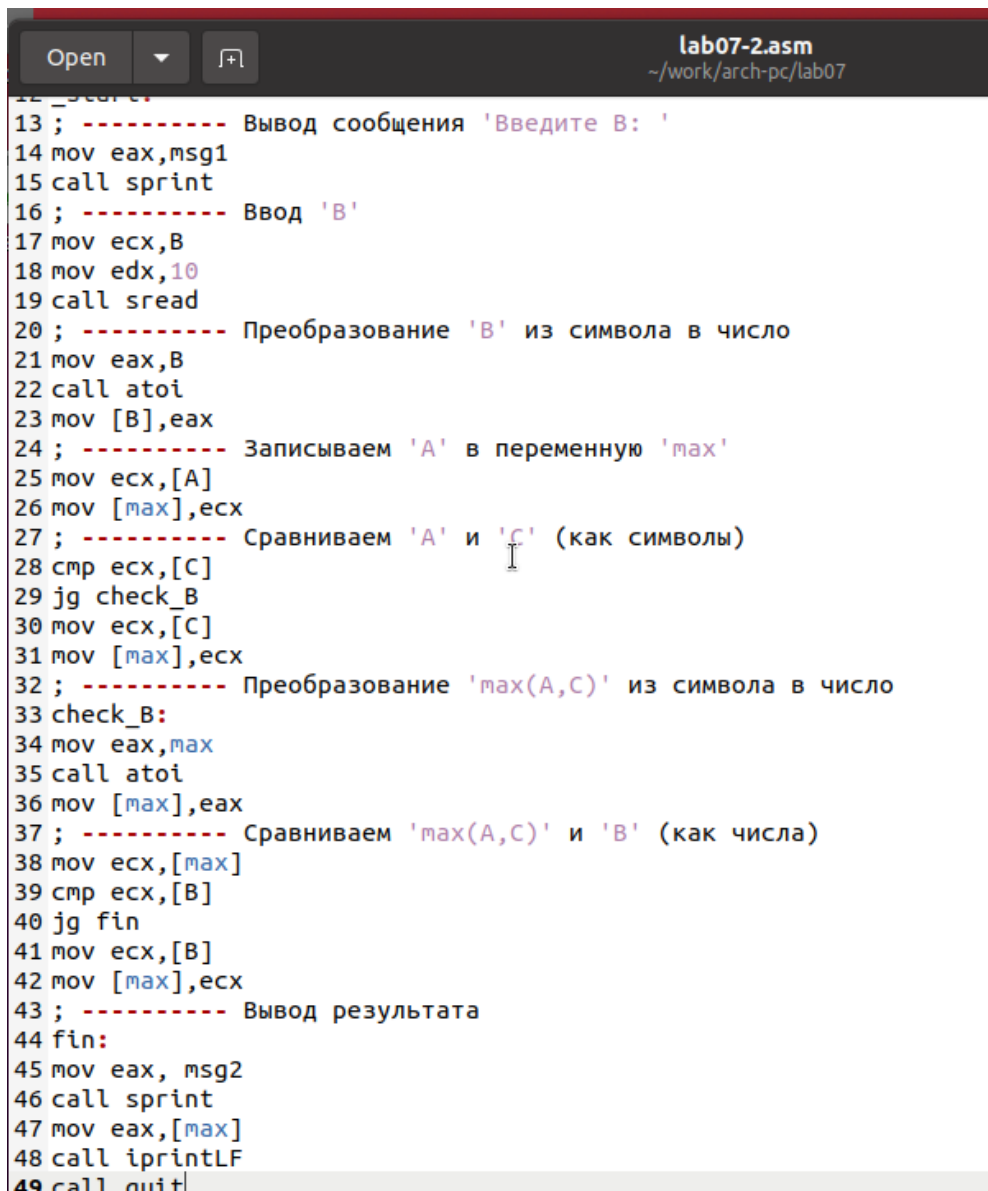
```
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-1.asm
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ld -m elf_i386 lab07-1.o -o lab07-1
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$
```

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

Команда `jmp` всегда приводит к переходу. Однако в программировании часто требуются условные переходы, когда переход выполняется только при определённом условии.

Рассмотрим программу, которая вычисляет и выводит наибольшее из трёх чисел: А, В и С. Значения А и С заданы в коде, а значение В вводится пользователем.

Скомпилировал программу и провёл тестирование с различными вводимыми значениями В.



```
lab07-2.asm
~/work/arch-pc/lab07

12 ; -----
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
23 mov [B],eax
24 ; ----- Записываем 'A' в переменную 'max'
25 mov ecx,[A]
26 mov [max],ecx
27 ; ----- Сравниваем 'A' и 'C' (как символы)
28 cmp ecx,[C]
29 jg check_B
30 mov ecx,[C]
31 mov [max],ecx
32 ; ----- Преобразование 'max(A,C)' из символа в число
33 check_B:
34 mov eax,max
35 call atoi
36 mov [max],eax
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 mov ecx,[max]
39 cmp ecx,[B]
40 jg fin
41 mov ecx,[B]
42 mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 mov eax, msg2
46 call sprint
47 mov eax,[max]
48 call iprintLF
49 call quit
```

Рис. 2.7: Код программы lab7-2.asm

```
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-2.asm
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ld -m elf_i386 lab07-2.o -o lab07-2
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-2
Введите В: 20
Наибольшее число: 50
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-2
Введите В: 40
Наибольшее число: 50
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-2
Введите В: 60
Наибольшее число: 60
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$
```

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

Обычно при компиляции с помощью `nasm` получается лишь объектный файл. Однако, чтобы сформировать файл листинга, следует использовать опцию `-l` и определить имя файла листинга через командную строку.

Сформировал листинг для кода, находящегося в `lab7-2.asm`.

```

190 14 000000E8 B8[00000000]    mov eax,msg1
191 15 000000ED E81DFFFFFF    call sprint
192 16                                ; ----- Ввод 'B'
193 17 000000F2 B9[0A000000]    mov ecx,B
194 18 000000F7 BA0A000000    mov edx,10
195 19 000000FC E842FFFFFF    call sread
196 20                                ; ----- Преобразование 'B' из символа в число
197 21 00000101 B8[0A000000]    mov eax,B
198 22 00000106 E891FFFFFF    call atoi
199 23 0000010B A3[0A000000]    mov [B],eax
200 24                                ; ----- Записываем 'A' в переменную 'max'
201 25 00000110 8B0D[35000000]    mov ecx,[A]
202 26 00000116 890D[00000000]    mov [max],ecx
203 27                                ; ----- Сравниваем 'A' и 'C' (как символы)
204 28 0000011C 3B0D[39000000]    cmp ecx,[C]
205 29 00000122 7F0C                jg check_B
206 30 00000124 8B0D[39000000]    mov ecx,[C]
207 31 0000012A 890D[00000000]    mov [max],ecx
208 32                                ; ----- Преобразование 'max(A,C)' из символа в
число
209 33                                check_B:
210 34 00000130 B8[00000000]    mov eax,max
211 35 00000135 E862FFFFFF    call atoi
212 36 0000013A A3[00000000]    mov [max],eax
213 37                                ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
214 38 0000013F 8B0D[00000000]    mov ecx,[max]
215 39 00000145 3B0D[0A000000]    cmp ecx,[B]
216 40 0000014B 7F0C                jg fin
217 41 0000014D 8B0D[0A000000]    mov ecx,[B]
218 42 00000153 890D[00000000]    mov [max],ecx
219 43                                ; ----- Вывод результата
220 44                                fin:
221 45 00000159 B8[13000000]    mov eax,msg2
222 46 0000015E E8ACFEFFFF    call sprint
223 47 00000163 A1[00000000]    mov eax,[max]

```

Рис. 2.9: Файл листинга lab7-2

Я внимательно ознакомился с форматом и содержимым файла листинга. Подробно объясню содержимое трёх строк из этого файла.

строка 213

- 38 - номер строки в подпрограмме
- 0000013F - адрес
- 8B0D[00000000] - машинный код
- mov ecx,[max] - код программы - копирует MAX в ecx

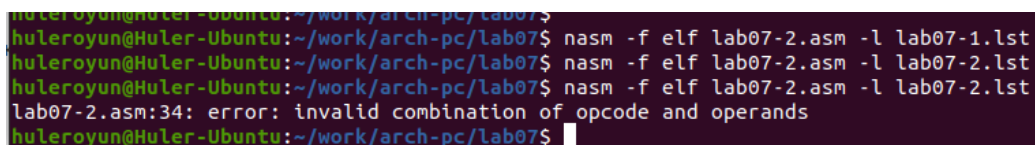
строка 214

- 39 - номер строки в подпрограмме
- 00000145 - адрес
- 3B0D[0A000000] - машинный код
- `cmp esx,[B]` - код программы - сравнивает `esx` и `B`

строка 215

- 40 - номер строки в подпрограмме
- 0000014B - адрес
- 7F0C - машинный код
- `jg fin` - код программы - если больше перейти к метке `fin`

Затем я открыл исходный код в `lab7-2.asm` и удалил один операнд из команды, содержащей два операнда. После этого произвел компиляцию с целью создания файла листинга.



```
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-2.asm -l lab07-1.lst
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-2.asm -l lab07-2.lst
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-2.asm -l lab07-2.lst
lab07-2.asm:34: error: invalid combination of opcode and operands
huleroyn@Huler-Ubuntu:~/work/arch-pc/lab07$
```

Рис. 2.10: Ошибка трансляции `lab7-2`

```

lab07-2.asm
lab07-2.lst

193 17 000000F2 B9[0A000000] mov ecx,B
194 18 000000F7 BA0A000000 mov edx,10
195 19 000000FC E842FFFFFF call sread
196 20 ; ----- Преобразование 'B' из символа в число
197 21 00000101 B8[0A000000] mov eax,B
198 22 00000106 E891FFFFFF call atoi
199 23 0000010B A3[0A000000] mov [B],eax
200 24 ; ----- Записываем 'A' в переменную 'max'
201 25 00000110 8B0D[35000000] mov ecx,[A]
202 26 00000116 890D[00000000] mov [max],ecx
203 27 ; ----- Сравниваем 'A' и 'C' (как символы)
204 28 0000011C 3B0D[39000000] cmp ecx,[C]
205 29 00000122 7F0C jg check_B
206 30 00000124 8B0D[39000000] mov ecx,[C]
207 31 0000012A 890D[00000000] mov [max],ecx
208 32 ; ----- Преобразование 'max(A,C)' из символа в
число
209 33 check_B:
210 34 mov eax,
211 34 ***** error: invalid combination of opcode and operands
212 35 00000130 E867FFFFFF call atoi
213 36 00000135 A3[00000000] mov [max],eax
214 37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
215 38 0000013A 8B0D[00000000] mov ecx,[max]
216 39 00000140 3B0D[0A000000] cmp ecx,[B]
217 40 00000146 7F0C jg fin
218 41 00000148 8B0D[0A000000] mov ecx,[B]
219 42 0000014E 890D[00000000] mov [max],ecx
220 43 ; ----- Вывод результата
221 44 fin:
222 45 00000154 B8[13000000] mov eax,msg2
223 46 00000159 E8B1FFFFFF call sprintf
224 47 0000015E A1[00000000] mov eax,[max]
225 48 00000163 E81EFFFFFF call iprintLF
226 49 00000168 F86FFFFFFF call quit

```

Рис. 2.11: Файл листинга с ошибкой lab7-2

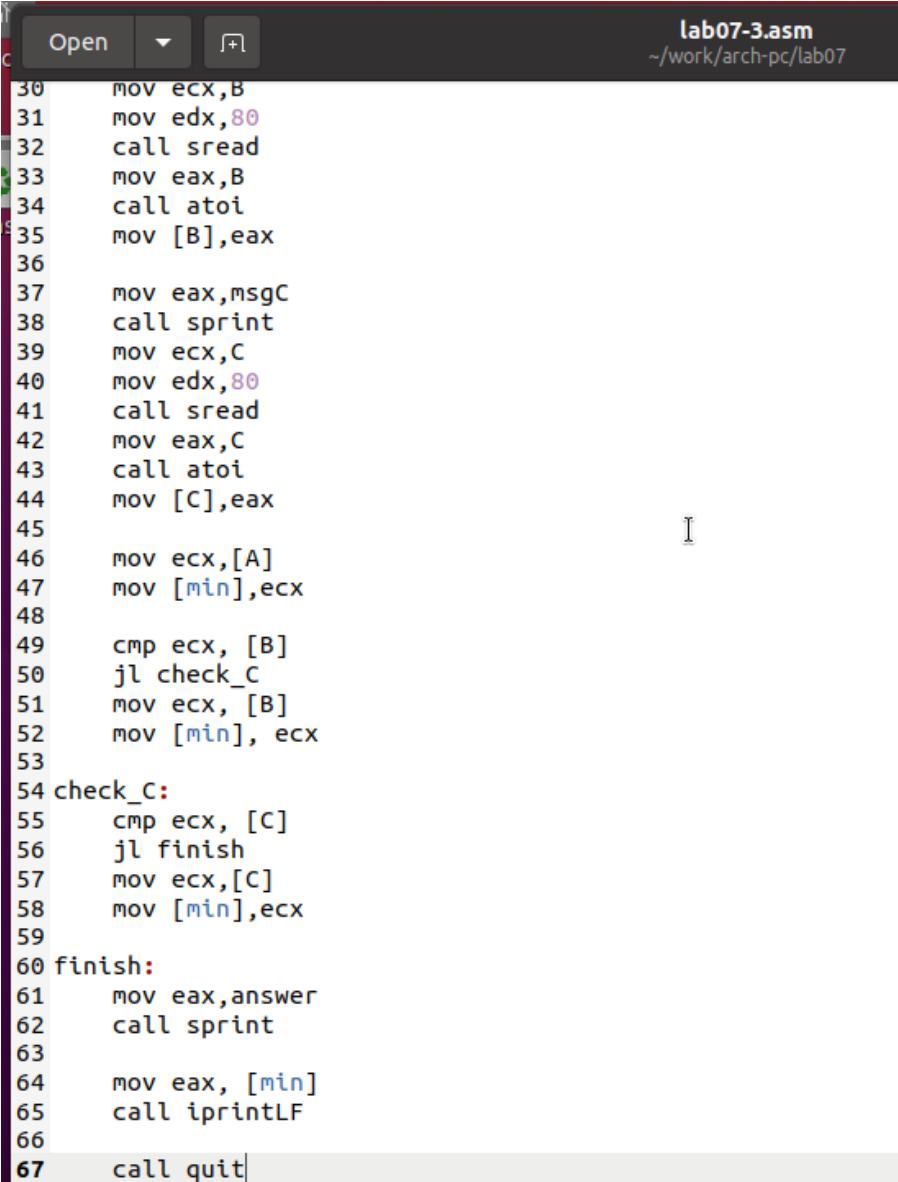
В итоге из-за синтаксической ошибки не удалось сгенерировать объектный файл, но был получен листинг программы, где было указано место возникновения ошибки.

## 2.1 Задание для самостоятельной работы

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



Мой вариант 4 - числа: 8,88,68



```
lab07-3.asm
~/work/arch-pc/lab07

30  mov ecx,B
31  mov edx,80
32  call sread
33  mov eax,B
34  call atoi
35  mov [B],eax
36
37  mov eax,msgC
38  call sprintf
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 sprintf
63
64  mov eax, [min]
65  call iprintLF
66
67  call quit
```

Рис. 2.12: Код программы lab7-3.asm

```
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-3.asm
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$ ld -m elf_i386 lab07-3.o -o lab07-3
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-3
Input A: 8
Input B: 88
Input C: 68
Smallest: 8
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$
```

Рис. 2.13: Компиляция и запуск 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.

Мой вариант 4

$$\begin{cases} 2x + a, a \neq 0 \\ 2x + 1, a = 0 \end{cases}$$

```

15     mov eax,msgA
16     call sprint
17     mov ecx,A
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
33     mov edx, 0
34     mov ebx, [A]
35     cmp ebx, edx
36     jne first
37     jmp second
38
39 first:
40     mov eax,[X]
41     mov ebx, 2
42     mul ebx
43     add eax,[A]
44     call iprintLF
45     call quit
46 second:
47     mov eax,[X]
48     mov ebx,2
49     mul ebx
50     add eax,1
51     call iprintLF
52     call quit

```

Рис. 2.14: Код программы lab7-4.asm

При  $x = 3, a = 0 f(x) = 7$

При  $x = 3, a = 2 f(x) = 8$

```
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$ nasm -f elf lab07-4.asm
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$ ld -m elf_i386 lab07-4.o -o lab07-4
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-4
Input A: 0
Input X: 3
7
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$ ./lab07-4
Input A: 2
Input X: 3
8
huleroyun@Huler-Ubuntu:~/work/arch-pc/lab07$
```

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

## 3 Выводы

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