

Отчёт по лабораторной работе 7

дисциплина: Архитектура компьютеров

Байрамов Керим Сапарович

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Реализация переходов в NASM	6
2.2	Изучение структуры файла листинга	12
2.3	Самостоятельное задание	15
3	Выводы	20

Список иллюстраций

2.1	Создан каталог	6
2.2	Программа lab7-1.asm	7
2.3	Запуск программы lab7-1.asm	8
2.4	Программа lab7-1.asm	8
2.5	Запуск программы lab7-1.asm	9
2.6	Программа lab7-1.asm	10
2.7	Запуск программы lab7-1.asm	10
2.8	Программа lab7-2.asm	11
2.9	Запуск программы lab7-2.asm	12
2.10	Файл листинга lab7-2	13
2.11	Ошибка трансляции lab7-2	14
2.12	Файл листинга с ошибкой lab7-2	15
2.13	Программа lab7-task1.asm	16
2.14	Запуск программы lab7-task1.asm	17
2.15	Программа lab7-task2.asm	18
2.16	Запуск программы lab7-task2.asm	19

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

2.1 Реализация переходов в NASM

Создал каталог для программ лабораторной работы № 7 и файл lab7-1.asm (рис. 2.1).

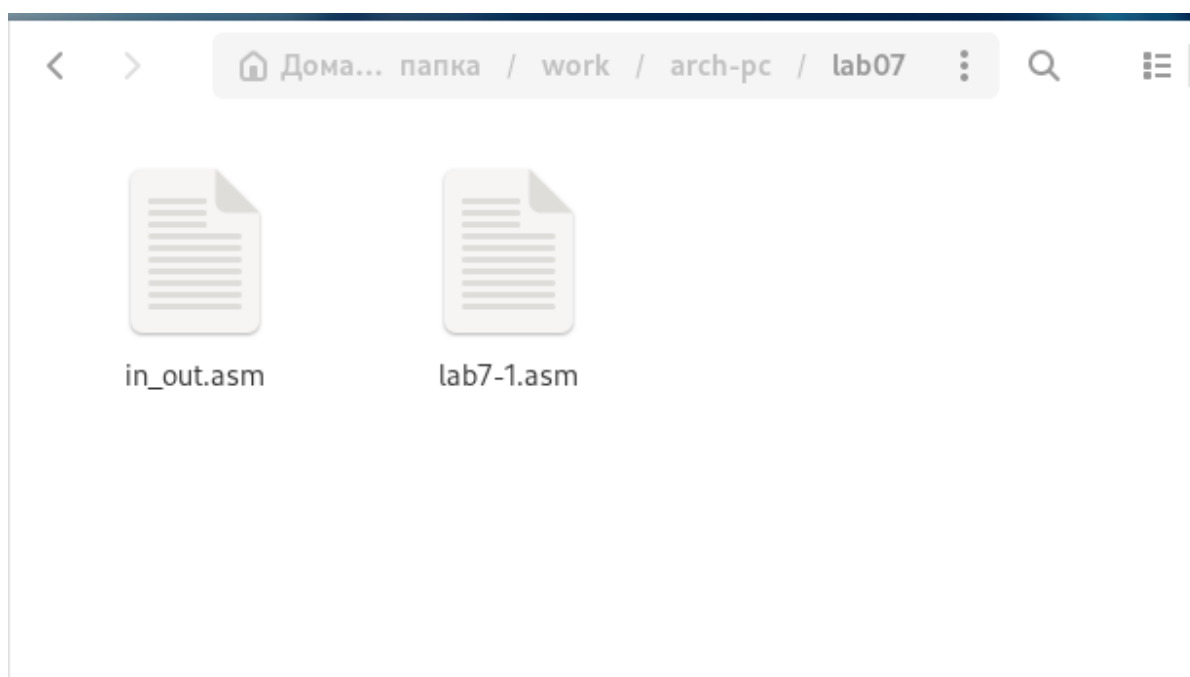
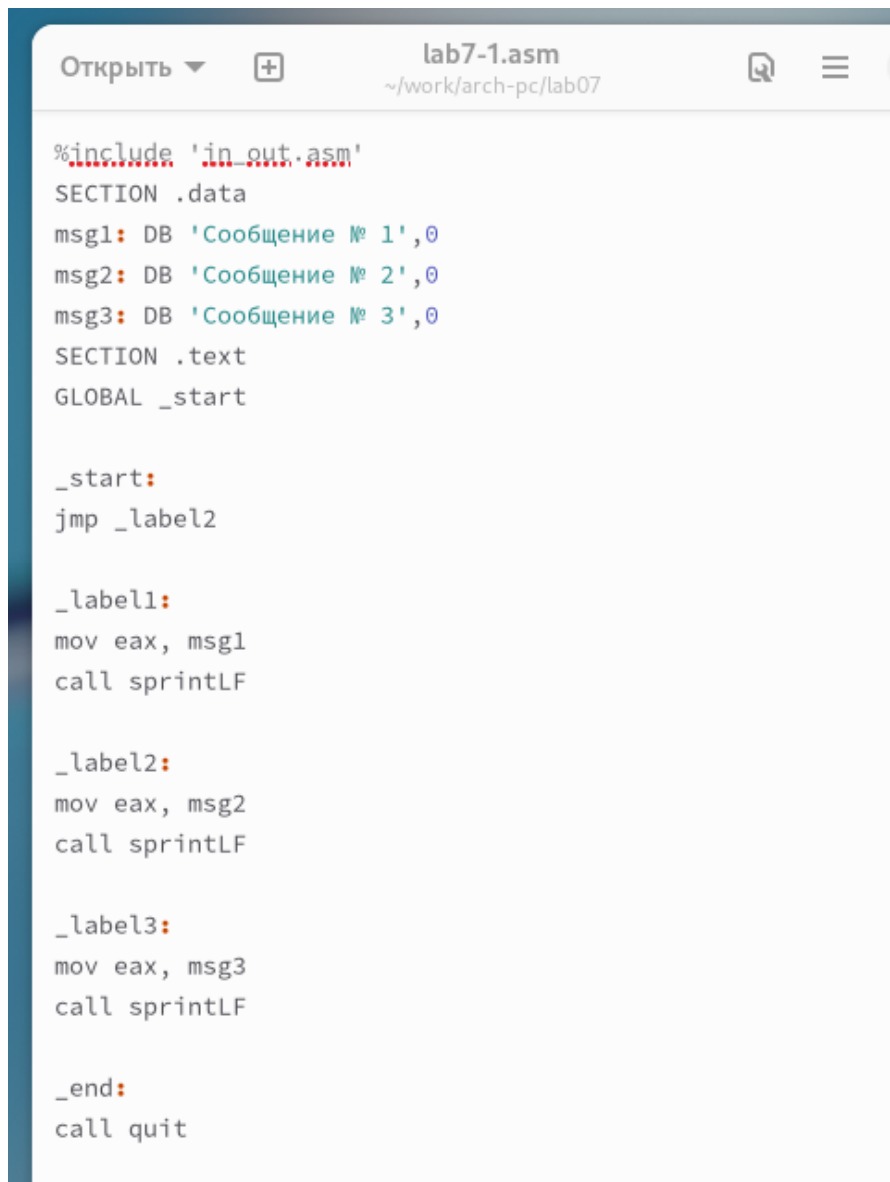


Рис. 2.1: Создан каталог

В NASM инструкция `jmp` используется для реализации безусловных переходов. Рассмотрим пример программы с использованием инструкции `jmp`. В файле `lab7-1.asm` разместил текст программы из листинга 7.1 (рис. 2.2).



```
%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. 2.2: Программа lab7-1.asm

Создал исполняемый файл и запустил его (рис. 2.3).

```
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ █
```

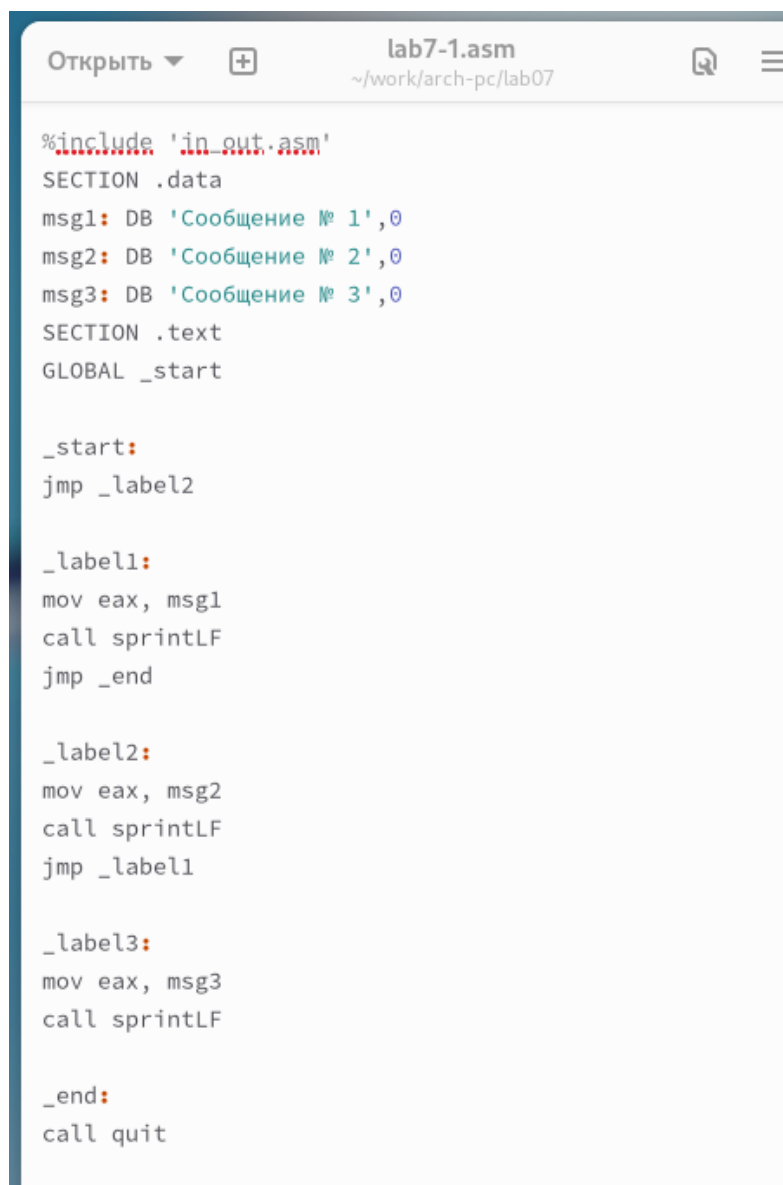
Рис. 2.3: Запуск программы lab7-1.asm

Инструкция `jmp` позволяет выполнять переходы как вперёд, так и назад. Изменил программу так, чтобы сначала выводилось сообщение № 2, затем сообщение № 1, после чего программа завершала работу. Для этого добавил в текст программы инструкцию `jmp` с меткой `_label1` после вывода сообщения № 2 (чтобы перейти к инструкции вывода сообщения № 1) и инструкцию `jmp` с меткой `_end` после вывода сообщения № 1 (для перехода к инструкции `call quit`).

Обновил текст программы согласно листингу 7.2 (рис. 2.4 и 2.5).

```
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ █
```

Рис. 2.4: Программа lab7-1.asm



```
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

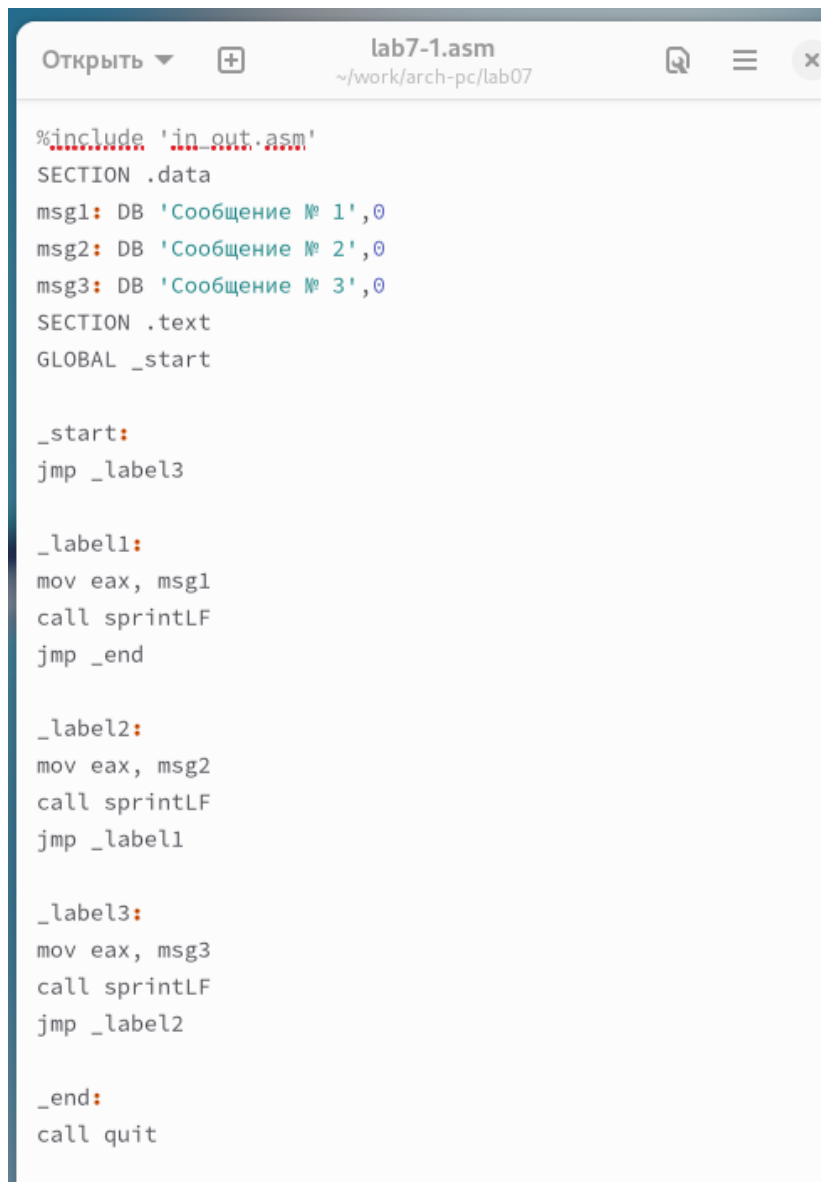
_label3:
mov eax, msg3
call sprintLF

_end:
call quit
```

Рис. 2.5: Запуск программы lab7-1.asm

Изменил текст программы так, чтобы итоговый вывод программы выглядел следующим образом (рис. 2.6 и 2.7):

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

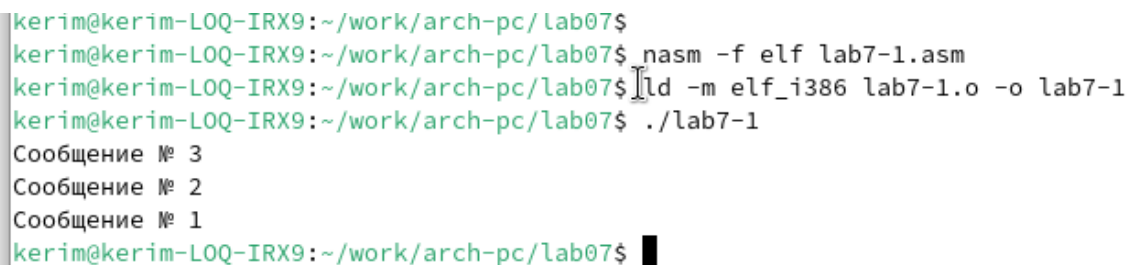
_label1:
mov eax, msg1
call sprintLF
jmp _end

_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

Рис. 2.6: Программа lab7-1.asm

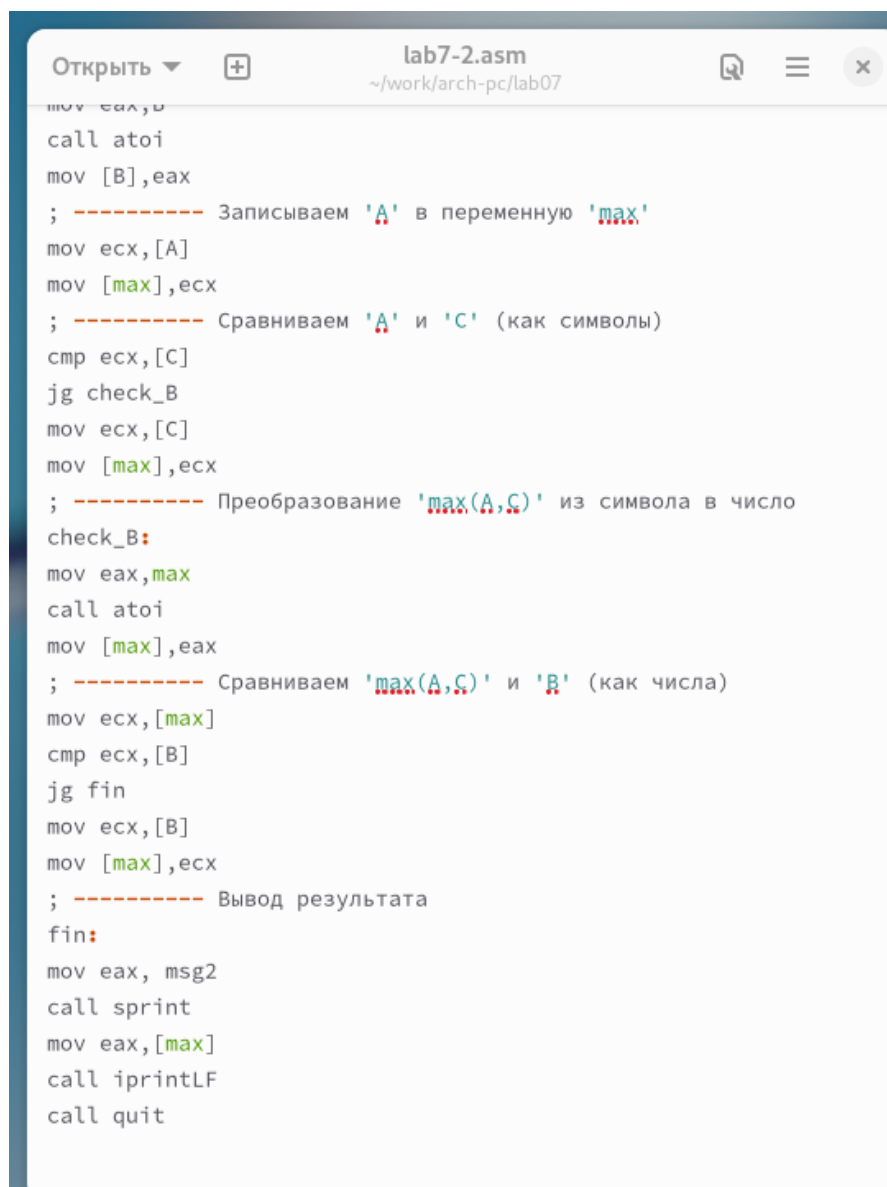


```
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$
```

Рис. 2.7: Запуск программы lab7-1.asm

Инструкция `jmp` всегда вызывает переход. Однако часто в программировании требуются условные переходы, которые выполняются только при соблюдении определённых условий. В качестве примера рассмотрим программу, определяющую и выводящую наибольшее значение среди трёх целочисленных переменных А, В и С. Значения для А и С заданы в программе, а В вводится с клавиатуры.

Создал исполняемый файл и проверил его работу для различных значений В (рис. 2.8 и 2.9).



```
Открыть ▾ + lab7-2.asm
~/work/arch-pc/lab07

mov eax, 0
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 sprintf
mov eax, [max]
call iprintLF
call quit
```

Рис. 2.8: Программа lab7-2.asm

```
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-2.o -o lab7-2  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 75  
Наибольшее число: 75  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 60  
Наибольшее число: 60  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 30  
Наибольшее число: 50  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$
```

Рис. 2.9: Запуск программы lab7-2.asm

2.2 Изучение структуры файла листинга

Обычно NASM создаёт только объектный файл. Чтобы получить файл листинга, нужно указать ключ `-l` и задать имя файла листинга в командной строке.

Создал файл листинга для программы из файла lab7-2.asm (рис. 2.10).

```

Открыть ▾ + lab7-2.lst ~/work/arch-pc/lab07
72 <1>
73 <1> ;----- iprint -----
74 <1> ; Функция вывода на экран чисел в формате ASCII
75 <1> ; входные данные: mov eax,<int>
76 <1> iprint:
77 00000054 50 <1> push eax
78 00000055 51 <1> push ecx
79 00000056 52 <1> push edx
80 00000057 56 <1> push esi
81 00000058 B900000000 <1> mov ecx, 0
82 <1>
83 <1> divideLoop:
84 0000005D 41 <1> inc ecx
85 0000005E BA00000000 <1> mov edx, 0
86 00000063 BF0A000000 <1> mov esi, 10
87 00000068 F7FF <1> idiv esi
88 0000006A 83C230 <1> add edx, 48
89 0000006D 52 <1> push edx
90 0000006E 83F800 <1> cmp eax, 0
91 00000071 75EA <1> jnz divideLoop
92 <1>
93 <1> printLoop:
94 00000073 49 <1> dec ecx
95 00000074 89E0 <1> mov eax, esp
96 00000076 F894FFFFFF <1> call sprintf
97 0000007B 58 <1> pop eax
98 0000007C 83F900 <1> cmp ecx, 0
99 0000007F 75E2 <1> jnz printLoop
100 <1>
101 00000081 5E <1> pop esi

```

Рис. 2.10: Файл листинга lab7-2

Рассмотрим его структуру:

- **Строка 211**

- 34 — номер строки
- 0000012E — адрес
- B8[00000000] — машинный код
- mov eax, max — код программы

- **Строка 212**

- 35 — номер строки

- 00000133 — адрес
- E864FFFFFF — машинный код
- call atoi — код программы

• Строка 213

- 36 — номер строки
- 00000138 — адрес
- A3[00000000] — машинный код
- mov [max], eax — код программы

Открыл файл lab7-2.asm, удалил один из операндов в инструкции с двумя операндами и выполнил трансляцию с получением файла листинга (рис. 2.11 и 2.12).

```

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

23 0000010B A3[0A000000]    mov [R],eax
24                                ; ----- Записываем 'A' в переменную 'max'
25 00000110 8B0D[35000000]    mov ecx,[A]
26 00000116 890D[00000000]    mov [max],ecx
27                                ; ----- Сравниваем 'A' и 'C' (как символы)
28 0000011C 3B0D[39000000]    cmp ecx,[C]
29 00000122 7F0C                                ig check_R
30 00000124 8B0D[39000000]    mov ecx,[C]
31 0000012A 890D[00000000]    mov [max],ecx
32                                ; ----- Преобразование 'max(A,C)' из символа в число
33                                check_R:
34                                mov eax,
35                                *****
36                                error: invalid combination of opcode and operands
37                                call atoi
38                                mov [max],eax
39                                ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
40                                mov ecx,[max]
41                                cmp ecx,[B]
42                                ig fin
43                                mov ecx,[B]
44                                mov [max],ecx
45                                ; ----- Вывод результата
46                                fin:
47                                mov eax, msg2
48                                call sprint
49                                mov eax,[max]
50                                call iprintf
51                                call quit

```

Рис. 2.11: Ошибка трансляции lab7-2

```
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm -l lab7-2.lst  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$  
kerim@kerim-LOQ-IRX9:~/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  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$  
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$
```

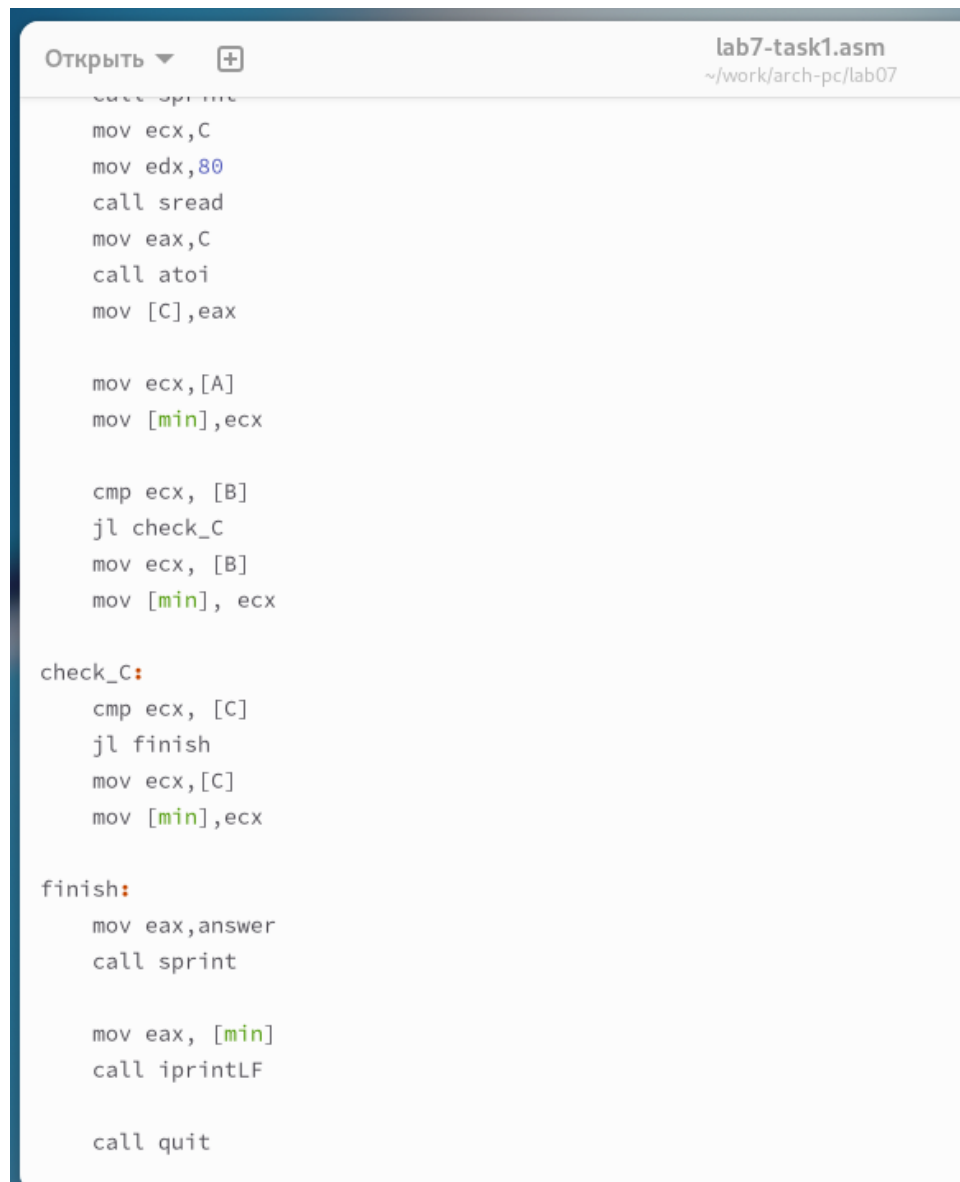
Рис. 2.12: Файл листинга с ошибкой lab7-2

Из-за ошибки объектный файл не был создан. Однако, листинг указал местоположение ошибки.

2.3 Самостоятельное задание

1. Найти наименьшее среди трёх целочисленных переменных a , b и c , используя значения из таблицы 7.5 для варианта, полученного при выполнении лабораторной работы № 6. Создать исполняемый файл и проверить его работу (рис. 2.13 и 2.14).

Для варианта 7: $a = 45$, $b = 67$, $c = 15$.



```
Открыть + lab7-task1.asm
~/.work/arch-pc/lab07

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. 2.13: Программа lab7-task1.asm


```

kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ nasm -f elf lab7-task1.asm
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-task1.o -o lab7-task1
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$ ./lab7-task1
Input A: 45
Input B: 67
Input C: 15
Smallest: 15
kerim@kerim-LOQ-IRX9:~/work/arch-pc/lab07$

```

Рис. 2.14: Запуск программы lab7-task1.asm

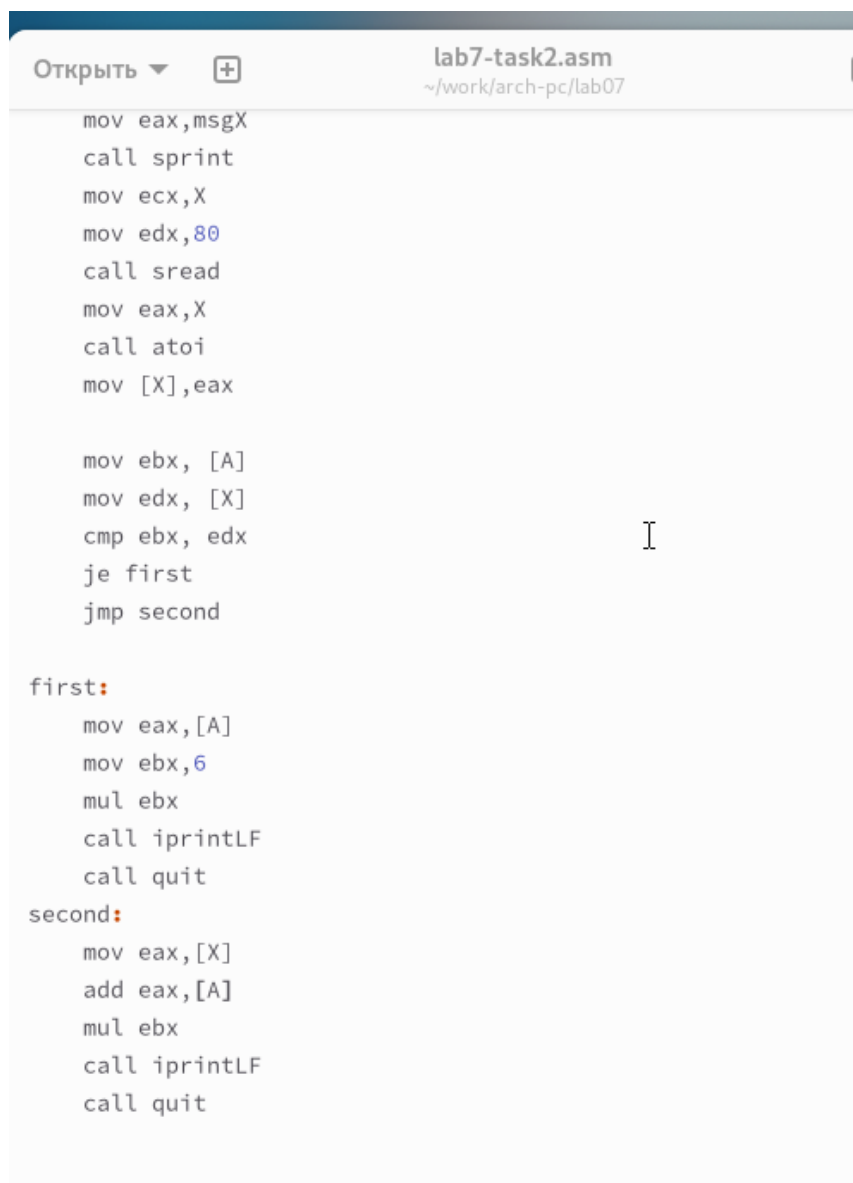
2. **Программа для вычисления функции $f(x)$** при введенных значениях x и a с клавиатуры. Вид функции $f(x)$ выбирается из таблицы 7.6 в зависимости от варианта, полученного для лабораторной работы № 7. Создать исполняемый файл и проверить его работу для значений x и a из таблицы 7.6 (рис. 2.15 и 2.16).

Для варианта 7:

$$f(x) = \begin{cases} 6a, & x = a \\ a + x, & x \neq a \end{cases}$$

При $x = 1, a = 1$ результат — 6.

При $x = 2, a = 1$ результат — 3.



```
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, [X]
cmp ebx, edx
je first
jmp second

first:
mov eax,[A]
mov ebx,6
mul ebx
call iprintLF
call quit
second:
mov eax,[X]
add eax,[A]
mul ebx
call iprintLF
call quit
```

Рис. 2.15: Программа lab7-task2.asm

```
kerim@kerim-L0Q-IRX9:~/work/arch-pc/lab07$  
kerim@kerim-L0Q-IRX9:~/work/arch-pc/lab07$ nasm -f elf lab7-task2.asm  
kerim@kerim-L0Q-IRX9:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-task2.o -o lab7-task2  
kerim@kerim-L0Q-IRX9:~/work/arch-pc/lab07$ ./lab7-task2  
Input A: 1  
Input X: 1  
6  
kerim@kerim-L0Q-IRX9:~/work/arch-pc/lab07$ ./lab7-task2  
Input A: 1  
Input X: 2  
3  
kerim@kerim-L0Q-IRX9:~/work/arch-pc/lab07$
```

Рис. 2.16: Запуск программы lab7-task2.asm

3 Выводы

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