

Отчёт по лабораторной работе 7

Архитектура компьютера

Шищенко Ярослав Викторович НКАбд-05-24

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	20

Список иллюстраций

2.1	Программа в файле lab7-1.asm	7
2.2	Запуск программы lab7-1.asm	7
2.3	Программа в файле lab7-1.asm:	8
2.4	Запуск программы lab7-1.asm:	9
2.5	Программа в файле lab7-1.asm	10
2.6	Запуск программы lab7-1.asm	10
2.7	Программа в файле lab7-2.asm	12
2.8	Запуск программы lab7-2.asm	12
2.9	Файл листинга lab7-2	13
2.10	Ошибка трансляции lab7-2	14
2.11	Файл листинга с ошибкой lab7-2	15
2.12	Программа в файле task.asm	16
2.13	Запуск программы task.asm	16
2.14	Программа в файле task2.asm	18
2.15	Запуск программы task2.asm	19

Список таблиц

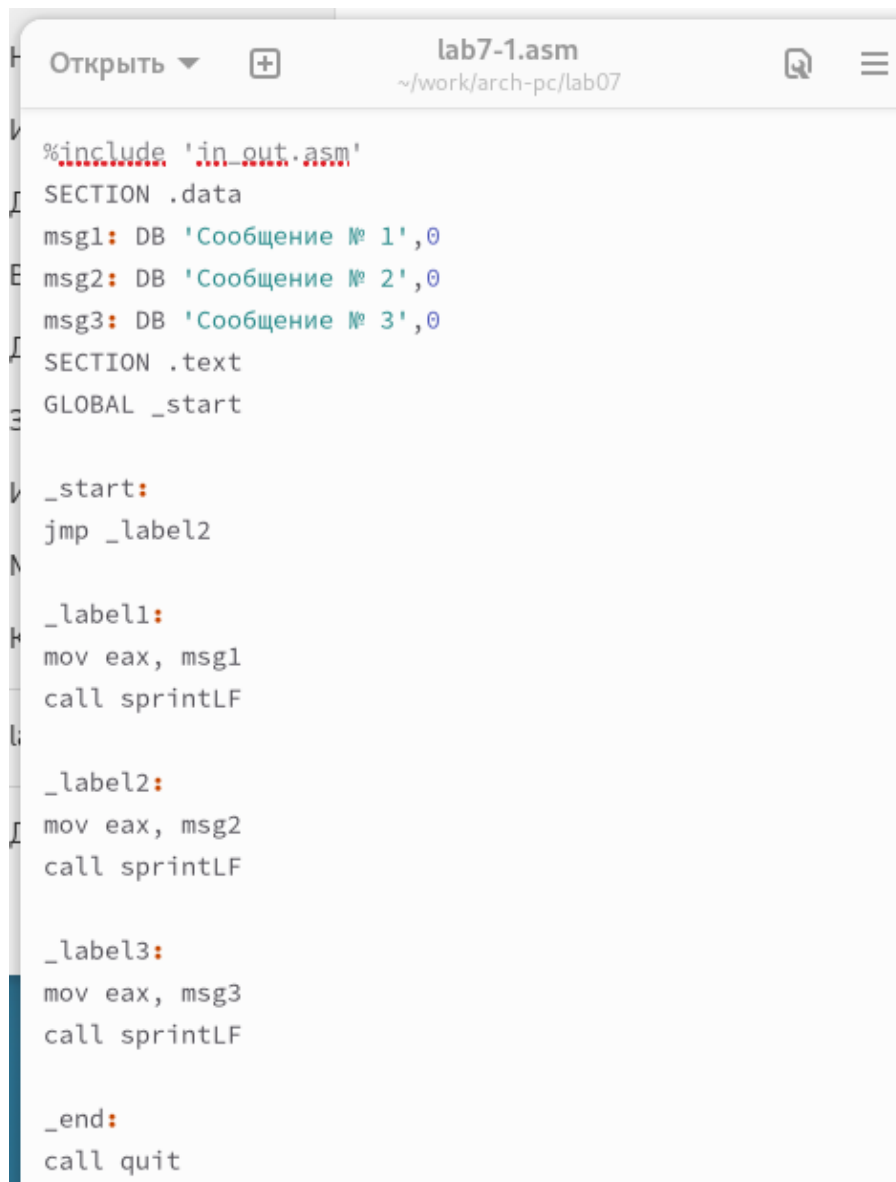
1 Цель работы

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

1. Создал каталог для программ 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.



```
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

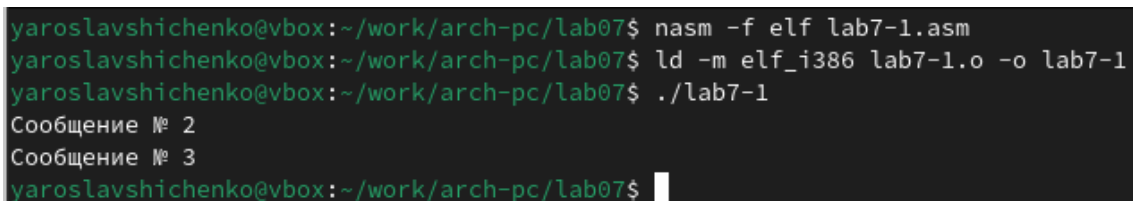
_label2:
mov eax, msg2
call sprintLF

_label3:
mov eax, msg3
call sprintLF

_end:
call quit
```

Рис. 2.1: Программа в файле lab7-1.asm

Создал исполняемый файл и запустил его.



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



```
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 sprintf
jmp _end

_label2:
mov eax, msg2
call sprintf
jmp _label1

_label3:
mov eax, msg3
call sprintf

_end:
call quit
```

Рис. 2.3: Программа в файле lab7-1.asm:


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

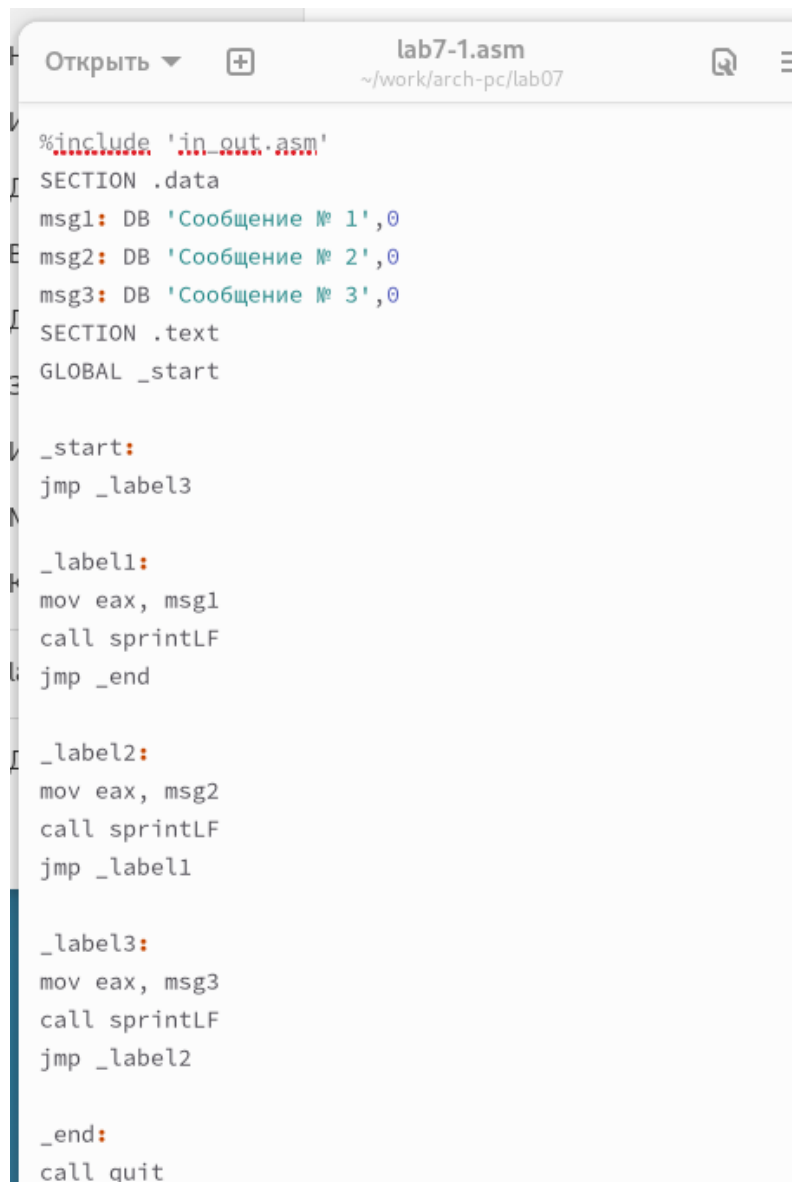
Рис. 2.4: Запуск программы lab7-1.asm:

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

Сообщение № 3

Сообщение № 2

Сообщение № 1



```
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

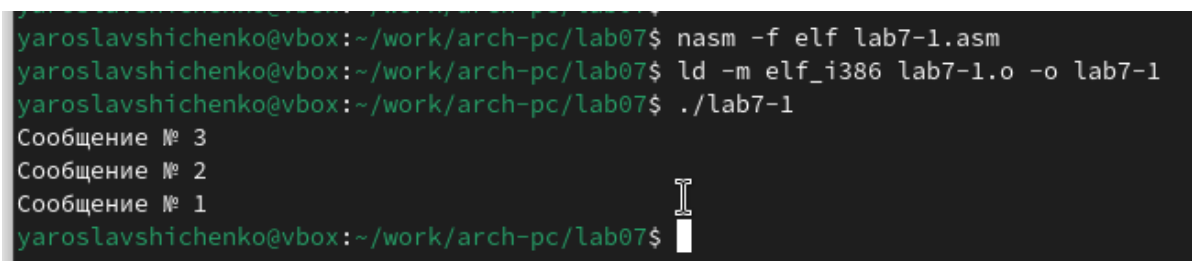
_label1:
mov eax, msg1
call sprintLF
jmp _end

_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

Рис. 2.5: Программа в файле lab7-1.asm

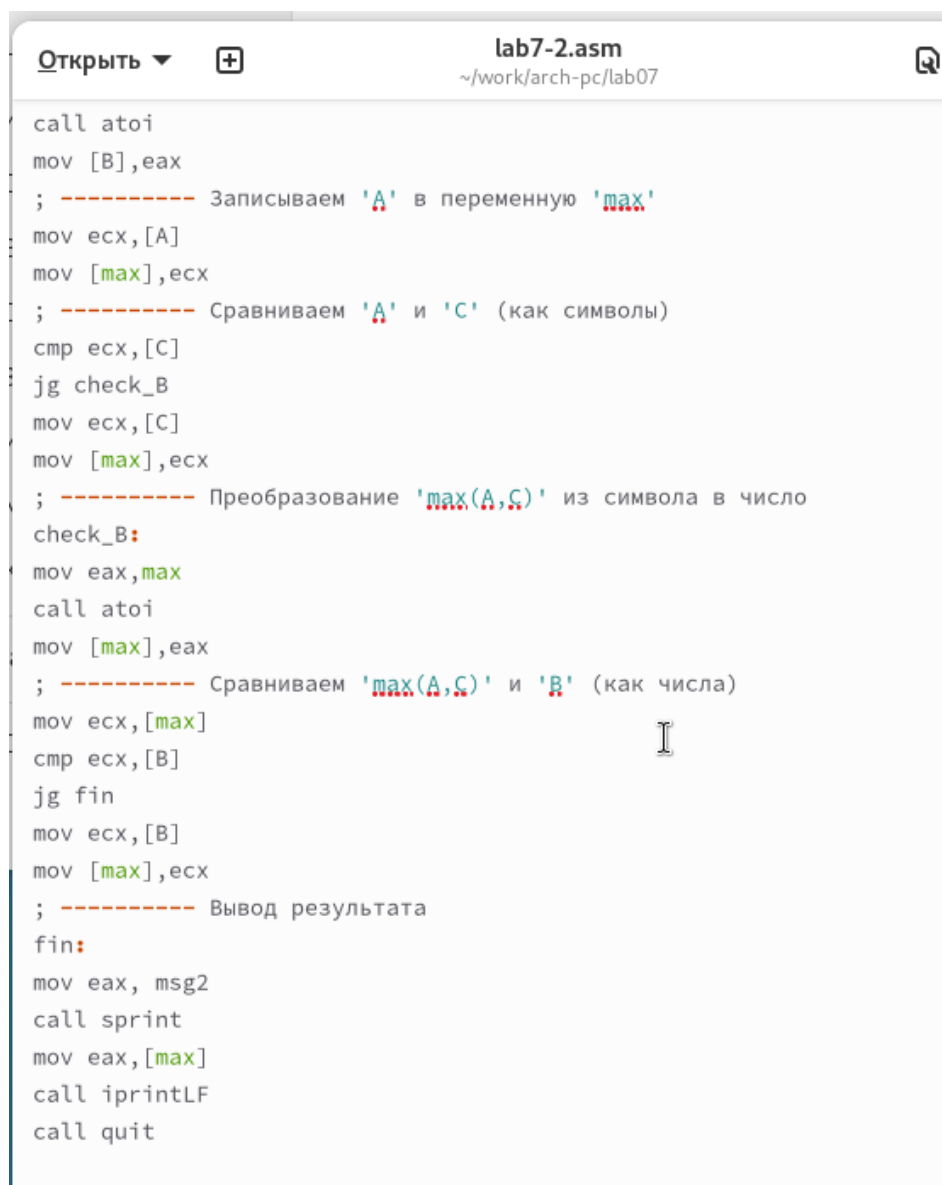


```
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
yaroslavshichenko@vbox:~/work/arch-pc/lab07$
```

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.

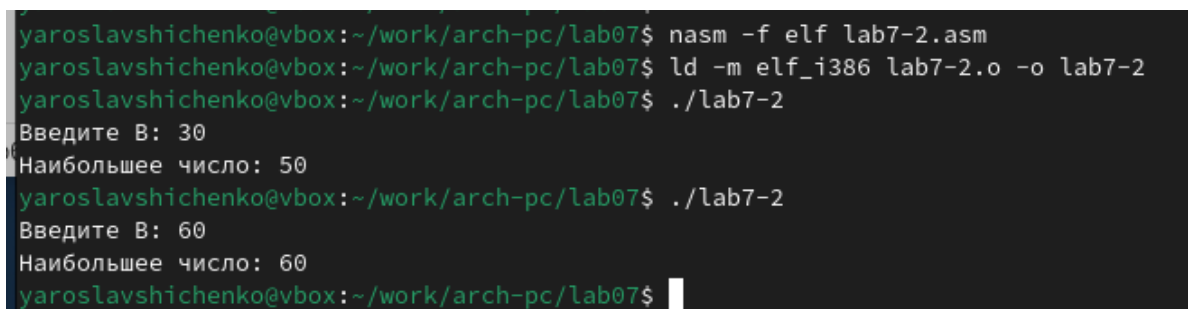
Создал исполняемый файл и проверил его работу для разных значений В.



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

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 в файле lab7-2.asm



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

```
196 21 00000101 B8[0A000000]      mov eax,B
197 22 00000106 F891FFFFFF      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      ig 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 F862FFFFFF      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      ig 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 F8ACFFFFFF      call sprint
222 47 00000163 A1[00000000]      mov eax,[max]
223 48 00000168 F819FFFFFF      call iprintf
224 49 0000016D F869FFFFFF      call quit
```

Рис. 2.9: Файл листинга lab7-2

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

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

```

yaroslavshichenko@vbox: ~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm -l lab7-2.lst
yaroslavshichenko@vbox: ~/work/arch-pc/lab07$
yaroslavshichenko@vbox: ~/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
yaroslavshichenko@vbox: ~/work/arch-pc/lab07$

```

Рис. 2.10: Ошибка трансляции lab7-2

```

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 ig 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 F867FFFFFF 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 ig 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 F8B1FFFFFF call sprint
223 47 0000015E A1[00000000] mov sax,[max]
224 48 00000163 F81FFFFFFF call iprintLF
225 49 00000168 F86FFFFFFF call quit

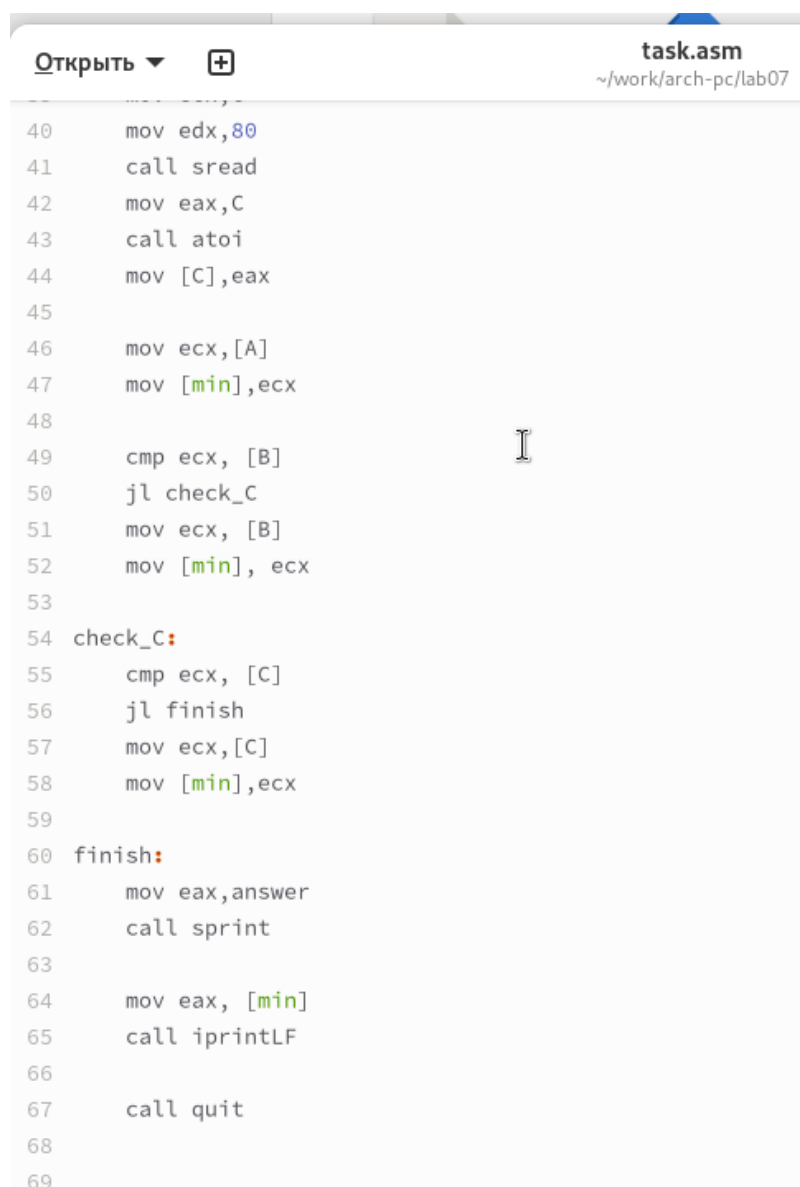
```

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

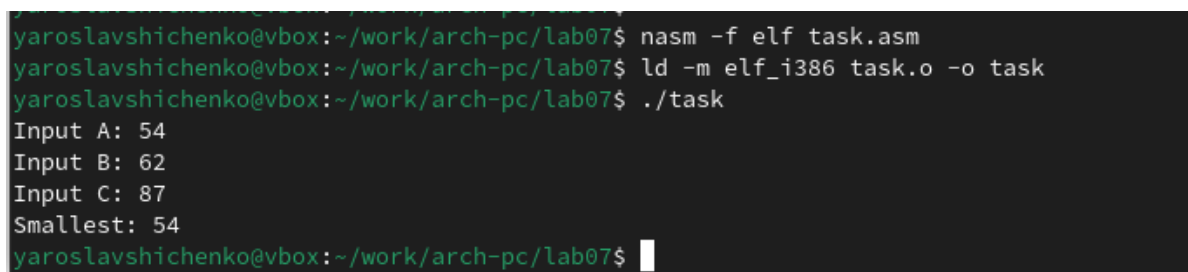
для варианта 5 - 54,62,87



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

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
68
69
```

Рис. 2.12: Программа в файле task.asm



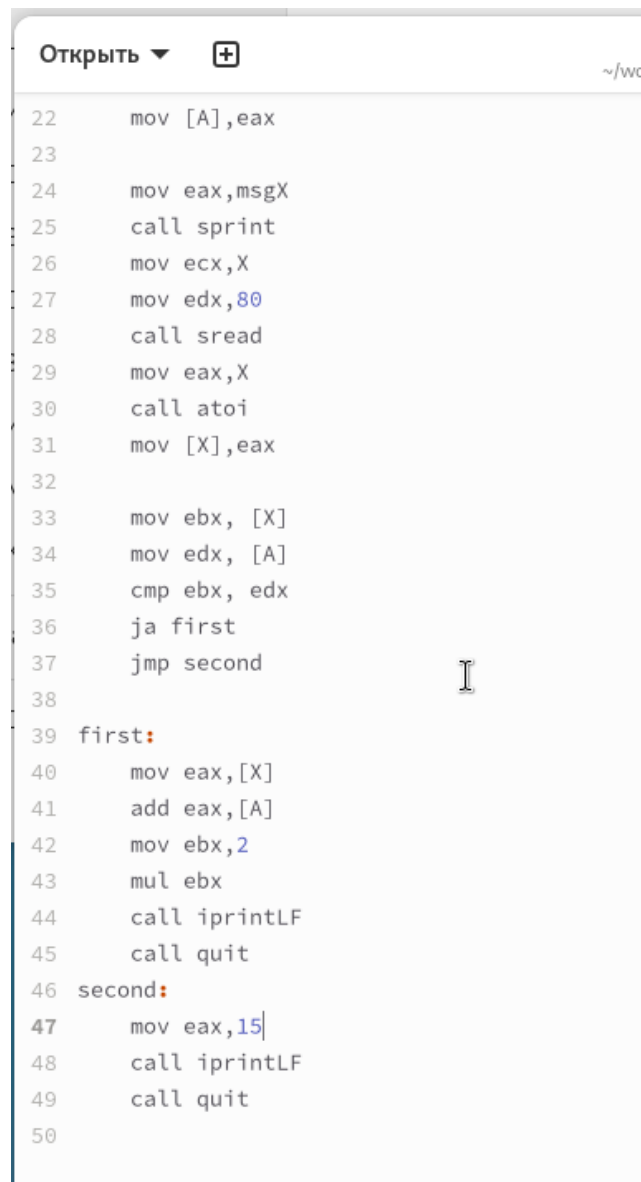
```
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ nasm -f elf task.asm
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 task.o -o task
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ ./task
Input A: 54
Input B: 62
Input C: 87
Smallest: 54
yaroslavshichenko@vbox:~/work/arch-pc/lab07$
```

Рис. 2.13: Запуск программы task.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 5

$$\begin{cases} 2(x - a), x > a \\ 15, x \leq a \end{cases}$$



```
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 ebx, [X]
34    mov edx, [A]
35    cmp ebx, edx
36    ja first
37    jmp second
38
39 first:
40    mov eax,[X]
41    add eax,[A]
42    mov ebx,2
43    mul ebx
44    call iprintLF
45    call quit
46 second:
47    mov eax,15
48    call iprintLF
49    call quit
50
```

Рис. 2.14: Программа в файле task2.asm

```
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ nasm -f elf task2.asm
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ ld -m elf_i386 task2.o -o task2
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ ./task2
Input A: 3
Input X: 2
15
yaroslavshichenko@vbox:~/work/arch-pc/lab07$ ./task2
Input A: 2
Input X: 4
12
yaroslavshichenko@vbox:~/work/arch-pc/lab07$
```

Рис. 2.15: Запуск программы task2.asm

3 Выводы

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