

Отчёт по лабораторной работе 7

дисциплина: Архитектура компьютера

Алиев Руслан Нияз оглы

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Реализация переходов в NASM	6
2.2	Изучение структуры файла листинга	11
2.3	Самостоятельное задание	14
3	Выводы	18

Список иллюстраций

Список таблиц

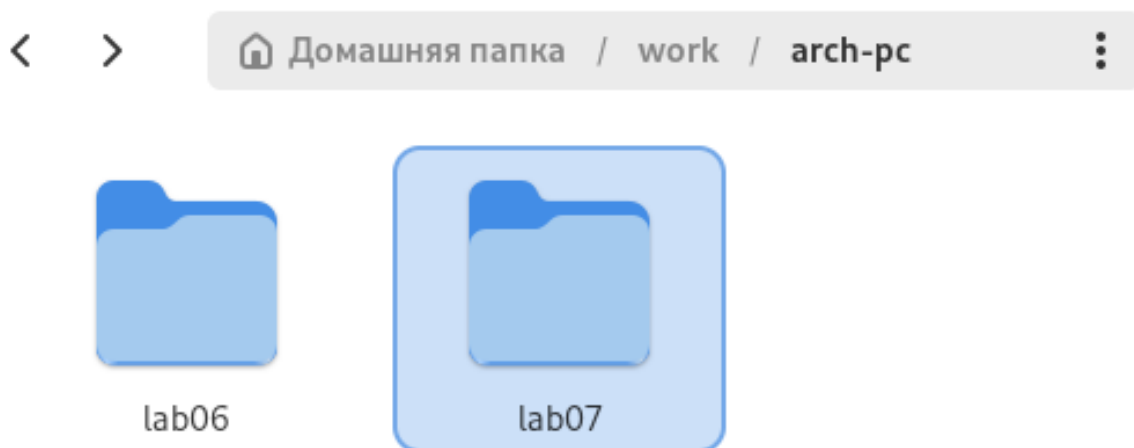
1 Цель работы

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

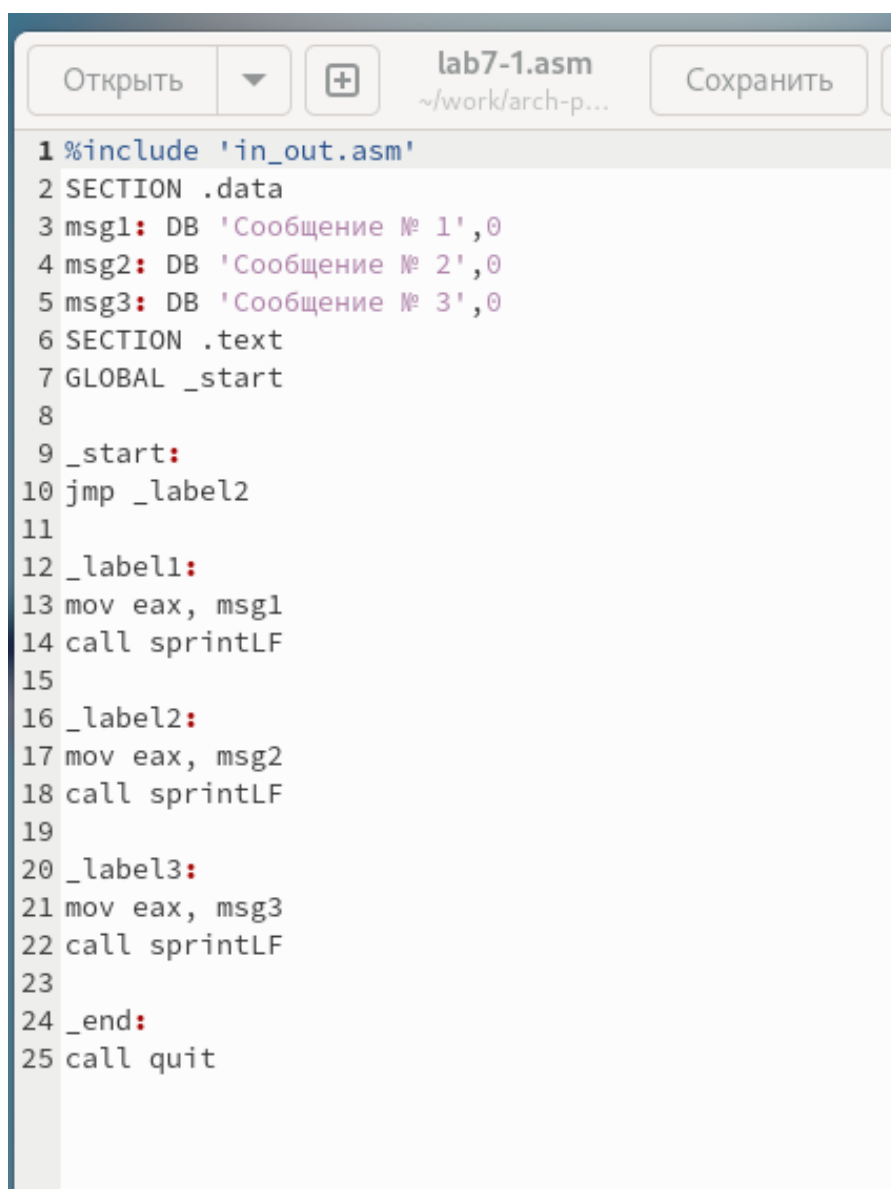
2 Выполнение лабораторной работы

2.1 Реализация переходов в NASM

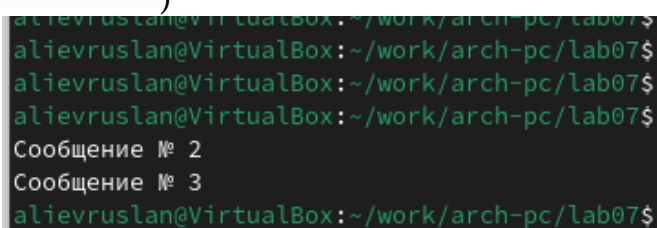
Создаю каталог для программ лабораторной работы № 7 и файл lab7-1.asm. (рис.



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. (рис.



```
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
```



```
alievruslan@VirtualBox: ~/work/arch-pc/lab07$
alievruslan@VirtualBox: ~/work/arch-pc/lab07$
alievruslan@VirtualBox: ~/work/arch-pc/lab07$
alievruslan@VirtualBox: ~/work/arch-pc/lab07$
Сообщение № 2
Сообщение № 3
alievruslan@VirtualBox: ~/work/arch-pc/lab07$
```

Создаю исполняемый файл и запускаю его. (рис.

Инструкция `jmp` позволяет осуществлять переходы не только вперед, но и назад. Изменяю программу таким образом, чтобы она выводила сначала «Сообщение № 2», затем «Сообщение № 1», и завершала работу. Для этого после вывода сообщения № 2 добавляю инструкцию `jmp` с меткой `_label1` (переход к

инструкциям вывода сообщения № 1), и после вывода сообщения № 1 добавляю инструкцию jmp с меткой _end (переход к инструкции call quit).

```
alievruslan@VirtualBox: ~/work
alievruslan@VirtualBox: ~/work
alievruslan@VirtualBox: ~/work
Сообщение № 2
Сообщение № 3
alievruslan@VirtualBox: ~/work
```

Изменяю текст программы в соответствии с листингом 7.2. (рис.

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

(рис.

)

После изменений программа выводит следующее: 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 sprintf
15 jmp _end
16
17 _label2:
18 mov eax, msg2
19 call sprintf
20 jmp _label1
21
22 _label3:
23 mov eax, msg3
24 call sprintf
25 jmp _label2
26
27 _end:
28 call quit
```

(рис.

)

(рис.

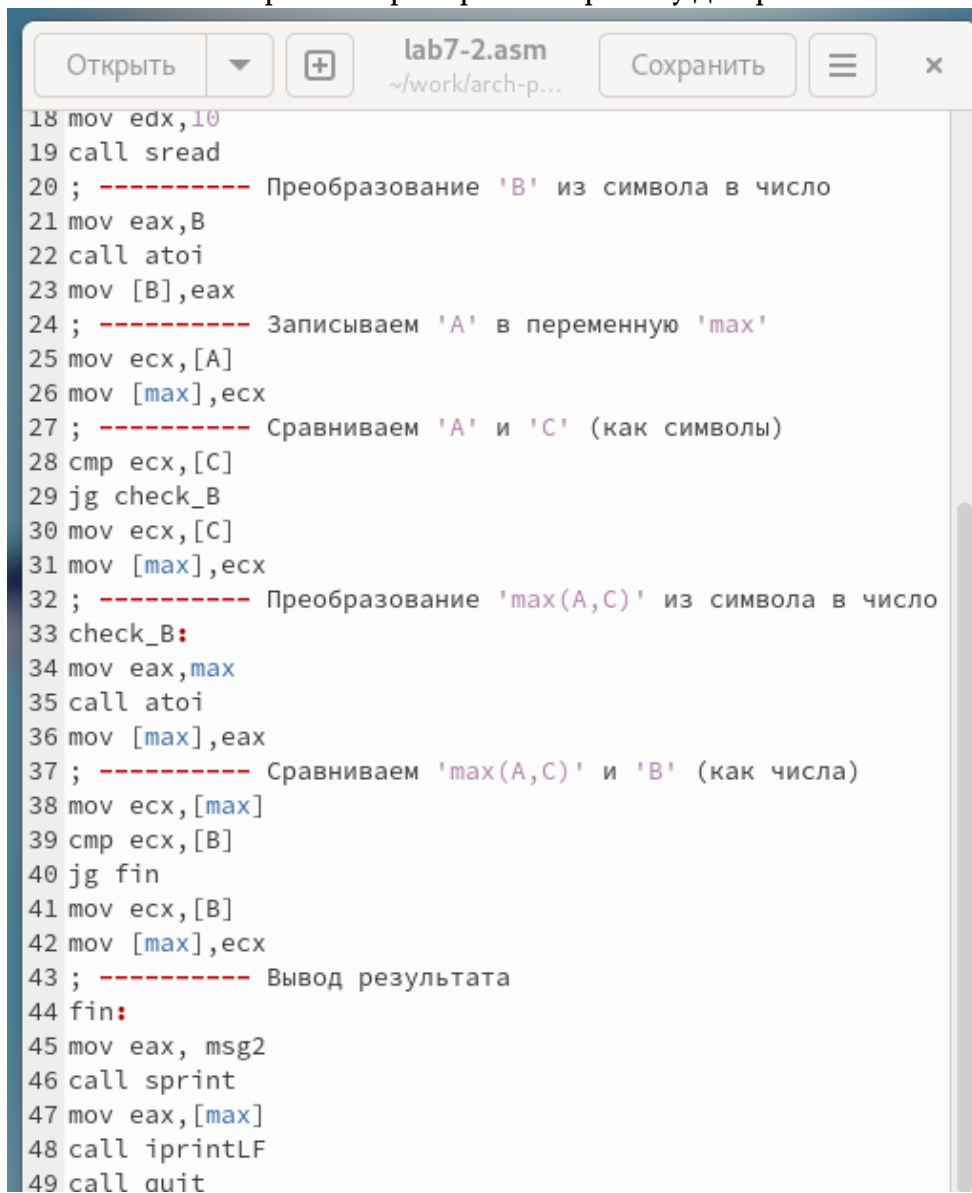
```
alievruslan@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-1.o -o lab7-1
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
alievruslan@VirtualBox:~/work/arch-pc/lab07$
```

)

Использование инструкции `jmp` приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, то есть переход должен осуществляться только при выполнении определенного

условия. В качестве примера рассмотрим программу, которая определяет и выводит на экран наибольшее из трех целочисленных переменных: А, В и С. Значения для А и С задаются в программе, значение В вводится с клавиатуры.

Создаю исполняемый файл и проверяю его работу для различных значений



```
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
```

В (рис.

) (рис.

```
alievruslan@VirtualBox:~/work/arch-pc/lab07$  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-2.asm  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-2.o -o lab7-2  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 50  
Наибольшее число: 50  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 55  
Наибольшее число: 55  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ./lab7-2  
Введите В: 6  
Наибольшее число: 50  
alievruslan@VirtualBox:~/work/arch-pc/lab07$
```

2.2 Изучение структуры файла листинга

Обычно `nasm` создает в результате ассемблирования только объектный файл. Чтобы получить файл листинга, необходимо указать ключ `-l` и задать имя файла листинга в командной строке.

lab7-2.asm			
196	21	00000101	B8[0A000000]
197	22	00000106	E891FFFFFF
198	23	0000010B	A3[0A000000]
199	24		
200	25	00000110	8B0D[35000000]
201	26	00000116	890D[00000000]
202	27		
203	28	0000011C	3B0D[39000000]
204	29	00000122	7F0C
205	30	00000124	8B0D[39000000]
206	31	0000012A	890D[00000000]
207	32		
	число		
208	33		
209	34	00000130	B8[00000000]
210	35	00000135	E862FFFFFF
211	36	0000013A	A3[00000000]
212	37		
213	38	0000013F	8B0D[00000000]
214	39	00000145	3B0D[0A000000]
215	40	0000014B	7F0C
216	41	0000014D	8B0D[0A000000]
217	42	00000153	890D[00000000]
218	43		
219	44		
220	45	00000159	B8[13000000]
221	46	0000015E	E8ACFEFFFF
222	47	00000163	A1[00000000]
223	48	00000168	E819FFFFFF
224	49	0000016D	E869FFFFFF

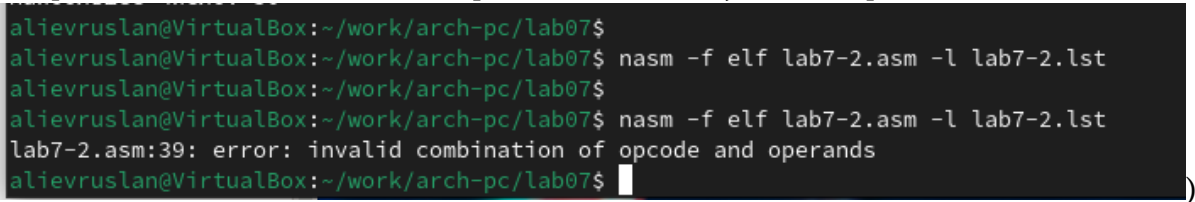
Создаю файл листинга для программы из файла lab7-2.asm. (рис.

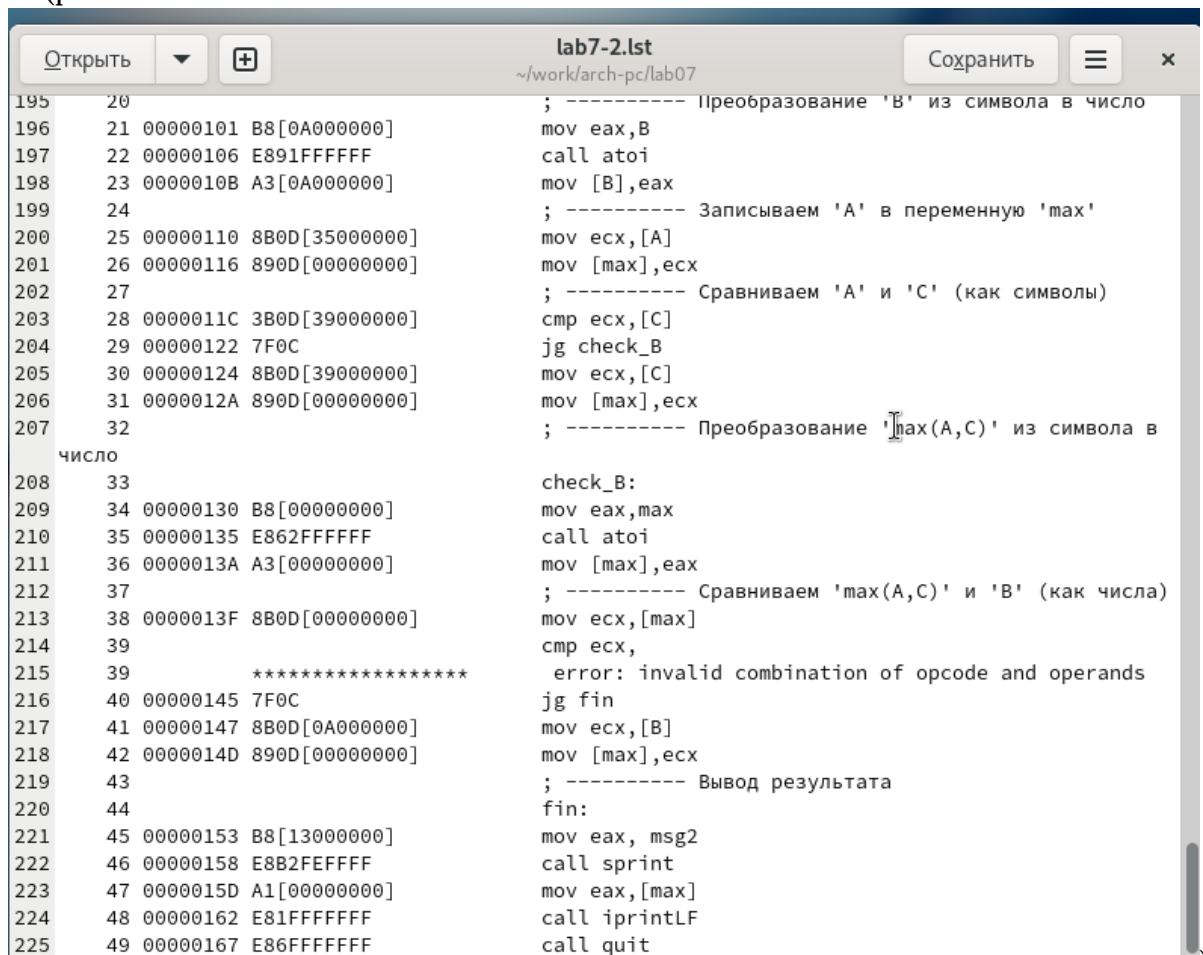
Ознакомимся с его форматом и содержимым.

- 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

Открываю файл с программой lab7-2.asm и удаляю один операнд из инструкции с двумя операндами. Затем выполняю трансляцию с получением файла ли-

стинга. (рис. )



```

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 cmp ecx,
215 39 *****
216 40 00000145 7F0C jg fin
217 41 00000147 8B0D[0A000000] mov ecx,[B]
218 42 0000014D 890D[00000000] mov [max],ecx
219 43 ; ----- Вывод результата
220 44 fin:
221 45 00000153 B8[13000000] mov eax, msg2
222 46 00000158 E8B2FFFFFF call sprint
223 47 0000015D A1[00000000] mov eax,[max]
224 48 00000162 E81FFFFFFF call iprintLF
225 49 00000167 E86FFFFFFF call quit
  
```

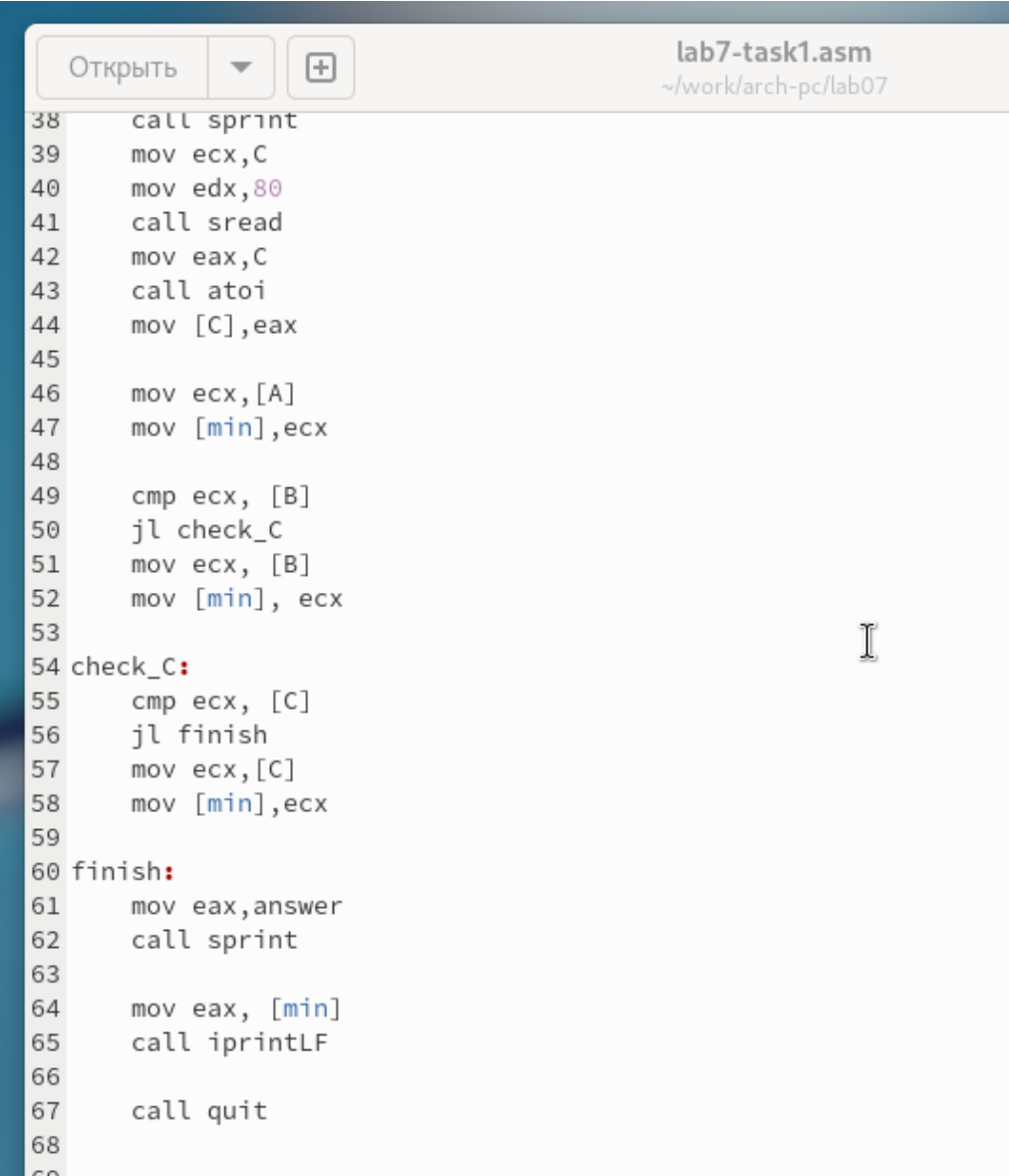
(рис. )

Объектный файл не смог создаться из-за ошибки, но файл листинга с выделен-

ным местом ошибки был получен.

2.3 Самостоятельное задание

Напишите программу нахождения наименьшей из трех целочисленных переменных a, b и c. Значения переменных выбрать из таблицы 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 6. Создаю исполняемый файл и проверяю его работу (рис.



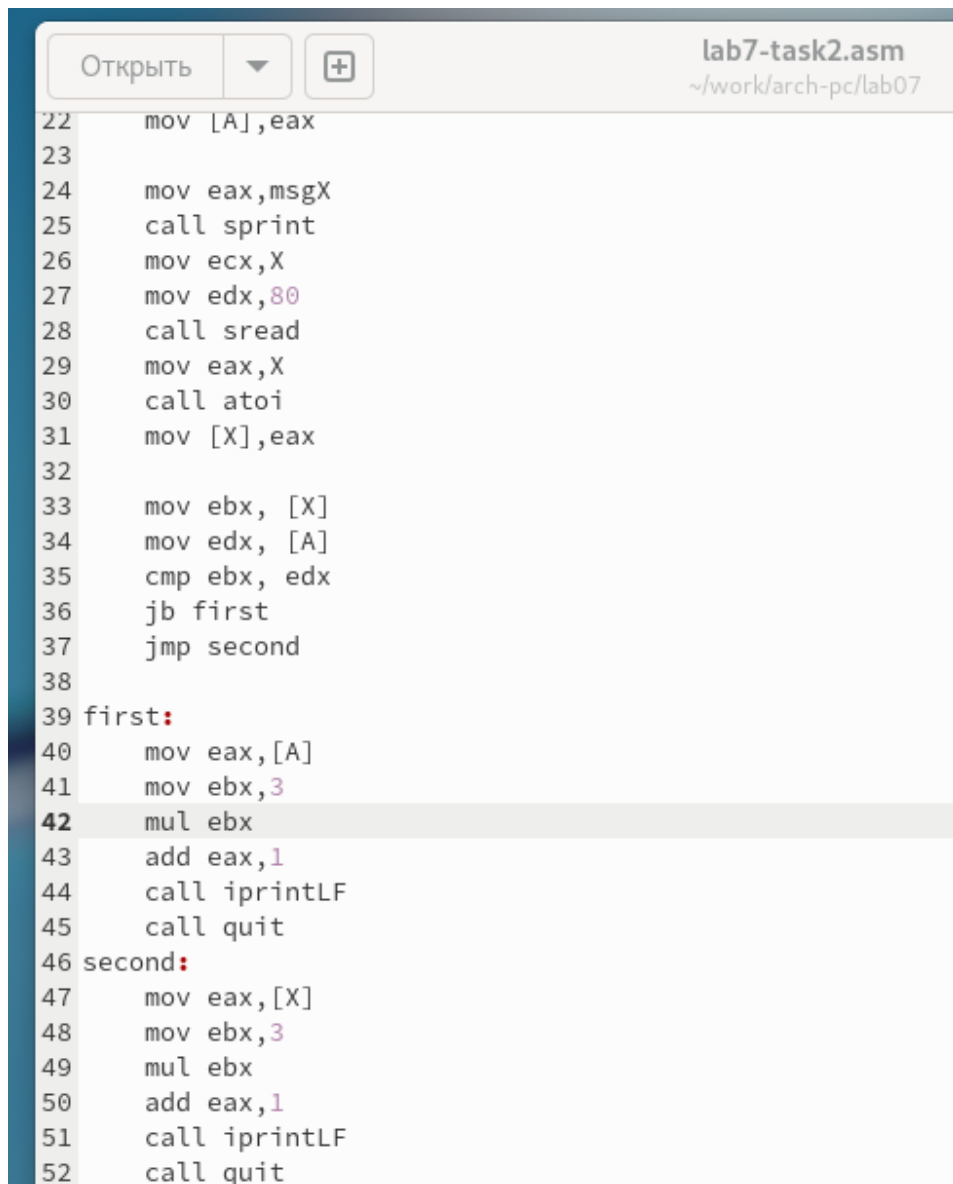
```
38    call sprint
39    mov ecx,C
40    mov edx,80
41    call sread
42    mov eax,C
43    call atoi
44    mov [C],eax
45
46    mov ecx,[A]
47    mov [min],ecx
48
49    cmp ecx, [B]
50    jnl check_C
51    mov ecx, [B]
52    mov [min], ecx
53
54 check_C:
55    cmp ecx, [C]
56    jnl finish
57    mov 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.

```
alievruslan@VirtualBox:~/work/arch-pc/lab07$  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-task1.asm  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-task1.o -o lab7-task1  
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ./lab7-task1  
Input A: 81  
Input B: 22  
Input C: 72  
Smallest: 22  
alievruslan@VirtualBox:~/work/arch-pc/lab07$
```

Для варианта 14 - 81,22,72

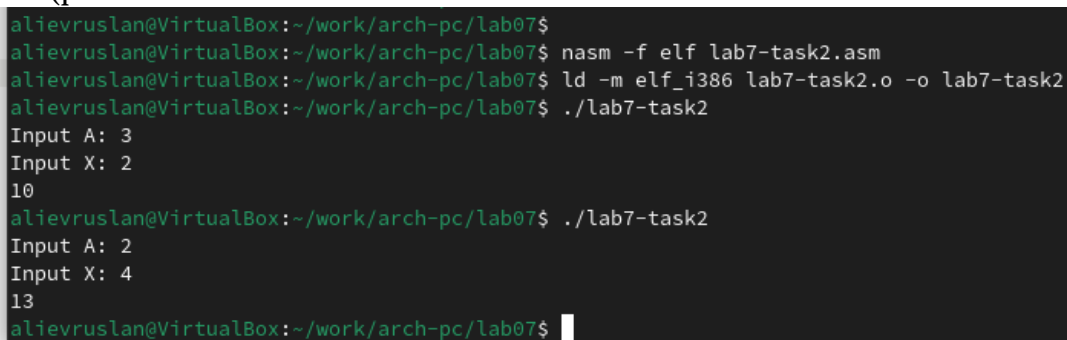
Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создаю исполняемый файл и проверяю его работу для значений X и a из



```
lab7-task2.asm
~/work/arch-pc/lab07

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     jb first
37     jmp second
38
39 first:
40     mov eax,[A]
41     mov ebx,3
42     mul ebx
43     add eax,1
44     call iprintLF
45     call quit
46 second:
47     mov eax,[X]
48     mov ebx,3
49     mul ebx
50     add eax,1
51     call iprintLF
52     call quit
```

7.6. (рис.) (рис.



```
alievruslan@VirtualBox:~/work/arch-pc/lab07$
alievruslan@VirtualBox:~/work/arch-pc/lab07$ nasm -f elf lab7-task2.asm
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ld -m elf_i386 lab7-task2.o -o lab7-task2
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ./lab7-task2
Input A: 3
Input X: 2
10
alievruslan@VirtualBox:~/work/arch-pc/lab07$ ./lab7-task2
Input A: 2
Input X: 4
13
alievruslan@VirtualBox:~/work/arch-pc/lab07$
```

Для варианта 14:

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

При $(x = 2, a = 3)$ получается 10.

При $(x = 4, a = 2)$ получается 13.

3 Выводы

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