

Отчёт по лабораторной работе 7

Архитектура компьютеров и операционные системы

Плетьяго Кирилл НММбд-03-23

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
2.1	Самостоятельное задание	15
3	Выводы	20

Список иллюстраций

2.1	Создал каталог и файл	6
2.2	Программа в файле lab7-1.asm	7
2.3	Запуск программы lab7-1.asm	8
2.4	Программа в файле lab7-1.asm	8
2.5	Запуск программы lab7-1.asm	9
2.6	Программа в файле lab7-1.asm	10
2.7	Запуск программы lab7-1.asm	11
2.8	Программа в файле lab7-2.asm	12
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 в файле prog1.asm	16
2.14	Запуск программы prog1.asm	17
2.15	Программа в файле prog2.asm	18
2.16	Запуск программы prog2.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

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

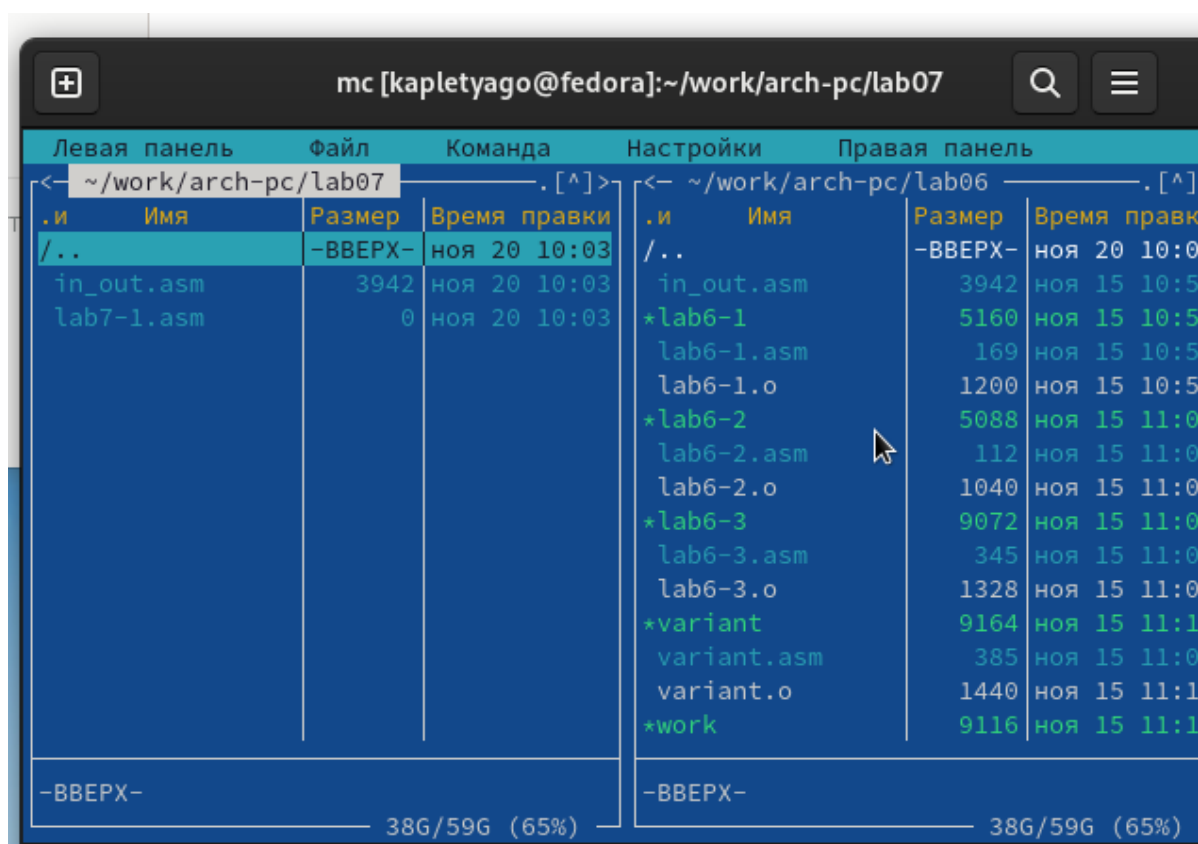
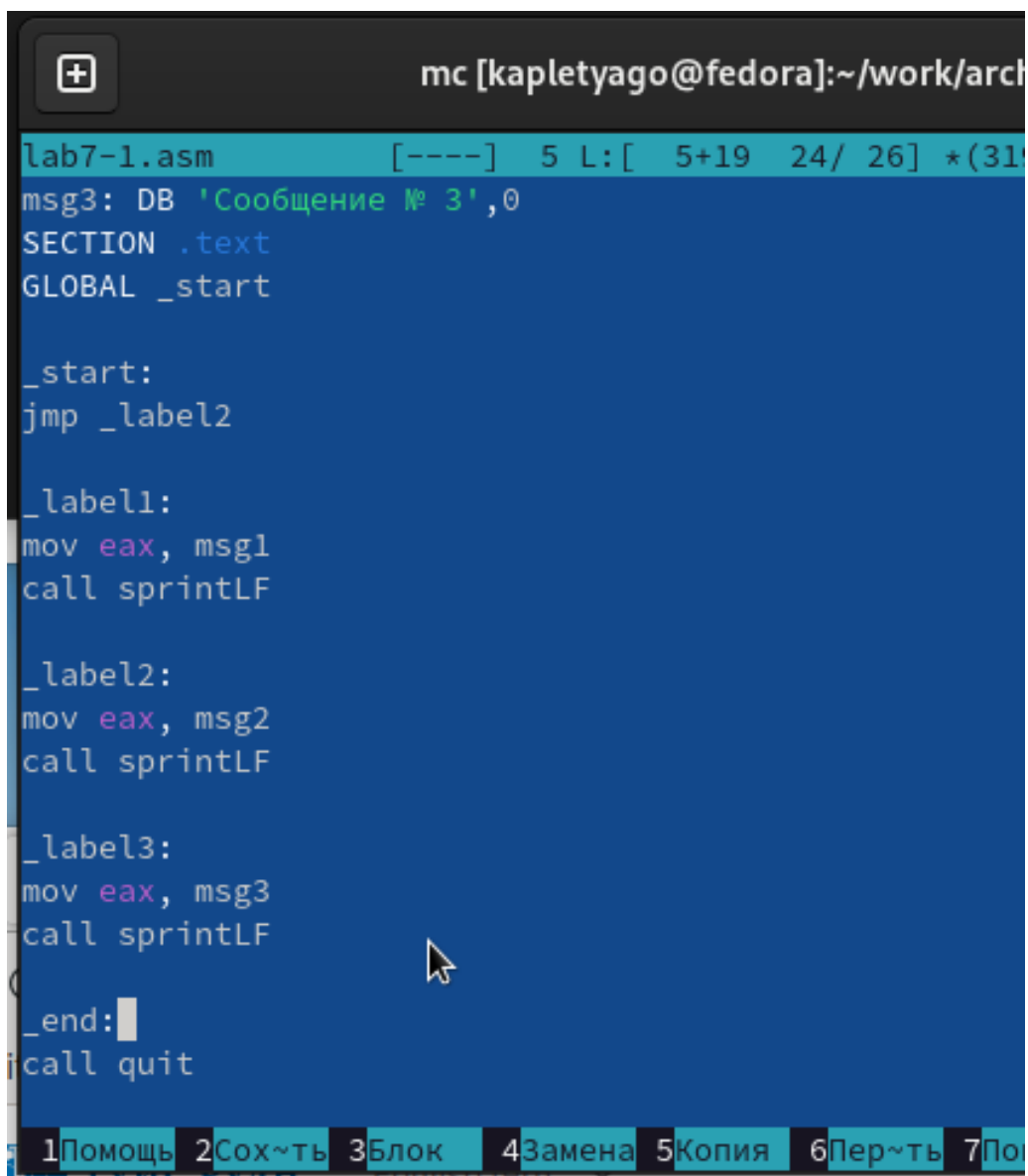


Рис. 2.1: Создал каталог и файл

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. [2.2])



```
mc [kapletyago@fedora]:~/work/arch
lab7-1.asm [----] 5 L:[ 5+19 24/ 26] *(31
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start

_start:
jmp _label2

_label1:
mov eax, msg1
call sprintfLF

_label2:
mov eax, msg2
call sprintfLF

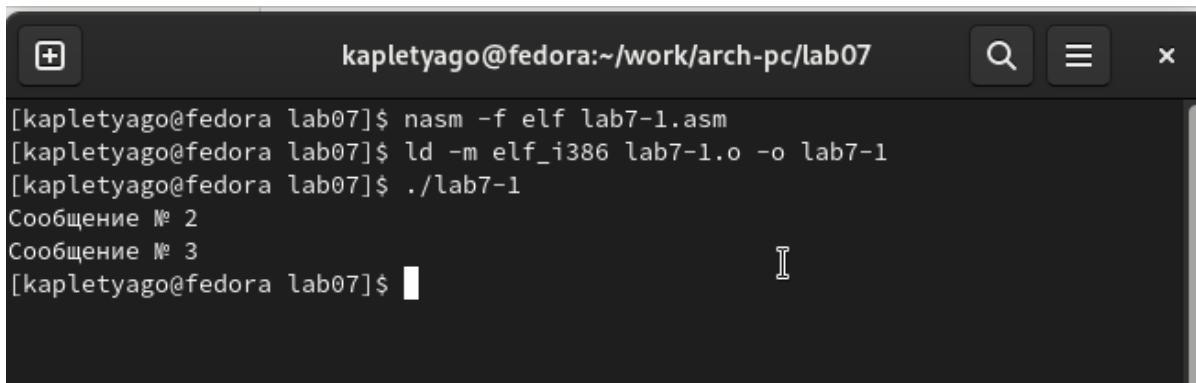
_label3:
mov eax, msg3
call sprintfLF

_end:
call quit
```

1Помощь 2Сох~ть 3Блок 4Замена 5Копия 6Пер~ть 7По

Рис. 2.2: Программа в файле lab7-1.asm

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

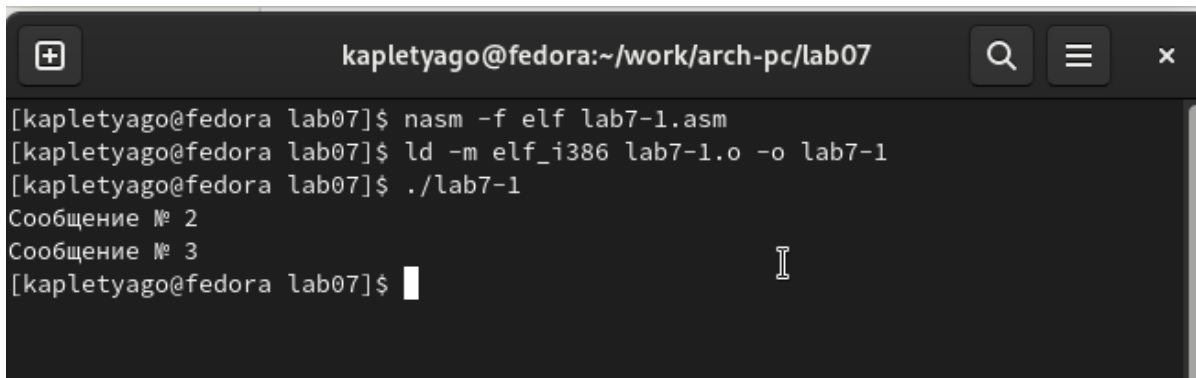
A terminal window titled 'kapletyago@fedora:~/work/arch-pc/lab07' with search, menu, and close icons. The terminal shows the following commands and output:

```
[kapletyago@fedora lab07]$ nasm -f elf lab7-1.asm
[kapletyago@fedora lab07]$ ld -m elf_i386 lab7-1.o -o lab7-1
[kapletyago@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 3
[kapletyago@fedora lab07]$
```

Рис. 2.3: Запуск программы lab7-1.asm

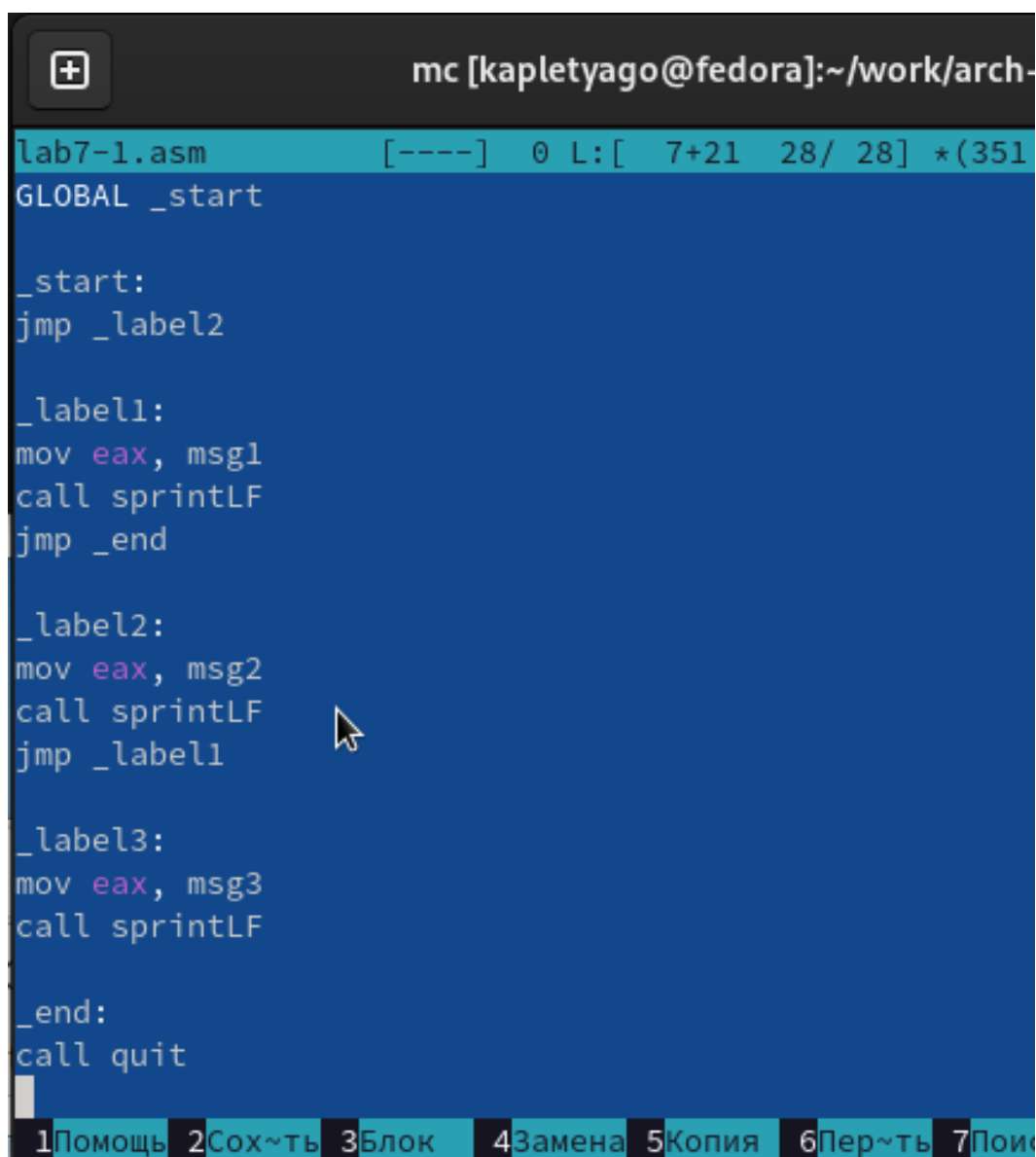
Инструкция `jmp` позволяет осуществлять переходы не только вперед но и назад. Изменим программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию `jmp` с меткой `_label1` (т.е. 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. (рис. [2.4]) (рис. [2.5])

A terminal window titled 'kapletyago@fedora:~/work/arch-pc/lab07' with search, menu, and close icons. The terminal shows the following commands and output:

```
[kapletyago@fedora lab07]$ nasm -f elf lab7-1.asm
[kapletyago@fedora lab07]$ ld -m elf_i386 lab7-1.o -o lab7-1
[kapletyago@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 3
[kapletyago@fedora lab07]$
```

Рис. 2.4: Программа в файле lab7-1.asm



```
mc [kapletyago@fedora]:~/work/arch-  
lab7-1.asm [----] 0 L:[ 7+21 28/ 28] *(351  
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  
  
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск
```

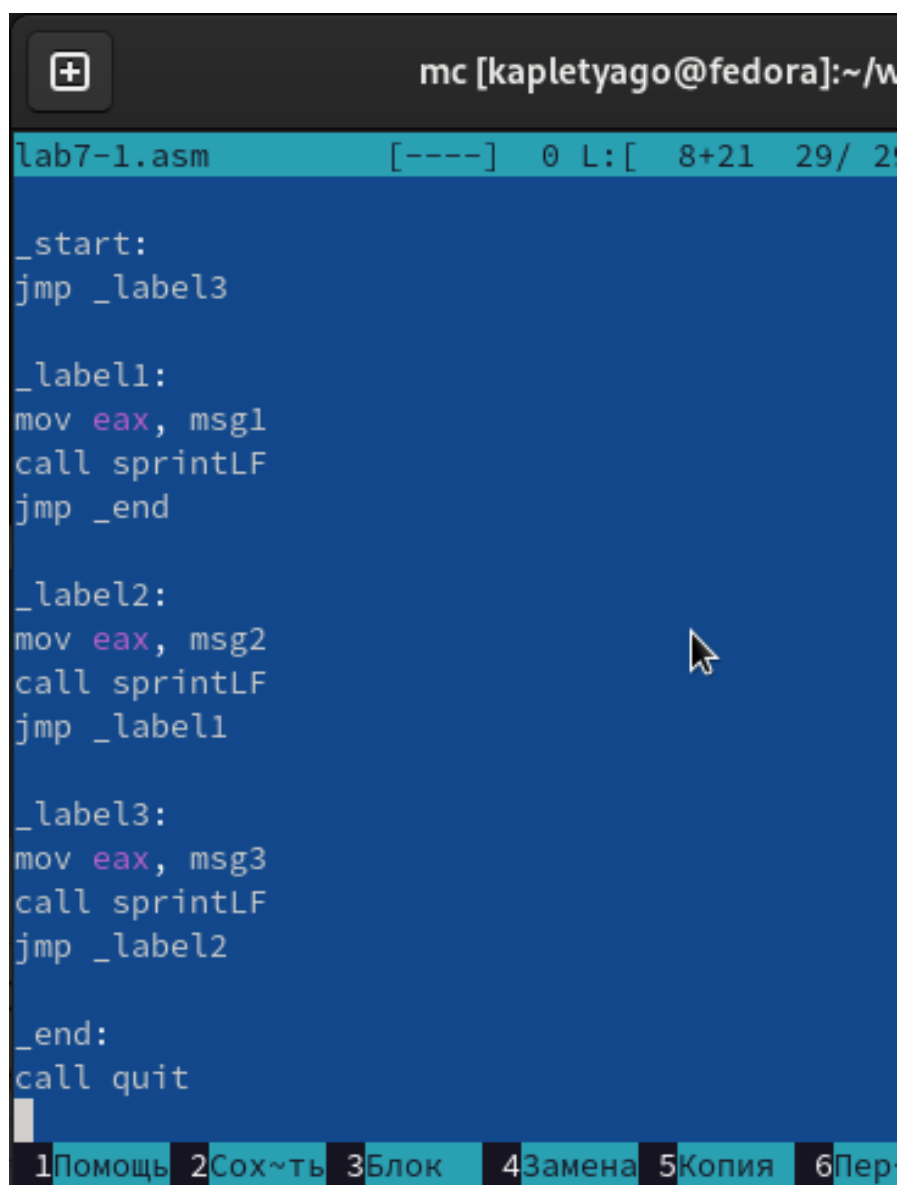
Рис. 2.5: Запуск программы lab7-1.asm

Изменил текст программы, изменив инструкции jmp, чтобы вывод программы был следующим (рис. [2.6]) (рис. [2.7]):

Сообщение № 3

Сообщение № 2

Сообщение № 1



```
mc [kapletyago@fedora]:~/w
lab7-1.asm [----] 0 L:[ 8+21 29/ 2
_start:
jmp _label3

_label1:
mov eax, msg1
call sprintLF
jmp _end

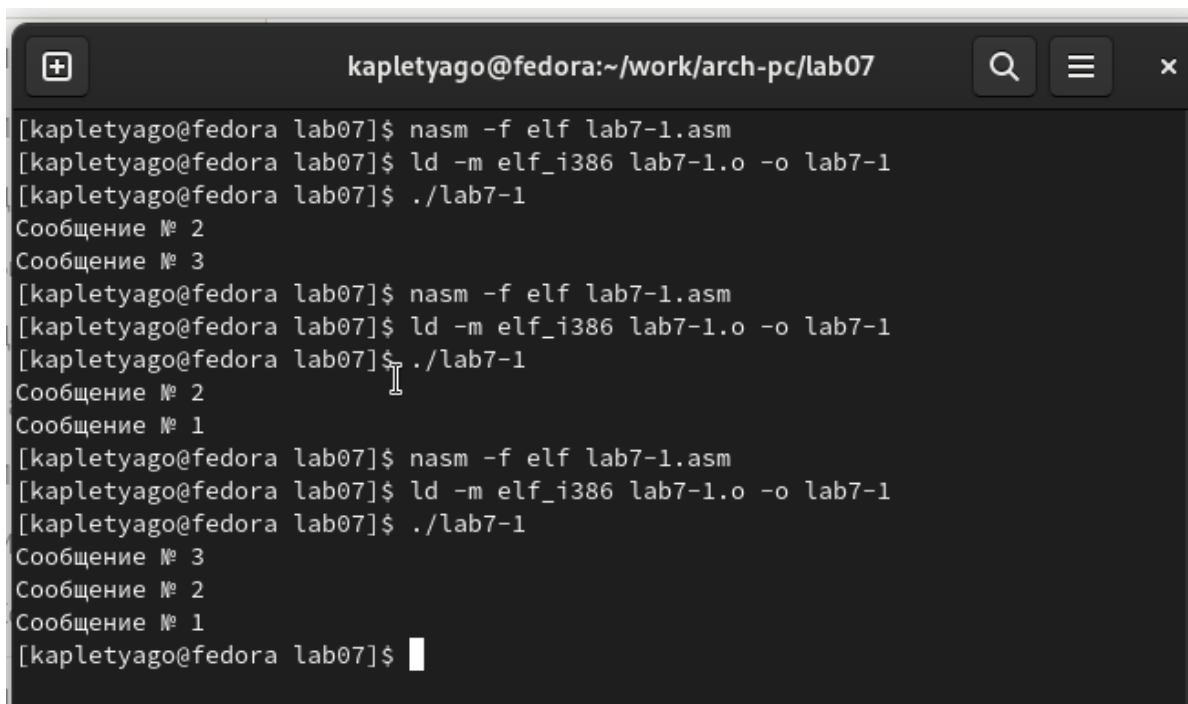
_label2:
mov eax, msg2
call sprintLF
jmp _label1

_label3:
mov eax, msg3
call sprintLF
jmp _label2

_end:
call quit
```

1Помощь 2Сох~ть 3Блок 4Замена 5Копия 6Пер

Рис. 2.6: Программа в файле lab7-1.asm

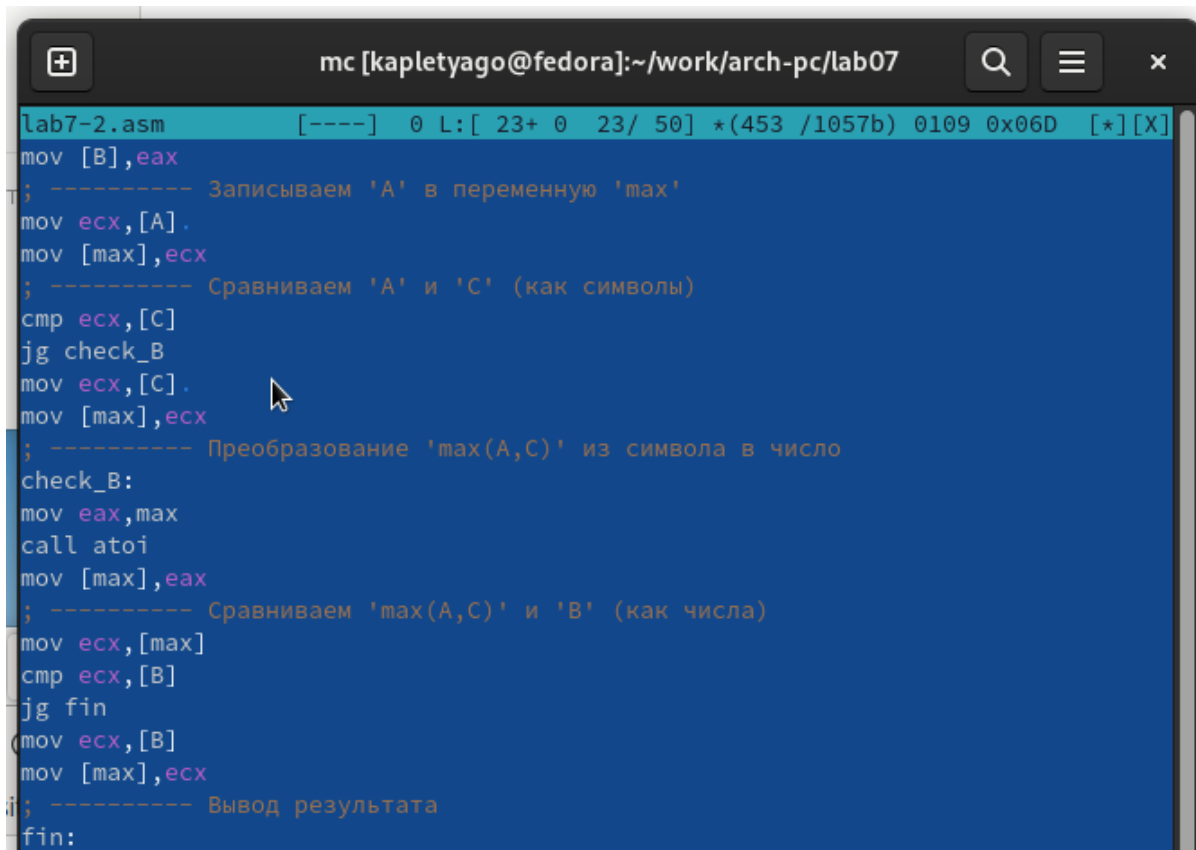


```
kapletyago@fedora:~/work/arch-pc/lab07
[kapletyago@fedora lab07]$ nasm -f elf lab7-1.asm
[kapletyago@fedora lab07]$ ld -m elf_i386 lab7-1.o -o lab7-1
[kapletyago@fedora lab07]$ ./lab7-1
Сообщение № 2
Сообщение № 3
[kapletyago@fedora lab07]$ nasm -f elf lab7-1.asm
[kapletyago@fedora lab07]$ ld -m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

Рис. 2.7: Запуск программы lab7-1.asm

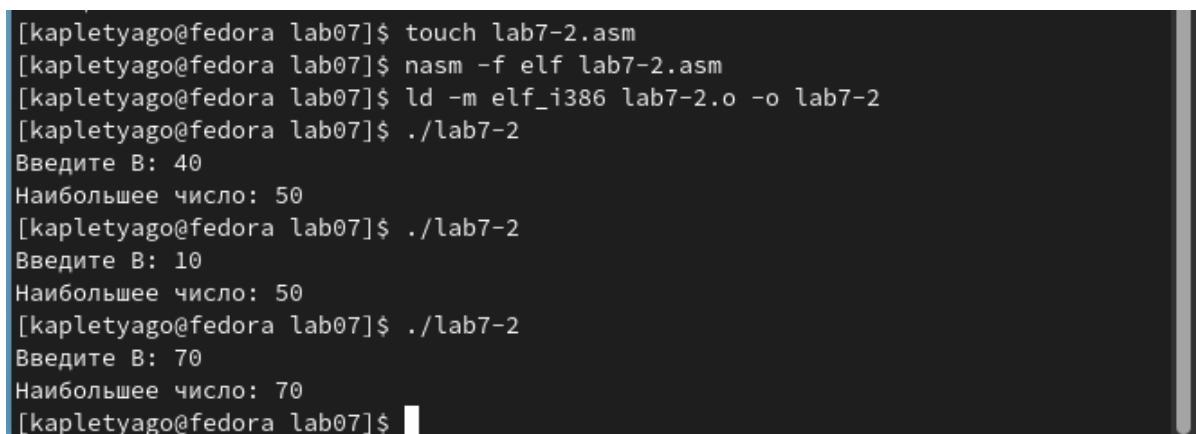
Использование инструкции `jmp` приводит к переходу в любом случае. Однако, часто при написании программ необходимо использовать условные переходы, т.е. переход должен происходить если выполнено какое-либо условие. В качестве примера рассмотрим программу, которая определяет и выводит на экран наибольшую 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 (рис. [2.8]) (рис. [2.9]).



```
mc [kapletyago@fedora]:~/work/arch-pc/lab07
lab7-2.asm [----] 0 L: [ 23+ 0 23/ 50] *(453 /1057b) 0109 0x06D [*][X]
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:
```

Рис. 2.8: Программа в файле lab7-2.asm



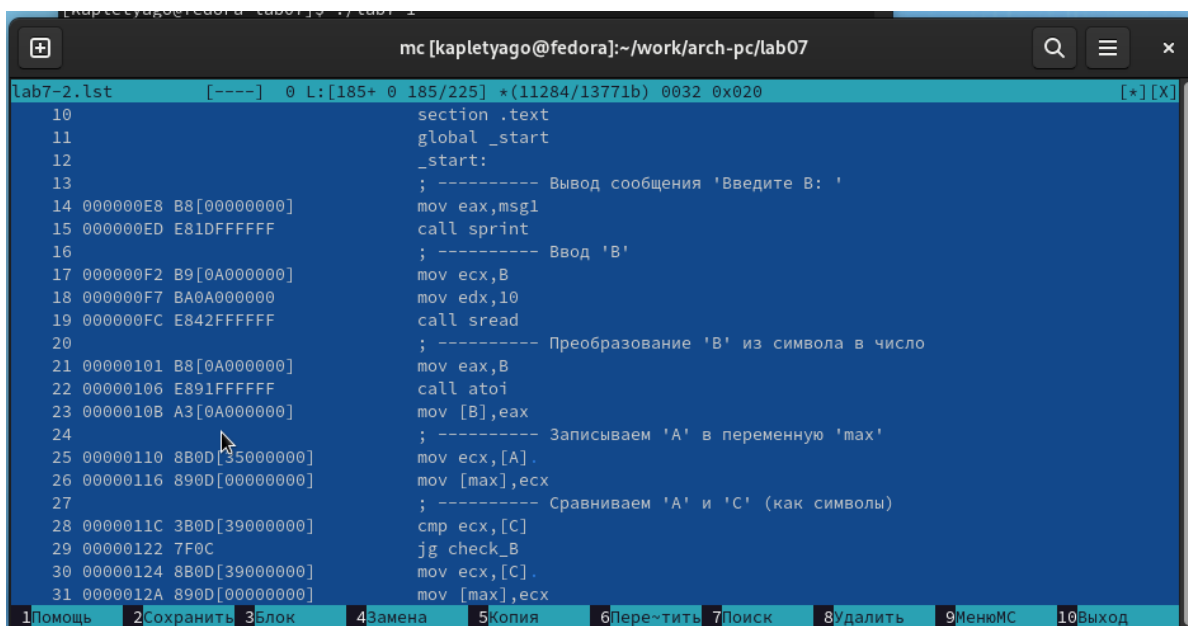
```
[kapletyago@fedora lab07]$ touch lab7-2.asm
[kapletyago@fedora lab07]$ nasm -f elf lab7-2.asm
[kapletyago@fedora lab07]$ ld -m elf_i386 lab7-2.o -o lab7-2
[kapletyago@fedora lab07]$ ./lab7-2
Введите B: 40
Наибольшее число: 50
[kapletyago@fedora lab07]$ ./lab7-2
Введите B: 10
Наибольшее число: 50
[kapletyago@fedora lab07]$ ./lab7-2
Введите B: 70
Наибольшее число: 70
[kapletyago@fedora lab07]$
```

Рис. 2.9: Запуск программы lab7-2.asm

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 (рис. [2.10])



```
lab7-2.lst [----] 0 L:[185+ 0 185/225] *(11284/13771b) 0032 0x020 [*] [X]
10 section .text
11 global _start
12 _start:
13 ; ----- Вывод сообщения 'Введите B: '
14 000000E8 B8[00000000] mov eax,msg1
15 000000ED E81DFFFFFF call sprint
16 ; ----- Ввод 'B'
17 000000F2 B9[0A000000] mov ecx,B
18 000000F7 BA0A000000 mov edx,10
19 000000FC E842FFFFFF call sread
20 ; ----- Преобразование 'B' из символа в число
21 00000101 B8[0A000000] mov eax,B
22 00000106 E891FFFFFF call atoi
23 0000010B A3[0A000000] mov [B],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 jg check_B
30 00000124 8B0D[39000000] mov ecx,[C].
31 0000012A 890D[00000000] mov [max],ecx
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Пере-тить 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 2.10: Файл листинга lab7-2

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

строка 189

- 14 - номер строки в подпрограмме
- 000000E8 - адрес
- B8[00000000] - машинный код
- mov eax,msg1 - код программы - перекладывает msg1 в eax

строка 190

- 15 - номер строки в подпрограмме
- 000000ED - адрес

- E81DFFFFFF - машинный код
- call sprint - код программы - вызов подпрограммы печати

строка 192

- 17 - номер строки в подпрограмме
- 000000F2 - адрес
- B9[0A000000] - машинный код
- mov ecx,B - код программы - перекладывает B в ecx

Открыл файл с программой lab7-2.asm и в инструкции с двумя операндами удалил один операнд. Выполнил трансляцию с получением файла листинга. (рис. [2.11]) (рис. [2.12])

```
[kapletyago@fedora lab07]$ nasm -f elf lab7-2.asm -l lab7-2.lst
[kapletyago@fedora lab07]$
[kapletyago@fedora lab07]$ nasm -f elf lab7-2.asm -l lab7-2.lst
lab7-2.asm:34: error: invalid combination of opcode and operands
[kapletyago@fedora lab07]$
[kapletyago@fedora lab07]$
```

Рис. 2.11: Ошибка трансляции lab7-2

```
mc [kapletyago@fedora:~/work/arch-pc/lab07]
lab7-2.lst [----] 50 L:[205+ 5 210/226] *(12949/13859b) 0118 0x076 [*][X]
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_B:
34 mov eax,
34 ***** error: invalid combination of opcode and operands
35 00000130 E867FFFFFF call atoi
36 00000135 A3[00000000] mov [max],eax
37 ; ----- Сравниваем 'max(A,C)' и 'B' (как числа)
38 0000013A 8B0D[0A000000] mov ecx,[max]
39 00000140 3B0D[0A000000] cmp ecx,[B]
40 00000146 7F0C jg fin
41 00000148 8B0D[0A000000] mov ecx,[B]
42 0000014E 890D[00000000] mov [max],ecx
43 ; ----- Вывод результата
44 fin:
45 00000154 B8[13000000] mov eax,msg2
46 00000159 E8B1FFFFFF call sprint
47 0000015E A1[00000000] mov eax,[max]
48 00000163 E81EFFFFFF call iprintLF
49 00000168 E86EFFFFFF call quit
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Пере-тить 7Поиск 8Удалить 9МенюМС 10Выход
```

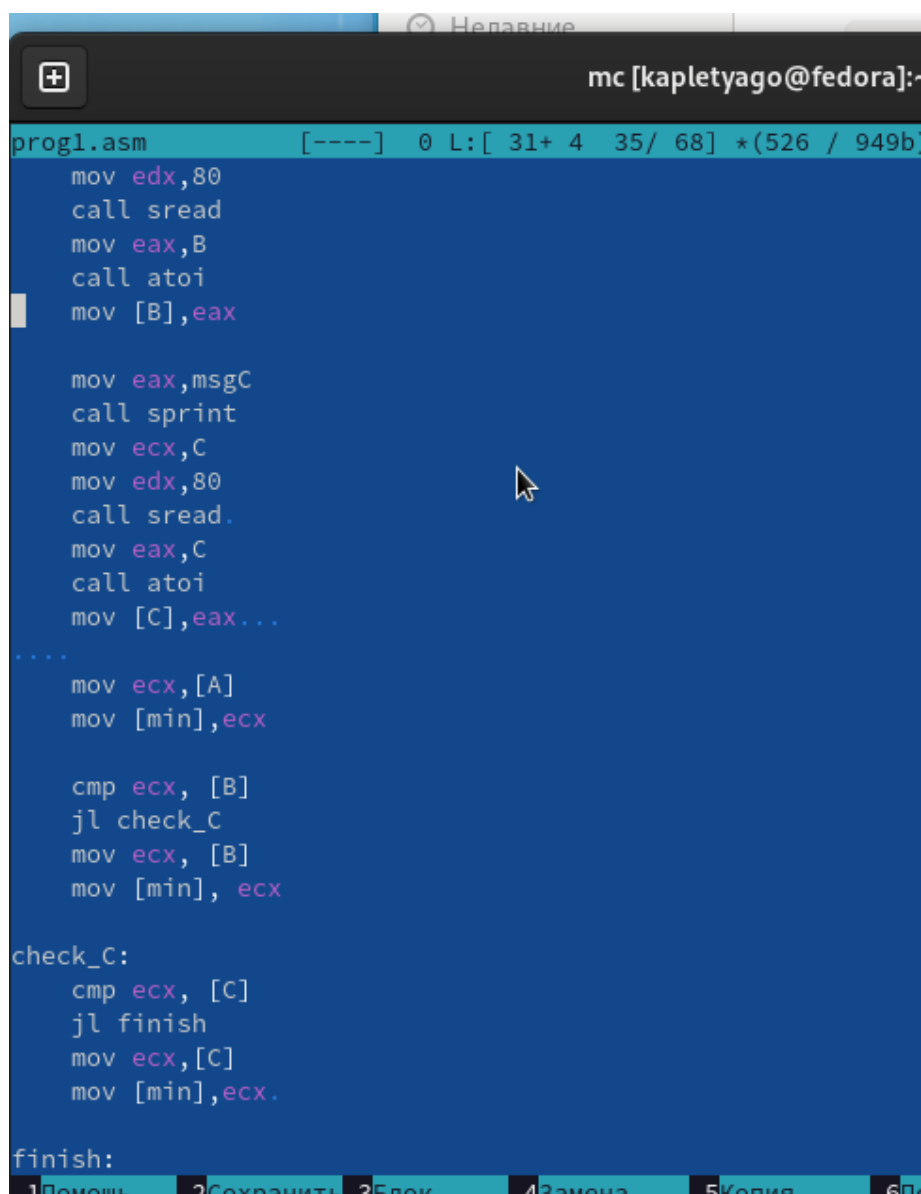
Рис. 2.12: Файл листинга с ошибкой lab7-2

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

2.1 Самостоятельное задание

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 (рис. [2.13]) (рис. [2.14])

для варианта 8 - 52, 33, 40



```
mc [kapletyago@fedora]:-
prog1.asm [----] 0 L: [ 31+ 4 35/ 68] *(526 / 949b)
    mov edx,80
    call sread
    mov eax,B
    call atoi
    mov [B],eax

    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:
```

Рис. 2.13: Программа в файле prog1.asm


```
[kapletyago@fedora lab07]$  
[kapletyago@fedora lab07]$ nasm -f elf prog1.asm  
[kapletyago@fedora lab07]$ ld -m elf_i386 prog1.o -o prog1  
[kapletyago@fedora lab07]$ ./prog1  
Input A: 52  
Input B: 33  
Input C: 40  
Smallest: 33  
[kapletyago@fedora lab07]$  
[kapletyago@fedora lab07]$
```

Рис. 2.14: Запуск программы prog1.asm

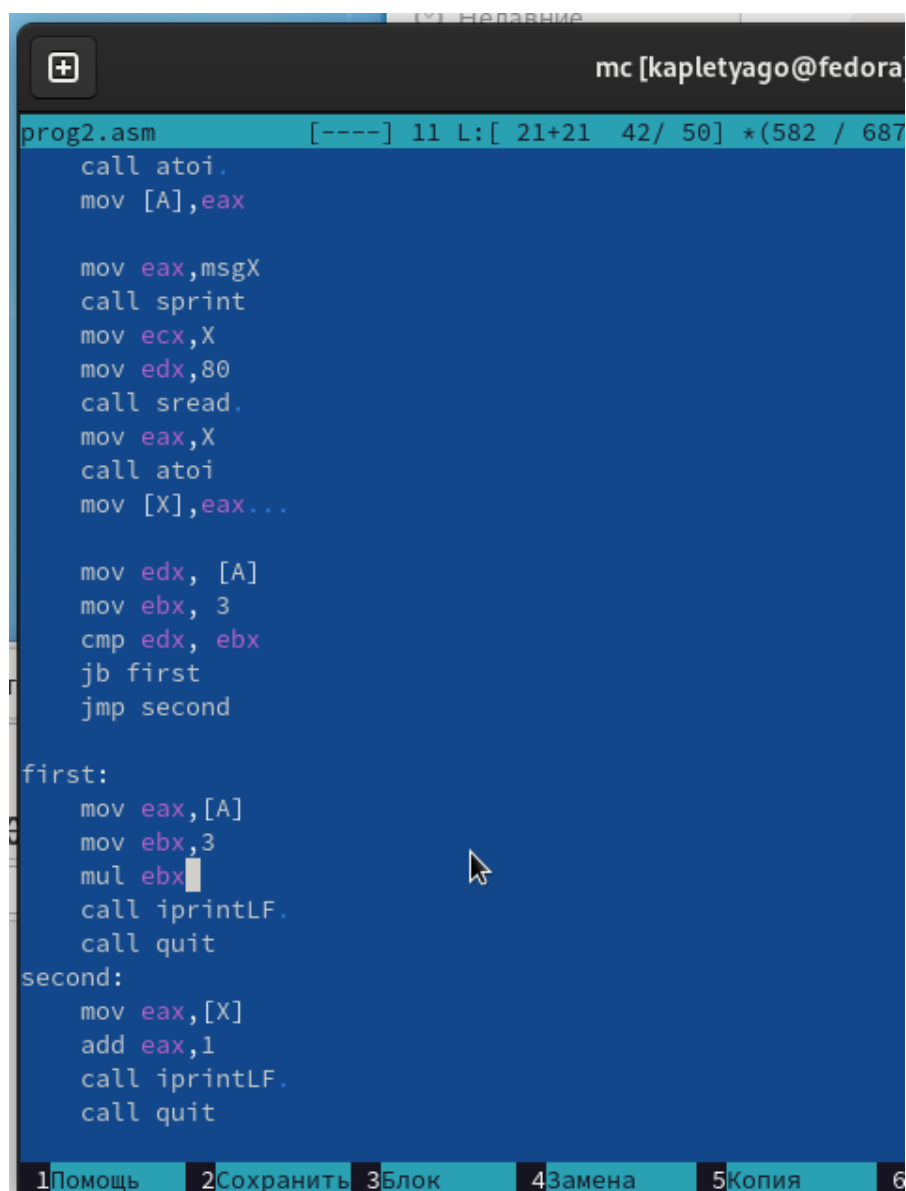
Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции $f(x)$ и выводит результат вычислений. Вид функции $f(x)$ выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений X и a из 7.6. (рис. [2.15]) (рис. [2.16])

для варианта 8

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

Если подставить $x = 1, a = 4$ получается $1 + 1 = 2$.

Если подставить $x = 1, a = 2$ получается $3 * 2 = 6$.



```
mc [kapletyago@fedora]
prog2.asm [----] 11 L: [ 21+21 42/ 50] *(582 / 687
    call atoi.
    mov [A],eax

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

    mov edx, [A]
    mov ebx, 3
    cmp edx, ebx
    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

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6
```

Рис. 2.15: Программа в файле prog2.asm

```
[kapletyago@fedora lab07]$  
[kapletyago@fedora lab07]$ nasm -f elf prog2.asm  
[kapletyago@fedora lab07]$ ld -m elf_i386 prog2.o -o prog2  
[kapletyago@fedora lab07]$ ./prog2  
Input A: 4  
Input X: 1  
2  
[kapletyago@fedora lab07]$ ./prog2  
Input A: 2  
Input X: 1  
6  
[kapletyago@fedora lab07]$
```

Рис. 2.16: Запуск программы prog2.asm

3 Выводы

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