

# **Отчёт по лабораторной работе №7**

**Арифметические операции в NASM**

Аскеров Александр Эдуардович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>5</b>
2.1	Символьные и численные данные в NASM . . . . .	5
2.2	Выполнение арифметических операций в NASM . . . . .	11
2.3	Задание для самостоятельной работы . . . . .	17
<b>3</b>	<b>Выводы</b>	<b>21</b>

## Список иллюстраций

2.1	Создание файла lab7-1.asm в каталоге lab07 . . . . .	5
2.2	Программа из листинга 7.1 . . . . .	6
2.3	Результат работы программы lab7-1 . . . . .	6
2.4	Изменённый текст программы lab7-1 . . . . .	7
2.5	Результат работы программы lab7-1 . . . . .	8
2.6	Создание файла lab7-2.asm в каталоге lab07 . . . . .	8
2.7	Программа из листинга 7.2 . . . . .	9
2.8	Результат работы программы lab7-2 . . . . .	9
2.9	Изменённый текст программы lab7-2 . . . . .	10
2.10	Результат работы программы lab7-2 . . . . .	10
2.11	Создание файла lab7-3.asm в каталоге lab07 . . . . .	11
2.12	Программа из листинга 7.3 . . . . .	12
2.13	Результат работы программы lab7-3 . . . . .	13
2.14	Изменённый текст программы lab7-3 . . . . .	13
2.15	Результат работы программы lab7-3 . . . . .	14
2.16	Создание файла variant.asm в каталоге lab07 . . . . .	14
2.17	Программа из листинга 7.4 . . . . .	15
2.18	Результат работы программы variant . . . . .	16
2.19	Вывод на экран сообщения 'Ваш вариант:' . . . . .	16
2.20	Строки, отвечающие за вычисление варианта . . . . .	17
2.21	Строки, отвечающие за вывод на экран результата вычислений . . . . .	17
2.22	Программа lab7-4 для самостоятельной работы . . . . .	19
2.23	Результат работы программы lab7-4 . . . . .	20

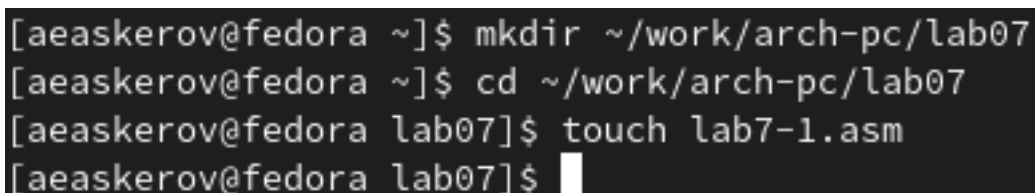
# 1 Цель работы

Освоить арифметические инструкции языка ассемблера NASM.

## 2 Выполнение лабораторной работы

### 2.1 Символьные и численные данные в NASM

1. Создадим каталог для программ лабораторной работы № 7, перейдём в него и создадим файл lab7-1.asm.



```
[aeaskerov@fedora ~]$ mkdir ~/work/arch-pc/lab07
[aeaskerov@fedora ~]$ cd ~/work/arch-pc/lab07
[aeaskerov@fedora lab07]$ touch lab7-1.asm
[aeaskerov@fedora lab07]$
```

Рис. 2.1: Создание файла lab7-1.asm в каталоге lab07

2. Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения, записанные в регистр eax.

Введём в файл lab7-1.asm текст программы из листинга 7.1. В данной программе в регистр eax записывается символ 6 (mov eax,'6'), в регистр ebx символ 4 (mov ebx,'4'). Далее к значению в регистре eax прибавляем значение регистра ebx (add eax,ebx, результат сложения запишется в регистр eax). Далее выводим результат. Так как для работы функции sprintf в регистр eax должен быть записан адрес, необходимо использовать дополнительную переменную. Для этого запишем значение регистра eax в переменную buf1 (mov [buf1],eax), а затем запишем адрес переменной buf1 в регистр eax (mov eax,buf1) и вызовем функцию sprintf.

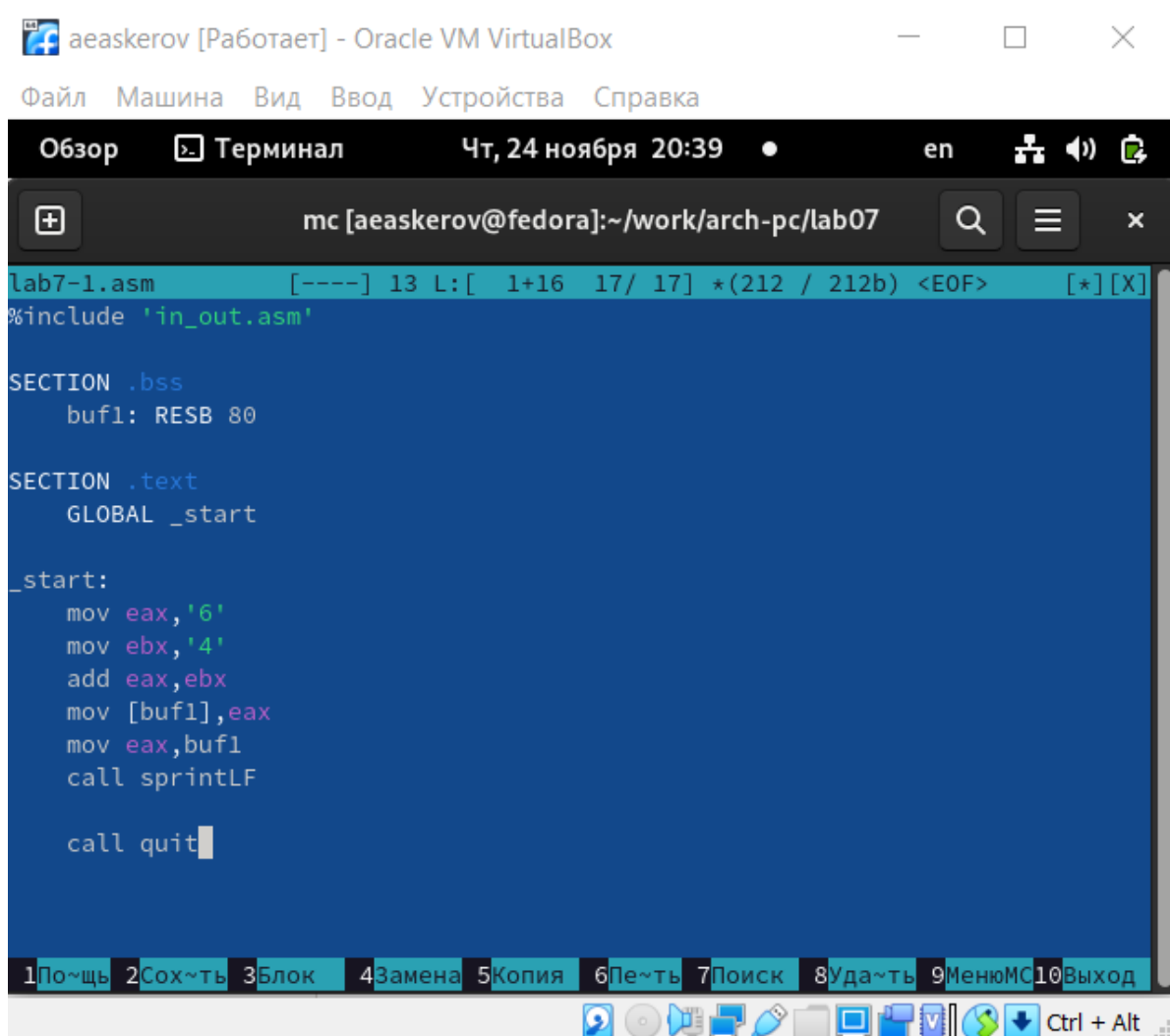


Рис. 2.2: Программа из листинга 7.1

Создадим исполняемый файл и запустим его.

```

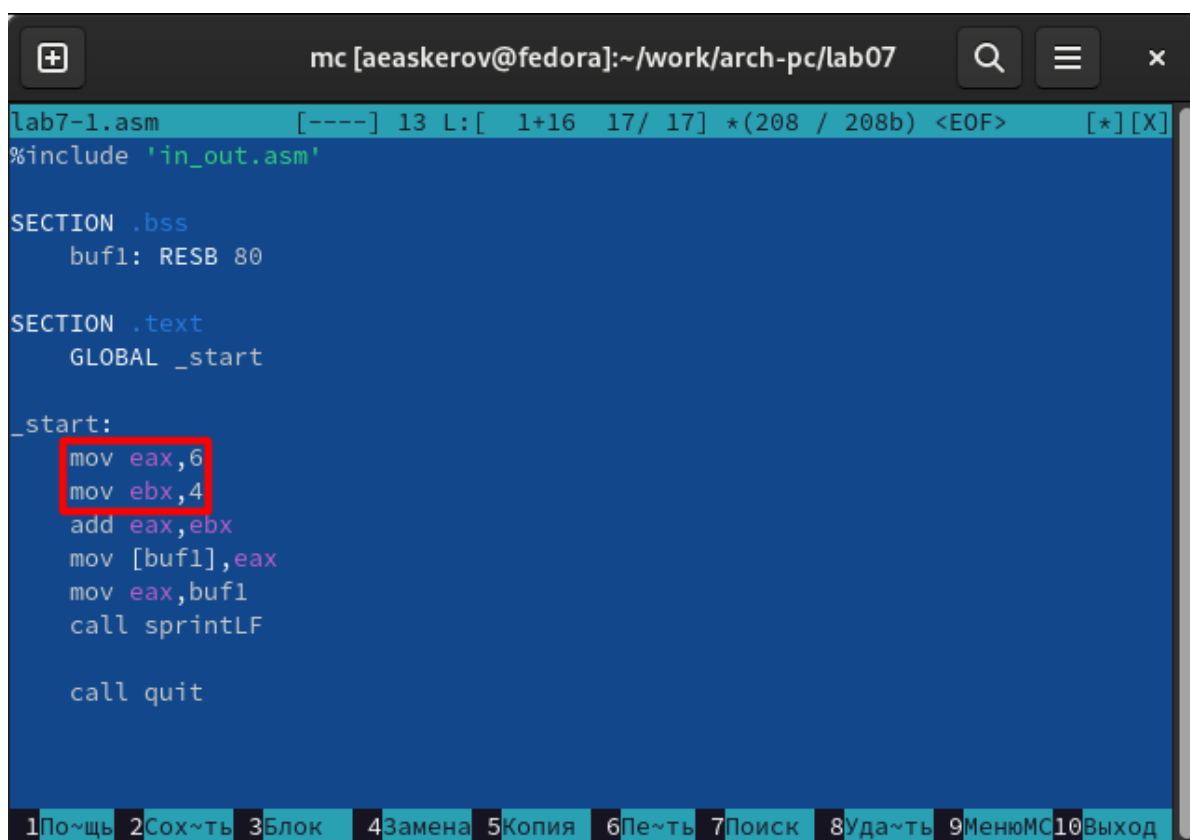
[aeaskerov@fedora lab07]$ nasm -f elf lab7-1.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[aeaskerov@fedora lab07]$ ./lab7-1
j
[aeaskerov@fedora lab07]$
  
```

Рис. 2.3: Результат работы программы lab7-1

В данном случае при выводе значения регистра `eax` мы ожидаем увидеть число 10. Однако результатом будет символ `j`. Это происходит потому, что код символа 6

равен 00110110 в двоичном представлении (или 54 в десятичном представлении), а код символа 4 – 00110100 (52). Команда `add eax,ebx` запишет в регистр `eax` сумму кодов – 01101010 (106), что в свою очередь является кодом символа `j` (см. таблицу ASCII в приложении).

3. Далее изменим текст программы и вместо символов, запишем в регистры числа. Исправим текст программы (Листинг 1) следующим образом: заменим строки `mov eax,'6'` `mov ebx,'4'` на строки `mov eax,6` `mov ebx,4`.



```
mc [aeaskerov@fedora]:~/work/arch-pc/lab07
lab7-1.asm [----] 13 L: [ 1+16 17/ 17] *(208 / 208b) <EOF> [*] [X]
#include 'in_out.asm'

SECTION .bss
    buf1: RESB 80

SECTION .text
    GLOBAL _start

_start:
    mov eax,6
    mov ebx,4
    add eax,ebx
    mov [buf1],eax
    mov eax,buf1
    call sprintf
    call quit
```

Рис. 2.4: Изменённый текст программы lab7-1

Создадим исполняемый файл и запустим его.

```
[aeaskerov@fedora lab07]$ nasm -f elf lab7-1.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[aeaskerov@fedora lab07]$ ./lab7-1

[aeaskerov@fedora lab07]$
```

Рис. 2.5: Результат работы программы lab7-1

Как и в предыдущем случае при исполнении программы мы не получим число 10. В данном случае выводится символ с кодом 10. Из таблицы ASCII видно, что это символ переноса строки. При выводе на экран он не отображается.

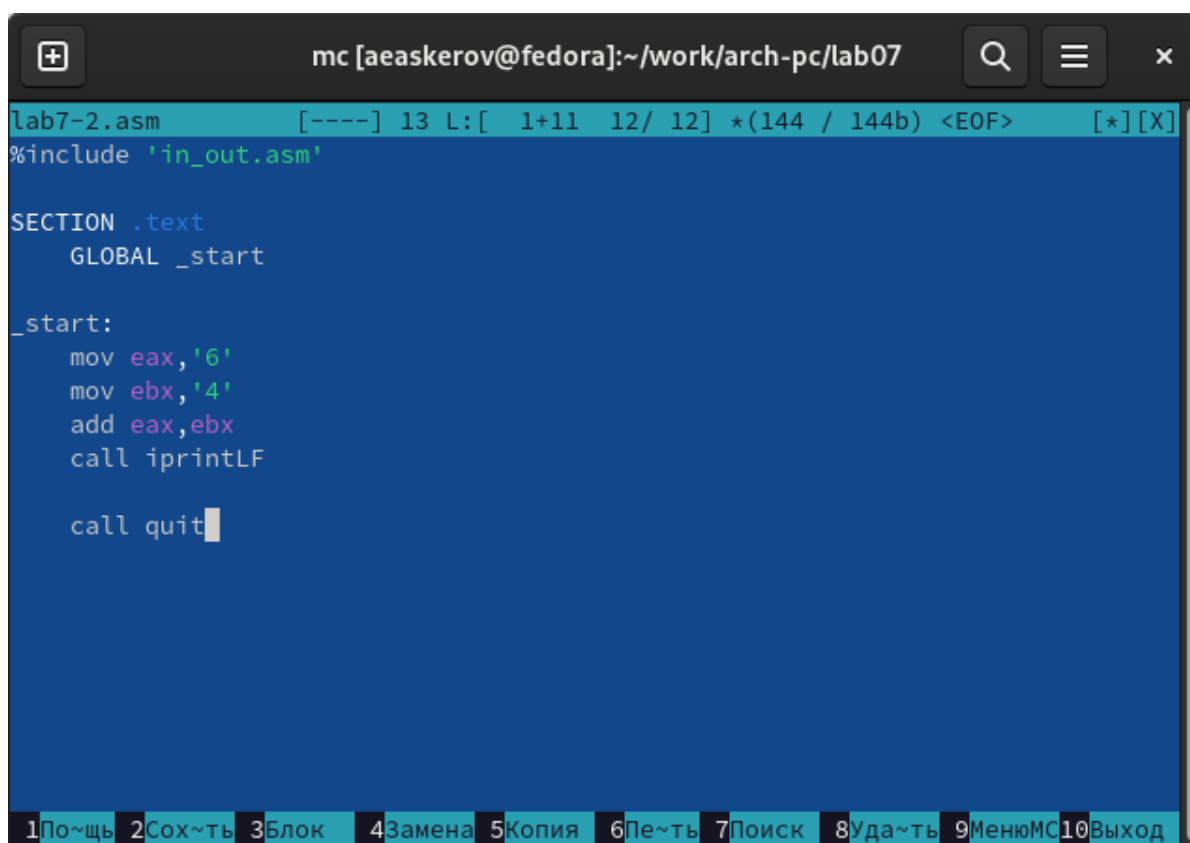
4. Как отмечалось выше, для работы с числами в файле `in_out.asm` реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Преобразуем текст программы из Листинга 7.1 с использованием этих функций.

Создадим файл `lab7-2.asm` в каталоге `~/work/arch-pc/lab07` и введём в него текст программы из листинга 7.2.

```
[aeaskerov@fedora lab07]$ touch ~/work/arch-pc/lab07/lab7-2.asm
[aeaskerov@fedora lab07]$
```

Рис. 2.6: Создание файла lab7-2.asm в каталоге lab07





```
lab7-2.asm [----] 13 L: [ 1+11 12/ 12] *(144 / 144b) <EOF> [*] [X]
#include 'in_out.asm'

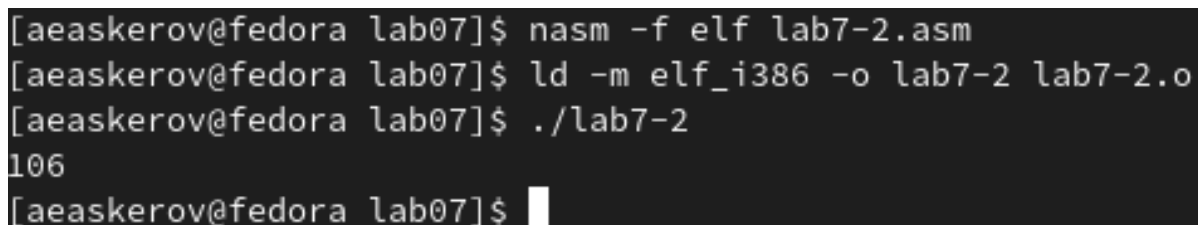
SECTION .text
    GLOBAL _start

_start:
    mov eax, '6'
    mov ebx, '4'
    add eax, ebx
    call iprintLF

    call quit
```

Рис. 2.7: Программа из листинга 7.2

Создадим исполняемый файл и запустим его.



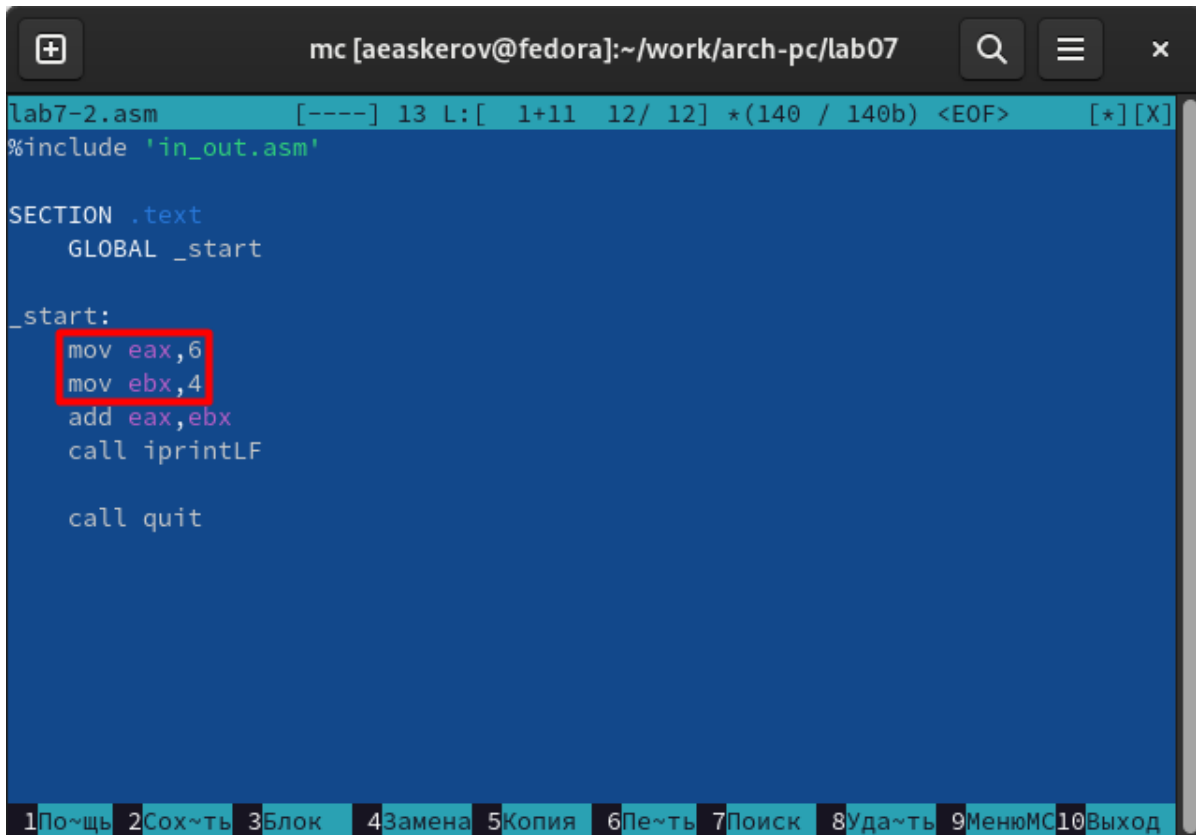
```
[aeaskerov@fedora lab07]$ nasm -f elf lab7-2.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[aeaskerov@fedora lab07]$ ./lab7-2
106
[aeaskerov@fedora lab07]$
```

Рис. 2.8: Результат работы программы lab7-2

В результате работы программы мы получаем число 106. В данном случае, как и в первом, команда `add` складывает коды символов '6' и '4' ( $54+52=106$ ). Однако, в отличие от программы из листинга 7.1, функция `iprintLF` позволяет вывести число, а не символ, кодом которого является это число.

5. Аналогично предыдущему примеру изменим символы на числа. Заменим

строки `mov eax, '6'` `mov ebx, '4'` на строки `mov eax, 6` `mov ebx, 4`.



```
lab7-2.asm [----] 13 L: [ 1+11 12/ 12] *(140 / 140b) <EOF> [*] [X]
#include 'in_out.asm'

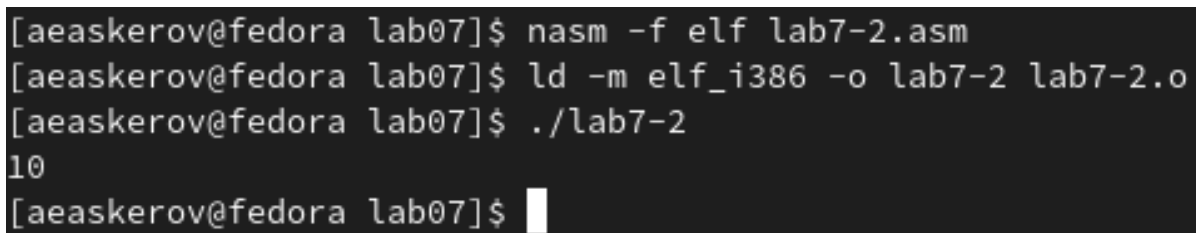
SECTION .text
GLOBAL _start

_start:
    mov eax, 6
    mov ebx, 4
    add eax, ebx
    call iprintLF

    call quit
```

Рис. 2.9: Изменённый текст программы lab7-2

Создадим исполняемый файл и запустим его. В результате мы получили 10.



```
[aeaskerov@fedora lab07]$ nasm -f elf lab7-2.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[aeaskerov@fedora lab07]$ ./lab7-2
10
[aeaskerov@fedora lab07]$
```

Рис. 2.10: Результат работы программы lab7-2

Заменяем функцию `iprintLF` на `iprint`. Создадим исполняемый файл и запустим его. Отличие вывода функции `iprintLF` от `iprint` заключается в том, что исчезает перенос строки после вывода информации на экран.

## 2.2 Выполнение арифметических операций в NASM

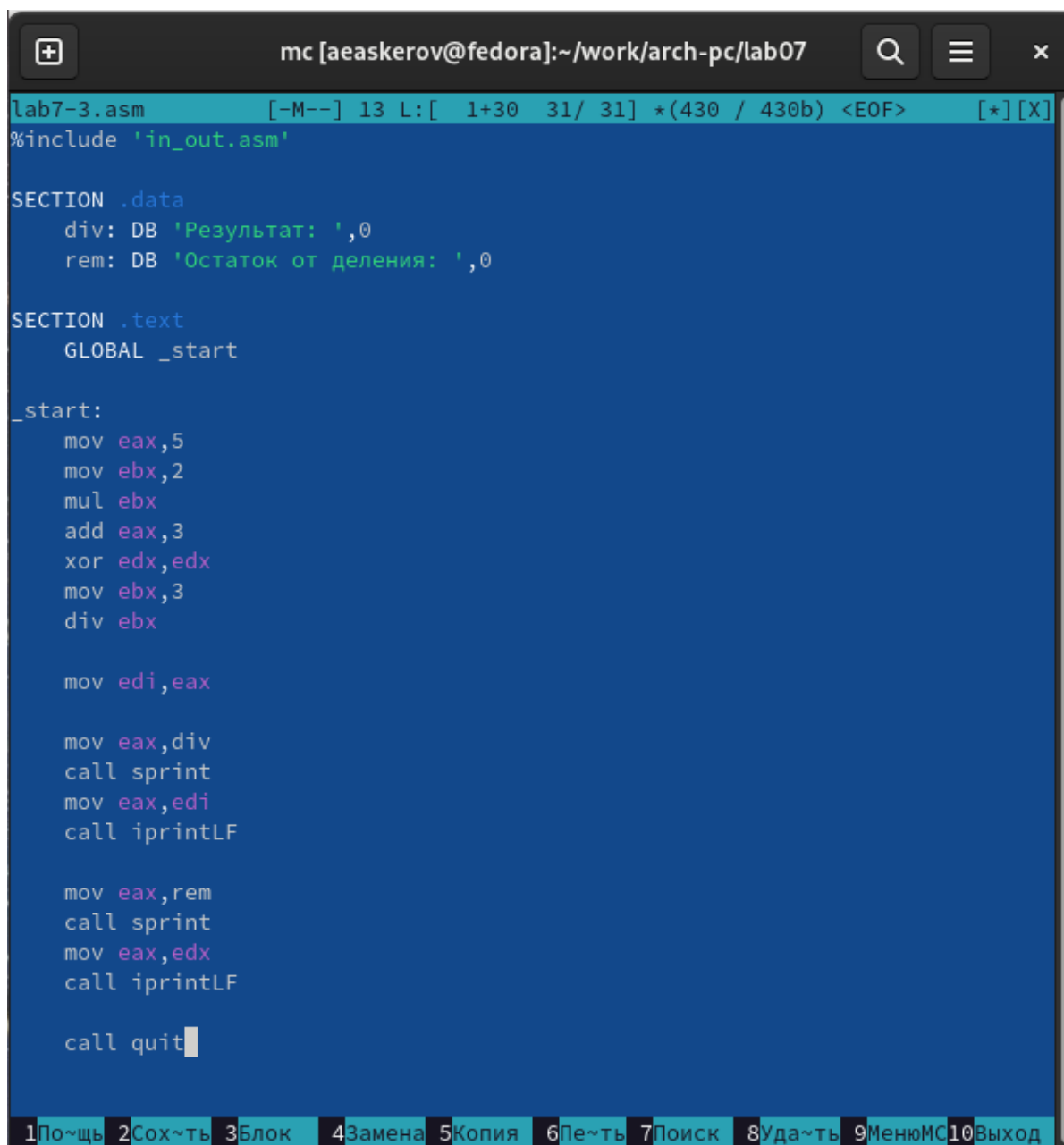
В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения  $f(x) = (5 * 2 + 3)/3$ .

Создадим файл lab7-3.asm в каталоге ~/work/arch-pc/lab07.

```
10[aeaskerov@fedora lab07]$ touch ~/work/arch-pc/lab07/lab7-3.asm  
[aeaskerov@fedora lab07]$
```

Рис. 2.11: Создание файла lab7-3.asm в каталоге lab07

Внимательно изучим текст программы из листинга 7.3 и введём в lab7- 3.asm.



```
lab7-3.asm [-M--] 13 L: [ 1+30 31/ 31] *(430 / 430b) <EOF> [*] [X]
#include 'in_out.asm'

SECTION .data
    div: DB 'Результат: ',0
    rem: DB 'Остаток от деления: ',0

SECTION .text
    GLOBAL _start

_start:
    mov eax,5
    mov ebx,2
    mul ebx
    add eax,3
    xor edx,edx
    mov ebx,3
    div ebx

    mov edi,eax

    mov eax,div
    call sprint
    mov eax,edi
    call iprintLF

    mov eax,rem
    call sprint
    mov eax,edx
    call iprintLF

    call quit
```

Рис. 2.12: Программа из листинга 7.3

Создадим исполняемый файл и запустим его.

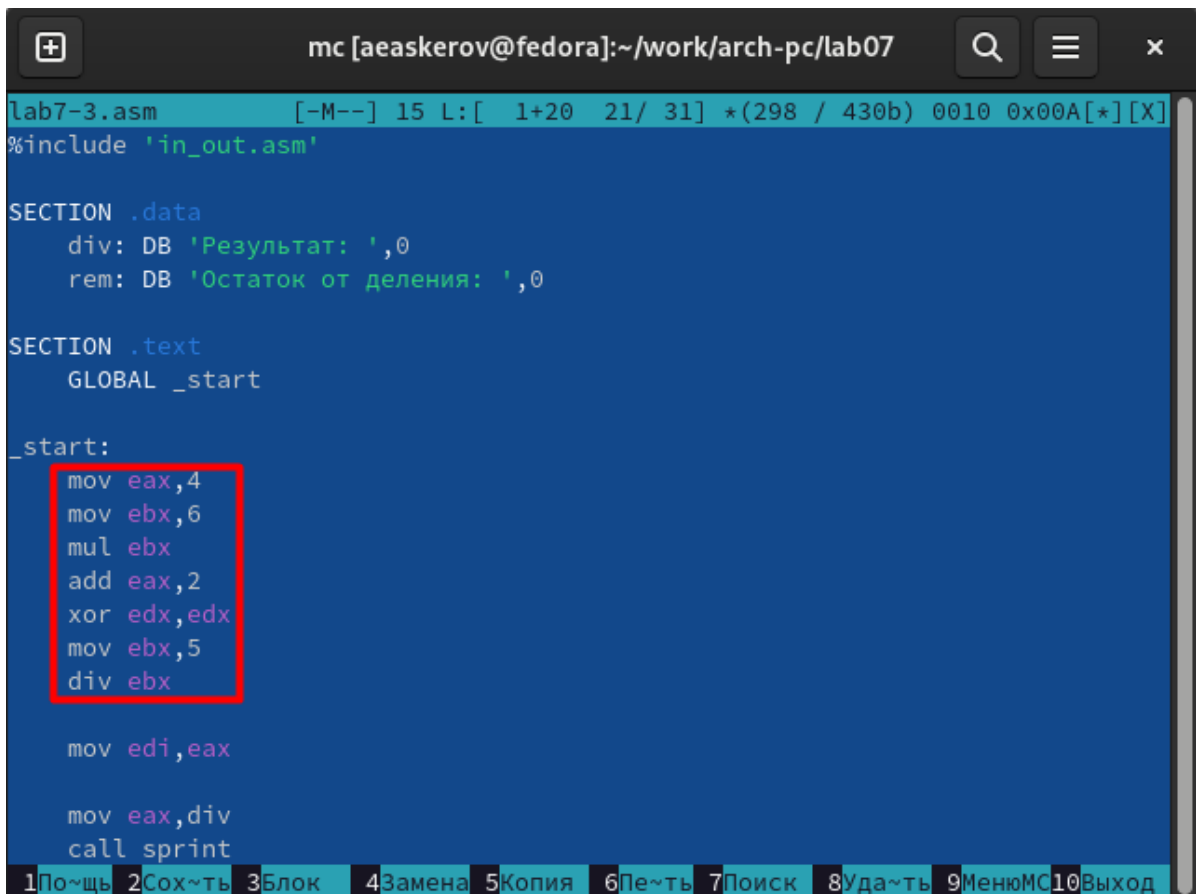
```

[aeaskerov@fedora lab07]$ nasm -f elf lab7-3.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[aeaskerov@fedora lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
[aeaskerov@fedora lab07]$

```

Рис. 2.13: Результат работы программы lab7-3

Изменим текст программы для вычисления выражения  $f(x) = (4 * 6 + 2)/5$ .



```

lab7-3.asm  [-M--] 15 L:[ 1+20 21/ 31] *(298 / 430b) 0010 0x00A[*][X]
#include 'in_out.asm'

SECTION .data
    div: DB 'Результат: ',0
    rem: DB 'Остаток от деления: ',0

SECTION .text
    GLOBAL _start

_start:
    mov eax,4
    mov ebx,6
    mul ebx
    add eax,2
    xor edx,edx
    mov ebx,5
    div ebx

    mov edi,eax

    mov eax,div
    call sprint

```

Рис. 2.14: Изменённый текст программы lab7-3

Создадим исполняемый файл и проверим его работу.

```
[aeaskerov@fedora lab07]$ nasm -f elf lab7-3.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[aeaskerov@fedora lab07]$ ./lab7-3
Результат: 5
Остаток от деления: 1
[aeaskerov@fedora lab07]$
```

Рис. 2.15: Результат работы программы lab7-3

7. В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму:

- вывести запрос на введение № студенческого билета
- вычислить номер варианта по формуле:  $(S_n \bmod 20) + 1$ , где  $S_n$  – номер студенческого билета (в данном случае  $a \bmod b$  – это остаток от деления  $a$  на  $b$ )
- вывести на экран номер варианта

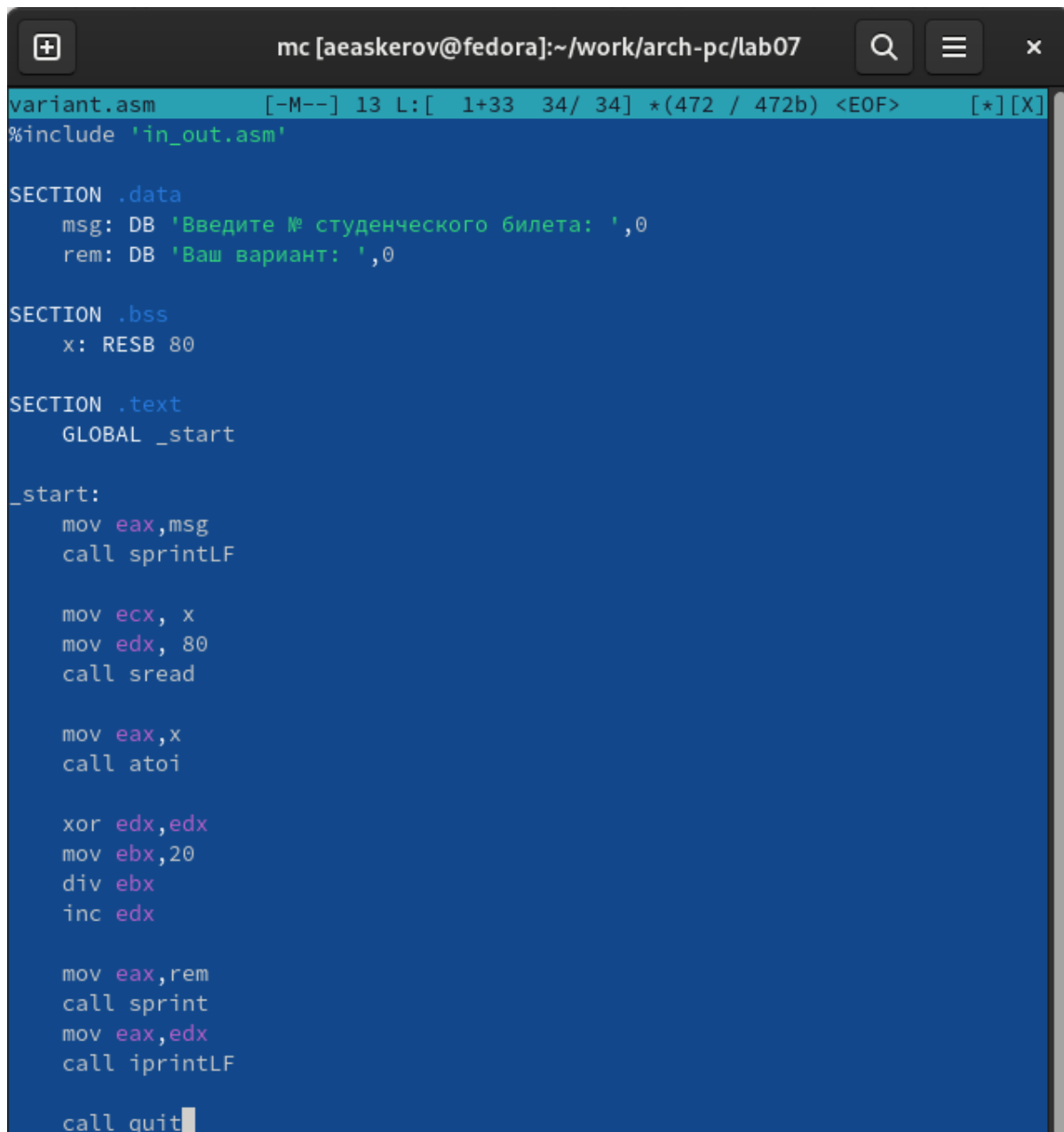
В данном случае число, над которым необходимо проводить арифметические операции, вводится с клавиатуры. Как отмечалось выше ввод с клавиатуры осуществляется в символьном виде, и для корректной работы арифметических операций в NASM символы необходимо преобразовать в числа. Для этого может быть использована функция `atoi` из файла `in_out.asm`.

Создадим файл `variant.asm` в каталоге `~/work/arch-pc/lab07`.

```
[aeaskerov@fedora lab07]$ touch ~/work/arch-pc/lab07/variant.asm
[aeaskerov@fedora lab07]$
```

Рис. 2.16: Создание файла `variant.asm` в каталоге `lab07`

Внимательно изучим текст программы из листинга 7.4 и введём в файл `variant.asm`.



```
variant.asm [-M--] 13 L:[ 1+33 34/ 34] *(472 / 472b) <EOF> [*] [X]
#include 'in_out.asm'

SECTION .data
    msg: DB 'Введите № студенческого билета: ',0
    rem: DB 'Ваш вариант: ',0

SECTION .bss
    x: RESB 80

SECTION .text
    GLOBAL _start

_start:
    mov eax,msg
    call sprintf

    mov ecx,x
    mov edx, 80
    call sread

    mov eax,x
    call atoi

    xor edx,edx
    mov ebx,20
    div ebx
    inc edx

    mov eax,rem
    call sprintf
    mov eax,edx
    call iprintLF

    call quit
```

Рис. 2.17: Программа из листинга 7.4

Создадим исполняемый файл и запустим его.

```
[aeaskerov@fedora lab07]$ nasm -f elf variant.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o variant variant.o
[aeaskerov@fedora lab07]$ ./variant
Введите № студенческого билета:
1132226538
Ваш вариант: 19
[aeaskerov@fedora lab07]$
```

Рис. 2.18: Результат работы программы variant

Ответы на вопросы:

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

```
mov eax,rem
call sprint
```

Рис. 2.19: Вывод на экран сообщения ‘Ваш вариант:’

2. Для чего используются следующие инструкции? Nasm mov ecx, x mov edx, 80 call sread.

Для того чтобы пользователь ввёл с клавиатуры сообщение размером не больше 80 байт, то есть студенческий номер, в переменную x.

3. Для чего используется инструкция “call atoi”?

Для того чтобы можно было оперировать не с символами из ASCII, стоящими под определёнными номерами, а с числами. “atoi– функция преобразует ascii-код символа в целое число и записывает результат в регистр eax, перед вызовом atoi в регистр eax необходимо записать число (mov eax,)”.

4. Какие строки листинга 7.4 отвечают за вычисление варианта?



```
xor edx,edx  
mov ebx,20  
div ebx  
inc edx
```

Рис. 2.20: Строки, отвечающие за вычисление варианта

5. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”?

В регистр edx.

6. Для чего используется инструкция “inc edx”?

Для того чтобы увеличить ответ, то есть остаток от деления номера студенческого билета на двадцать, на единицу в соответствии с условием задания.

7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?

```
mov eax,edx  
call iprintLF
```

Рис. 2.21: Строки, отвечающие за вывод на экран результата вычислений

## 2.3 Задание для самостоятельной работы

1. Написать программу вычисления выражения  $y = f(x)$ . Программа должна выводить выражение для вычисления, выводить запрос на ввод значения  $x$ , вычислять заданное выражение в зависимости от введенного  $x$ , выводить результат вычислений. Вид функции  $f(x)$  выбрать из таблицы 7.3 вариантов заданий в соответствии с номером, полученным при выполнении лабораторной работы. Создайте исполняемый файл и проверьте его работу для значений  $x_1$  и  $x_2$  из 7.3.

Функция 19:  $f(x) = (x^{1/3} + 5)^7$ .  $x_1 = 3; x_2 = 9$ .

```
mc [aeaskerov@fedora]:~/work/arch-pc/lab07
lab7-4.asm [----] 0 L:[ 1+ 1 2/ 42] *(22 / 867b) 0010 0x00A[*][X]
#include 'in_out.asm'

SECTION .data
    expression: DB 'Выражение: f(x)=(x*1/3+5)*7',0h
    request: DB 'Введите x: ',0h
    result: DB 'Результат: ',0h

SECTION .bss
    buf1: RESB 80

SECTION .text
    GLOBAL _start

_start:
    mov eax,expression
    call sprintLF

    mov eax,request
    call sprint

    mov ecx,buf1
    mov edx,80
    call sread ; Вводим x в ecx

    mov eax,result
    call sprint

    xor eax,eax
    mov eax,buf1
    call atoi
    xor edx,edx ; Обнуляем остаток
    mov ebx,3
    div ebx ; Деление x на 3. Деление eax на ebx. Результат в eax

    add eax,5 ; Прибавляем 5 к eax

    mov ebx,7
    mul ebx ; Умножение на 7. Результат в eax

    call iprintLF ; Печать eax. Результат вычислений

    call quit
```

1По~щъ 2Сох~ть 3Блок 4Замена 5Копия 6Пе~ть 7Поиск 8Уда~ть 9МенюМС10Выход

Рис. 2.22: Программа lab7-4 для самостоятельной работы

```
[aeaskerov@fedora lab07]$ nasm -f elf lab7-4.asm
[aeaskerov@fedora lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[aeaskerov@fedora lab07]$ ./lab7-4
Выражение:  $f(x) = (x * 1/3 + 5) * 7$ 
Введите x: 3
Результат: 42
[aeaskerov@fedora lab07]$ ./lab7-4
Выражение:  $f(x) = (x * 1/3 + 5) * 7$ 
Введите x: 9
Результат: 56
[aeaskerov@fedora lab07]$
```

Рис. 2.23: Результат работы программы lab7-4

## **3 Выводы**

Освоены арифметические инструкции языка ассемблера NASM.