

# **Отчёта по лабораторной работе №6**

**Арифметические операции в NASM.**

Джаллох Ишмаил

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
3.1	Символьные и численные данные в NASM . . . . .	6
3.2	Выполнение арифметических операций в NASM . . . . .	12
3.3	Ответы на вопросы по программе . . . . .	15
3.4	Задание для самостоятельной работы . . . . .	16
<b>4</b>	<b>Выводы</b>	<b>19</b>

## Список иллюстраций

3.1	Создаем каталог с помощью команды <code>mkdir</code> и файл с помощью команды <code>touch</code> . . . . .	6
3.2	Заполняем файл . . . . .	7
3.3	Запускаем файл и смотрим на его работу . . . . .	7
3.4	Изменяем файл . . . . .	8
3.5	Запускаем файл и смотрим на его работу . . . . .	8
3.6	Создаем файл . . . . .	8
3.7	Заполняем файл . . . . .	9
3.8	Смотрим на работу программы . . . . .	9
3.9	Изменяем файл . . . . .	10
3.10	Смотрим на работу программы . . . . .	10
3.11	Изменяем файл . . . . .	11
3.12	Смотрим на работу программы . . . . .	11
3.13	Создаем файл . . . . .	12
3.14	Заполняем файл . . . . .	12
3.15	Смотрим на результат работы программы . . . . .	13
3.16	Редактируем файл . . . . .	13
3.17	Смотрим на результат работы программы . . . . .	14
3.18	Создаем файл . . . . .	14
3.19	Заполняем файл . . . . .	14
3.20	Проверяем результат работы программы . . . . .	15
3.21	Создаем файл . . . . .	16
3.22	Заполняем файл . . . . .	17
3.23	Проверяем работу программы . . . . .	17
3.24	Проверяем работу программы . . . . .	18

# 1 Цель работы

Освоить арифметических инструкций языка ассемблера NASM и написать программы для вычисления арифметических выражений с неизвестной.

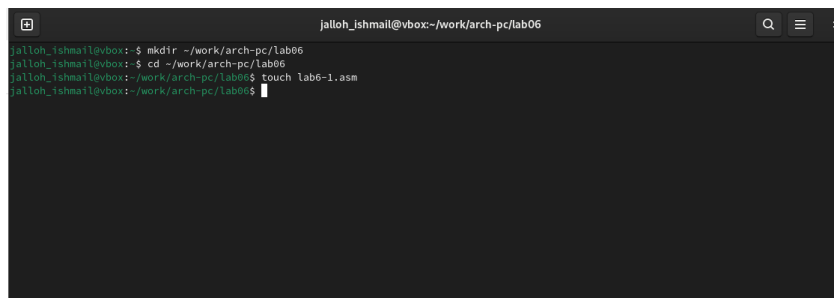
## 2 Задание

Написать программы для решения выражений.

## 3 Выполнение лабораторной работы

### 3.1 Символьные и численные данные в NASM

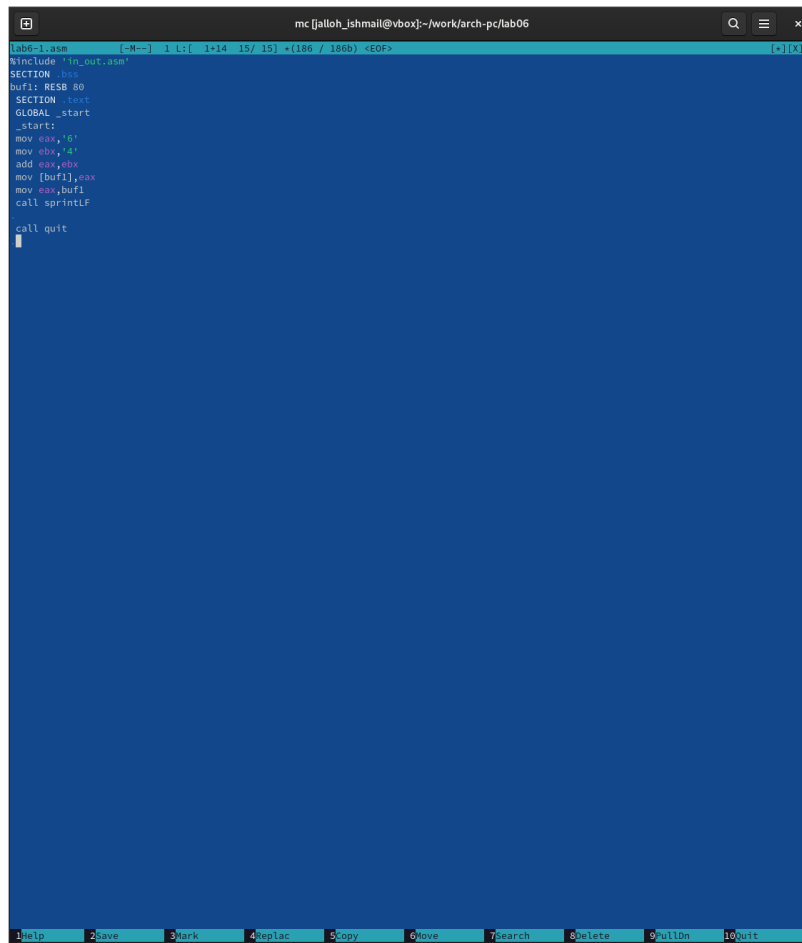
Создаем каталог для программ ЛБ6, и в нем создаем файл (рис. fig. 3.1).

A terminal window with a dark background. The title bar reads 'jalloh\_ishmail@vbox:~/work/arch-pc/lab06'. The terminal shows the following commands and their outputs:

```
jalloh_ishmail@vbox:~$ mkdir -p /work/arch-pc/lab06
jalloh_ishmail@vbox:~$ cd /work/arch-pc/lab06
jalloh_ishmail@vbox:~/work/arch-pc/lab06$ touch lab6-1.asm
jalloh_ishmail@vbox:~/work/arch-pc/lab06$
```

Рис. 3.1: Создаем каталог с помощью команды `mkdir` и файл с помощью команды `touch`

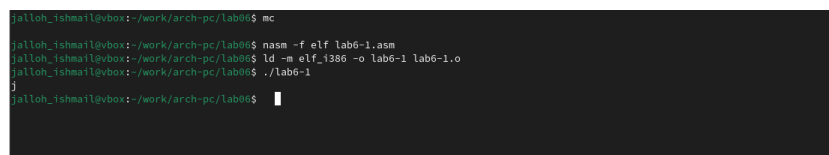
Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 6.1 (рис. fig. 3.2).



```
lab6-1.asm [-M--] 1 L: 1+14 15/ 15) + (186 / 186b) <EOF>
#include "in_out.asm"
SECTION .bss
buf: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, 10
mov ebx, 14
add eax, ebx
mov [buf], eax
mov ecx, buf
call sprintf
call quit
```

Рис. 3.2: Заполняем файл

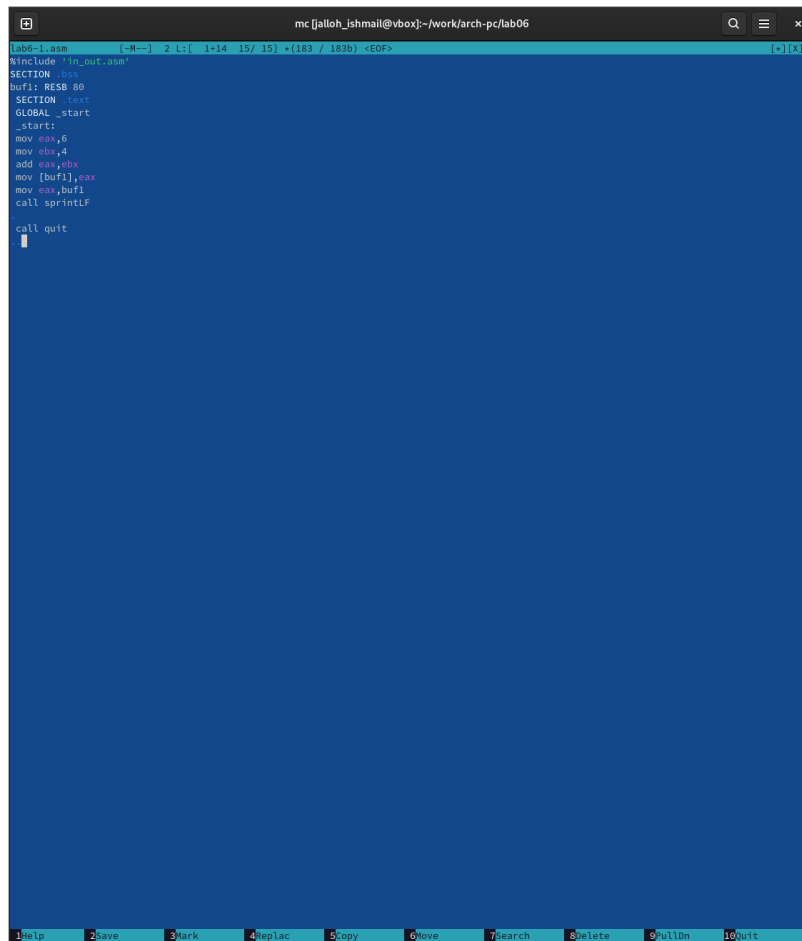
Создаем исполняемый файл и запускаем его (рис. fig. 3.3).



```
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ mc
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ./lab6-1
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.3: Запускаем файл и смотрим на его работу

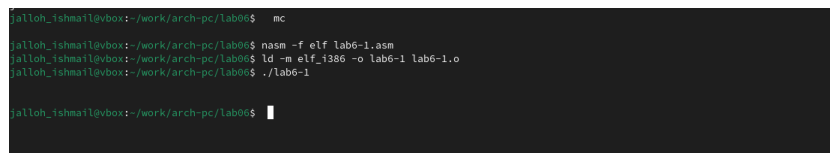
Снова открываем файл для редактирования и убираем кавычки с числовых значений (рис. fig. 3.4).



```
mc [jalloh_ishmail@vbox:~/work/arch-pc/lab06]
lab6-1.asm [-M--] 2 L: 1+14 15/ 15) + (183 / 183b) <EOF>
#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 ecx,buf1
call sprintf
call quit
```

Рис. 3.4: Изменяем файл

Создаем исполняемый файл и запускаем его (рис. fig. 3.5).



```
jalloh_ishmail@vbox:~/work/arch-pc/lab06$ mc
jalloh_ishmail@vbox:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
jalloh_ishmail@vbox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
jalloh_ishmail@vbox:~/work/arch-pc/lab06$ ./lab6-1
jalloh_ishmail@vbox:~/work/arch-pc/lab06$
```

Рис. 3.5: Запускаем файл и смотрим на его работу

Создаем новый файл в каталоге (рис. fig. 3.6).

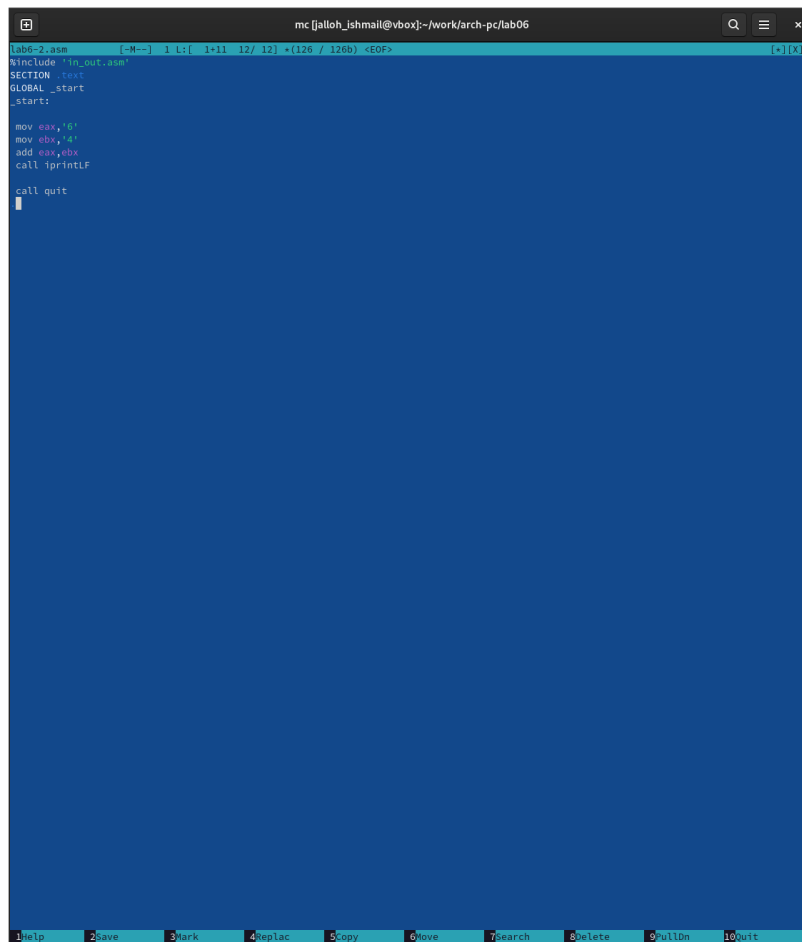


```
jalloh_ishmail@vbox:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-2.asm
jalloh_ishmail@vbox:~/work/arch-pc/lab06$
```

Рис. 3.6: Создаем файл



Заполняем файл в соответствии с листингом 6.2 (рис. fig. 3.7).

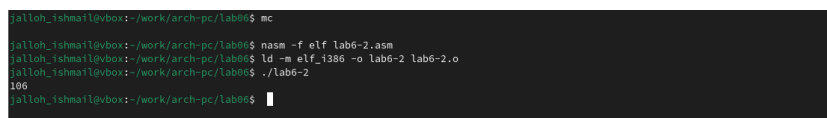


```
lab6-2.asm [~M~] 1 L: 1+11 12/ 12] *(126 / 126b) <EOF>
#include "hw_out.asm"
SECTION .text
GLOBAL _start
_start:

    mov eax, '6'
    mov ebx, '4'
    add eax, ebx
    call 'printf'
    call 'quit'
```

Рис. 3.7: Заполняем файл

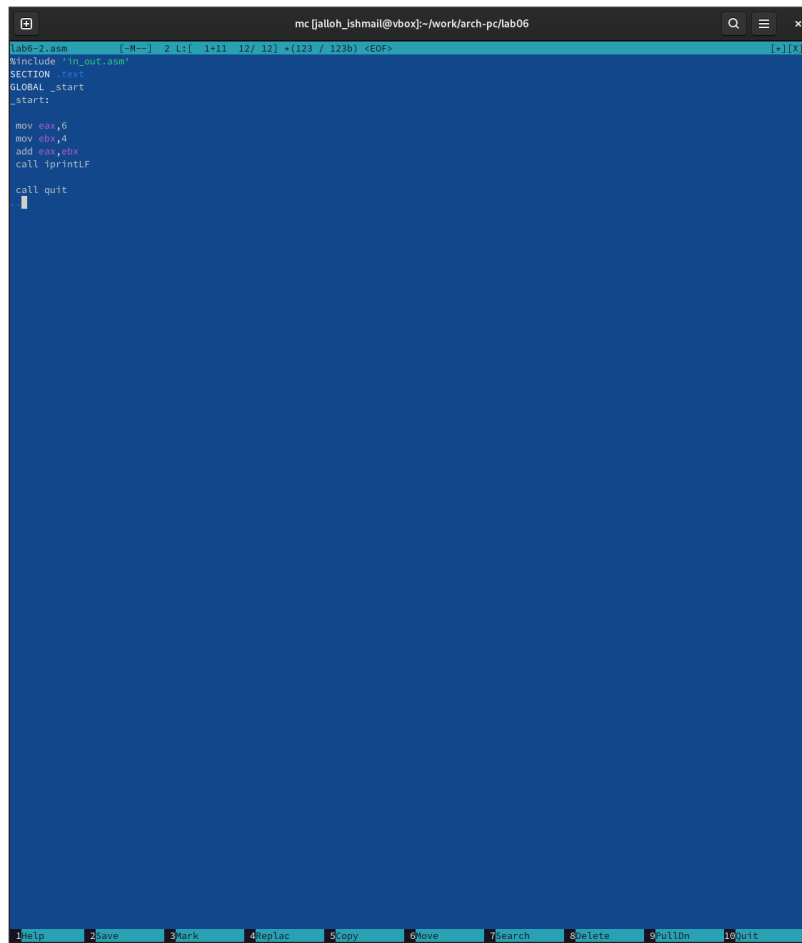
Создаем исполняемый файл и запускаем его (рис. fig. 3.8).



```
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ mc
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ./lab6-2
106
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.8: Смотрим на работу программы

Снова открываем файл для редактирования и убираем кавычки с числовых значений (рис. fig. 3.9).

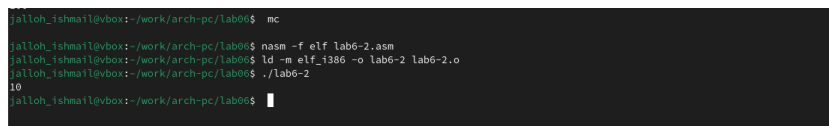


```
mc [jalloh_ishmail@vbox]~/work/arch-pc/lab06
lab6-2.asm [-M--] 2 L: [ 1+11 12/ 12] *(123 / 123b) <EOF>
#include "in_out.asm"
SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Рис. 3.9: Изменяем файл

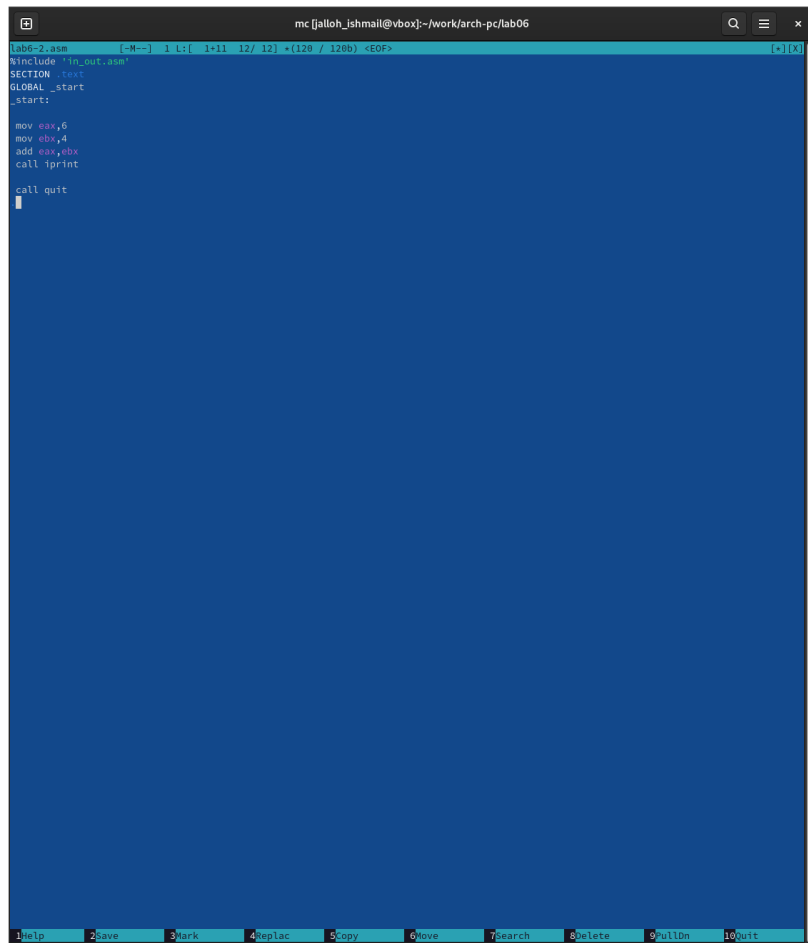
Создаем исполняемый файл и запускаем его (рис. fig. 3.10).



```
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ mc
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ./lab6-2
10
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.10: Смотрим на работу программы

Снова открываем файл для редактирования и меняем iprintLF на iprint (рис. fig. 3.11).

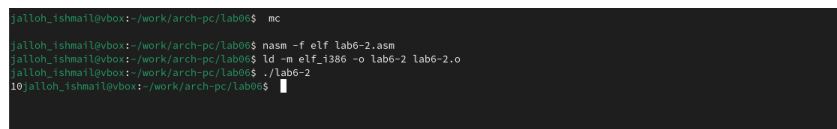


```
mc [alloh_ishmail@vbox]~/work/arch-pc/lab06
lab6-2.asm [-M--] 1 L: [ 1+11 12/ 12] *(120 / 120b) <EOF>
#include "in_out.asm"
SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис. 3.11: Изменяем файл

Создаем исполняемый файл и запускаем его (рис. fig. 3.12).



```
alloh_ishmail@vbox: ~/work/arch-pc/lab06$ mc
alloh_ishmail@vbox: ~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
alloh_ishmail@vbox: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
alloh_ishmail@vbox: ~/work/arch-pc/lab06$ ./lab6-2
10: alloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.12: Смотрим на работу программы

Вывод функций `iprintLF` и `iprint` отличаются только тем, что `LF` переносит на новую строку.

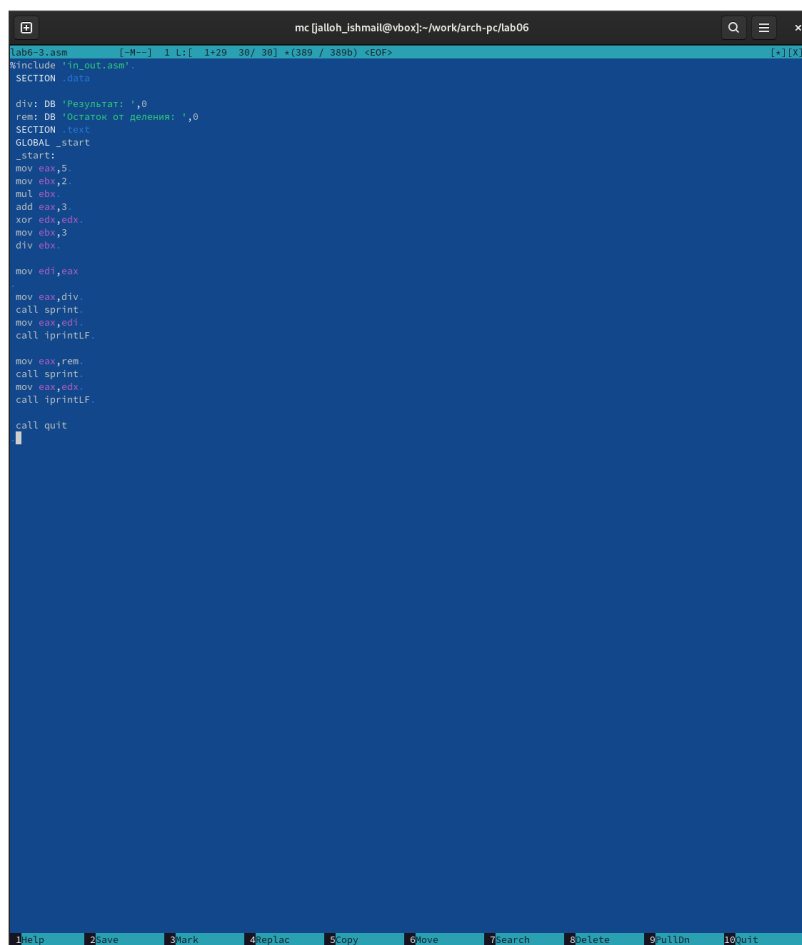
## 3.2 Выполнение арифметических операций в NASM

Создаем новый файл в каталоге (рис. fig. 3.13).

```
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$  
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm  
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.13: Создаем файл

Открываем файл и редактируем в соответствии с листингом 6.3 (рис. fig. 3.14).



```
lab6-3.asm [-M--] 1 L: [ 1+29 30/ 30] *(389 / 389b) <EOF> [x] [D]  
%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 printf  
mov eax,rem  
call sprint  
mov eax,edx  
call printf  
call quit
```

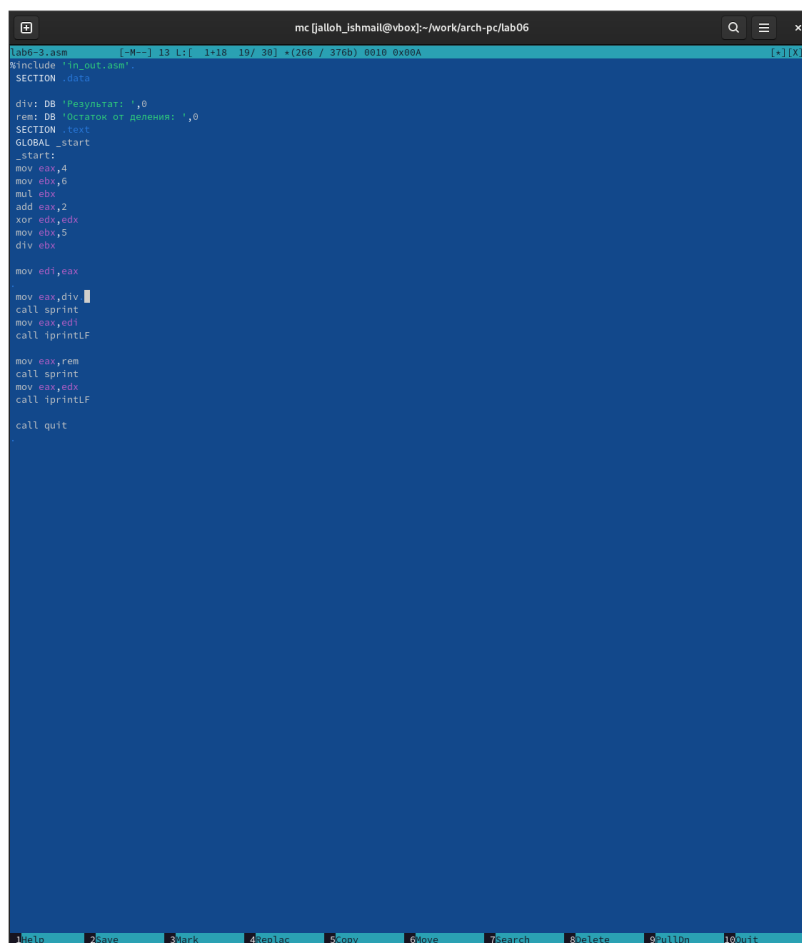
Рис. 3.14: Заполняем файл

Создаем исполняемый файл и запускаем его (рис. fig. 3.15).

```
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ mc
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.15: Смотрим на результат работы программы

Открываем файл и редактируем его для вычисления выражения  $f(x) = (4 \cdot x + 2)/5$  (рис. fig. 3.16).



```
lab6-3.asm  [~M~] 13 L: 1+18 19/ 30 +(266 / 376b) 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
mov eax,edi
call !printf
mov eax,rem
call sprint
mov eax,edi
call !printf
call quit
```

Рис. 3.16: Редактируем файл

Компилируем файл и запускаем программу (рис. fig. 3.17).

```

jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ mc
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$

```

Рис. 3.17: Смотрим на результат работы программы

Создаем новый файл в каталоге (рис. fig. 3.18).

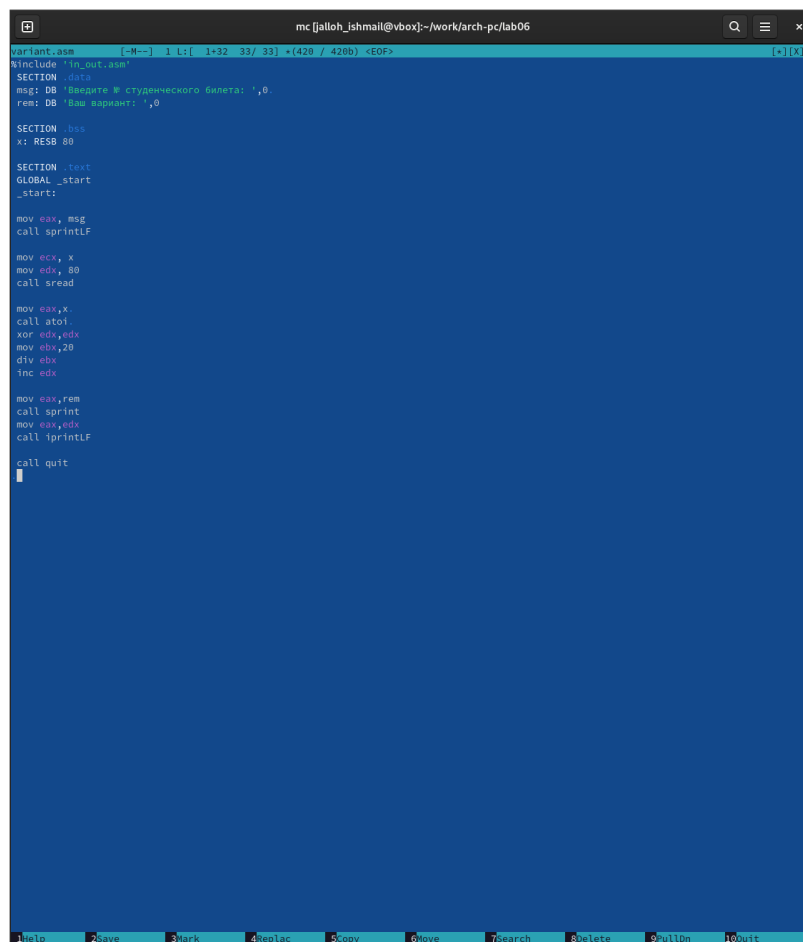
```

jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$

```

Рис. 3.18: Создаем файл

Открываем файл и редактируем в соответствии с листингом 6.4 (рис. fig. 3.19).



```

variant.asm  [-M--]  1 L: [ 1+32  33/ 33] *(420 / 420b) <EOF>
#include "in_out.asm"
SECTION .data
msg: DB "Ваше имя студентского 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 read

mov eax, x
call atoi
xor edx, edx
mov ebx, 20
div ebx
inc edx

mov eax, rem
call sprintf
mov ebx, edx
call sprintf

call quit

```

Рис. 3.19: Заполняем файл

Компилируем файл и запускаем его (рис. fig. 3.20).

```
jalloh_ishmail@box: /work/arch-pc/lab06$ mc
jalloh_ishmail@box: /work/arch-pc/lab06$ nasm -f elf variant.asm
jalloh_ishmail@box: /work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
jalloh_ishmail@box: /work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032239614
Ваш вариант: 15
jalloh_ishmail@box: /work/arch-pc/lab06$
```

Рис. 3.20: Проверяем результат работы программы

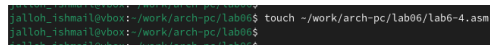
### 3.3 Ответы на вопросы по программе

1. Строка “mov eax,rem” и строка “call sprint” отвечают за вывод на экран сообщения ‘Ваш вариант:’.
2. Эти инструкции используются для чтения строки с вводом данных от пользователя. Начальный адрес строки сохраняется в регистре esx, а количество символов в строке (максимальное количество символов, которое может быть считано) сохраняется в регистре edx. Затем вызывается процедура sread, которая выполняет чтение строки.
3. Инструкция “call atoi” используется для преобразования строки в целое число. Она принимает адрес строки в регистре eax и возвращает полученное число в регистре eax.
4. Строка “xor edx,edx” обнуляет регистр edx перед выполнением деления. Строка “mov ebx,20” загружает значение 20 в регистр ebx. Строка “div ebx” выполняет деление регистра eax на значение регистра ebx с сохранением частного в регистре eax и остатка в регистре edx.
5. Остаток от деления записывается в регистр edx.
6. Инструкция “inc edx” используется для увеличения значения в регистре edx на 1. В данном случае, она увеличивает остаток от деления на 1.

7. Строка “mov eax,edx” передает значение остатка от деления в регистр eax. Строка “call iprintLF” вызывает процедуру iprintLF для вывода значения на экран вместе с переводом строки.

### 3.4 Задание для самостоятельной работы

Создаем новый файл в каталоге (рис. fig. 3.21).

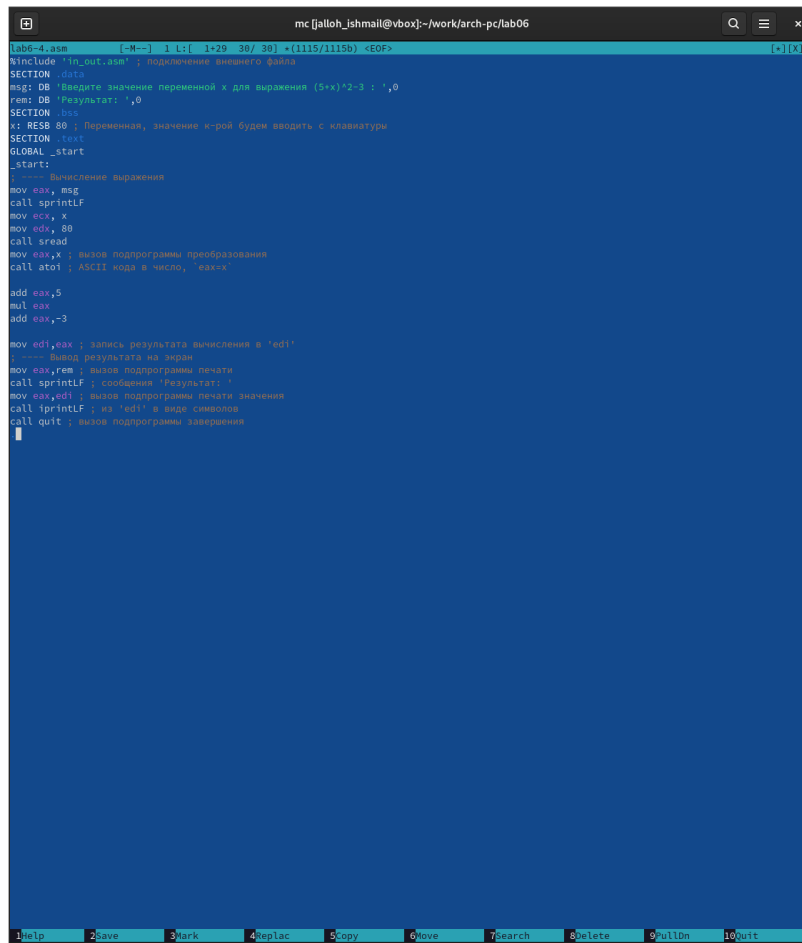


```
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-4.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.21: Создаем файл

Открываем его и заполняем, чтобы решалось выражение  $f(x) = (5 + x)^2 - 3$  (рис. fig. 3.22).





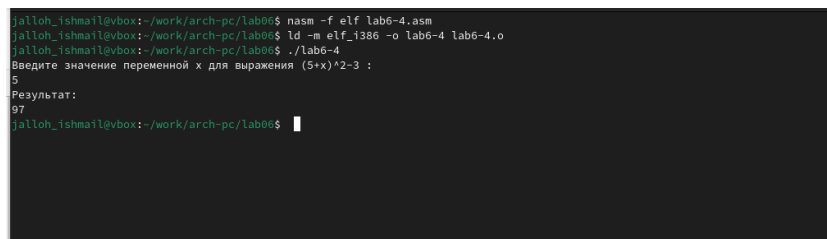
```
lab6-4.asm [-M--] 1 L: 1-29 30/ 30) +(1115/1115b) <EOF>
#include "in_out.asm" ; подключение внешнего файла
SECTION .data
msg: DB 'Введите значение переменной x для выражения (5+x)^2-3 : ',0
rem: DB 'Результат: ',0
SECTION .bss
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры
SECTION .text
GLOBAL _start
_start:
; ----- Вычисление выражения
mov eax, msg
call sprintf
mov ecx, x
mov edx, 80
call sread
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax:x'

add eax, 5
mul eax
add eax, -3

mov edi, eax ; запись результата вычисления в 'edi'
; ----- Вывод результата на экран
mov eax, rem ; вызов подпрограммы печати
call sprintf ; сообщением 'Результат: '
mov eax, edi ; вызов подпрограммы печати значения
call iprintf ; из 'edi' в виде символов
call iquit ; вызов подпрограммы завершения
```

Рис. 3.22: Заполняем файл

Компилируем программу и проверяем для  $x=5$  (рис. fig. 3.23).



```
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$ ./lab6-4
Введите значение переменной x для выражения (5+x)^2-3 :
5
Результат:
97
jalloh_ishmail@vbox: ~/work/arch-pc/lab06$
```

Рис. 3.23: Проверяем работу программы

Компилируем программу и проверяем для  $x=1$  (рис. fig. 3.24).

```
jalloh_ishmail@vbox:~/work/arch-pc/lab6$ nasm -f elf lab6-4.asm
jalloh_ishmail@vbox:~/work/arch-pc/lab6$ ld -m elf_i386 -o lab6-4 lab6-4.o
jalloh_ishmail@vbox:~/work/arch-pc/lab6$ ./lab6-4
Введите значение переменной x для выражения (5+x)^2-3 :
1
Результат:
33
jalloh_ishmail@vbox:~/work/arch-pc/lab6$
```

Рис. 3.24: Проверяем работу программы

## 4 Выводы

Мы приобрели навыки создания исполнительных файлов для решения выражений и освоили арифметические инструкции в NASM.