

# **Лабораторная работа номер 6**

**Архитектура компьютера**

Титков Ярослав Максимович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
<b>5</b>	<b>4.1 Данные в NASM</b>	<b>10</b>
<b>6</b>	<b>4.2 Выполнение арифметических операций в NASM</b>	<b>17</b>
<b>7</b>	<b>4.2.1 Ответы на вопросы:</b>	<b>21</b>
<b>8</b>	<b>4.3 Задания для самостоятельной работы:</b>	<b>22</b>
<b>9</b>	<b>Выводы</b>	<b>24</b>

## Список иллюстраций

5.1	Создал каталог для программ лабораторной работы номер 6, перешёл в него и создал файл lab6-1.asm . . . . .	10
5.2	Ввёл в файл lab6-1.asm текст программы из листинга 6.1 . . . . .	11
5.3	Создал исполняемый файл и запустил его . . . . .	12
5.4	Изменил текст программы . . . . .	13
5.5	Повторно создал и запустил . . . . .	14
5.6	Создал файл lab6-2.asm ввёл туда текст и запустил программу . . .	15
5.7	Изменил программу и повторно запустил . . . . .	16
6.1	Создал файл lab6-3.asm и скопировал туда программу . . . . .	17
6.2	Запустил файл lab6-3 . . . . .	18
6.3	Изменил текст для вычисления формулы $f(x) = (4 \cdot x + 2)/5$ . . . . .	19
6.4	Запустил программу . . . . .	19
6.5	Аналогично создал файл variant.asm и ввёл данные . . . . .	19
6.6	Запустил программу . . . . .	20
8.1	Создал программу . . . . .	22
8.2	Проверил её работоспособность . . . . .	23

## **Список таблиц**

# 1 Цель работы

Цель данной лабораторной работы - освоение арифметических инструкций языка ассемблера NASM.

## **2 Задание**

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

### 3 Теоретическое введение

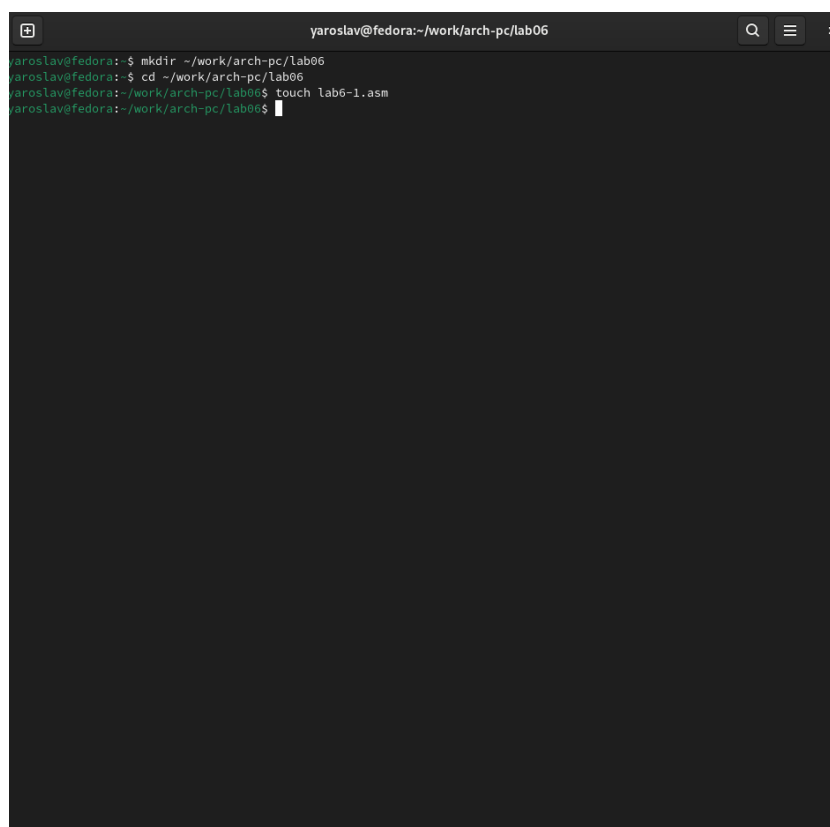
Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. - Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: `mov ax,bx`. - Непосредственная адресация – значение операнда задается непосредственно в команде, Например: `mov ax,2`. - Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию. Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от American Standard Code for Information Interchange (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). Поэтому, например, чтобы вывести число, надо предварительно преобразовать его цифры в ASCII-коды этих цифр и выводить на экран эти коды, а не само число. Если же выводить число на экран непосредственно, то экран воспримет его не как число, а как последовательность ASCII-символов – каждый байт числа будет воспринят как один ASCII-символ – и выведет на экран эти символы. Аналогичная ситуация происходит и при вводе данных с клавиатуры. Введенные данные будут представлять собой символы, что сделает невозможным получение корректного результата при выполнении над ними

арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно



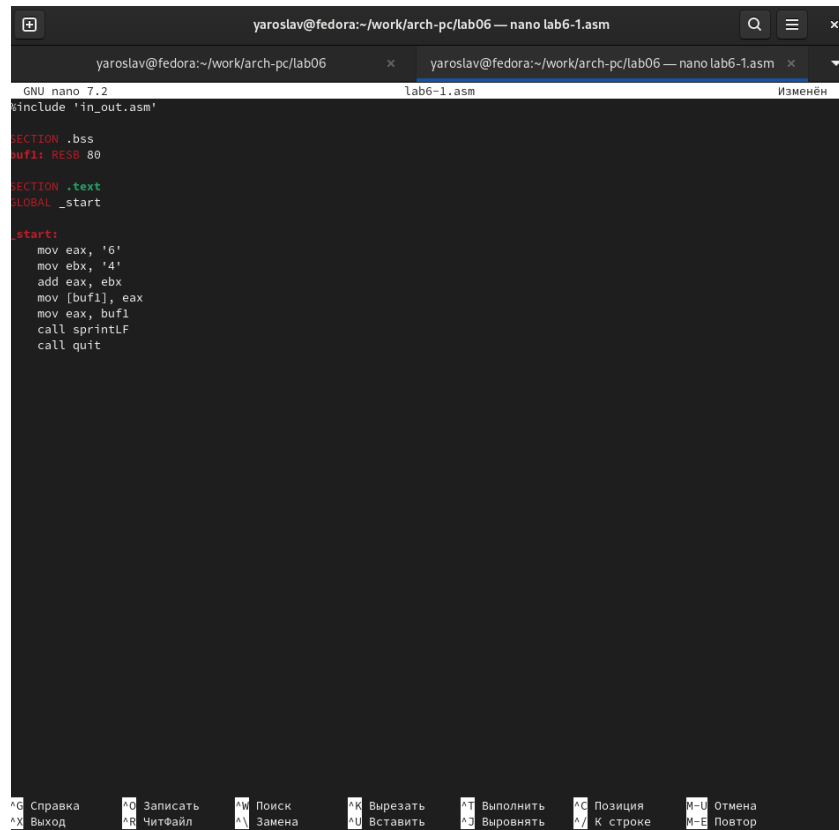
## **4 Выполнение лабораторной работы**

## 5 4.1 Данные в NASM



```
yaroslav@fedora:~/work/arch-pc/lab06$ mkdir ~/work/arch-pc/lab06
yaroslav@fedora:~/work/arch-pc/lab06$ cd ~/work/arch-pc/lab06
yaroslav@fedora:~/work/arch-pc/lab06$ touch lab6-1.asm
yaroslav@fedora:~/work/arch-pc/lab06$
```

Рис. 5.1: Создал каталог для программ лабораторной работы номер 6, перешёл в него и создал файл lab6-1.asm



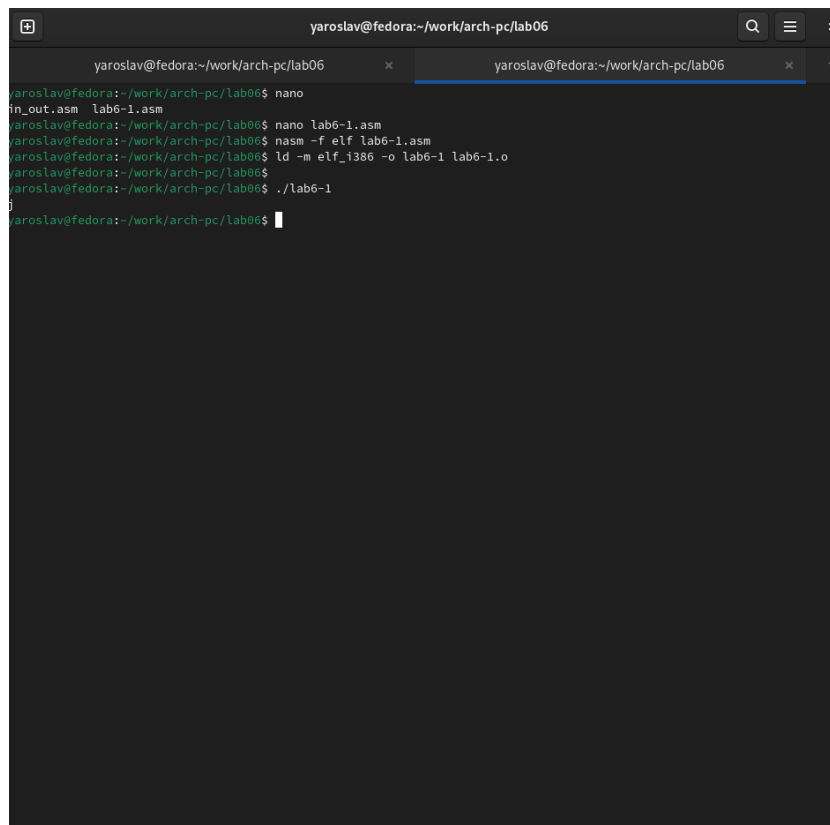
```
GNU nano 7.2 lab6-1.asm
#include "in_out.asm"

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start

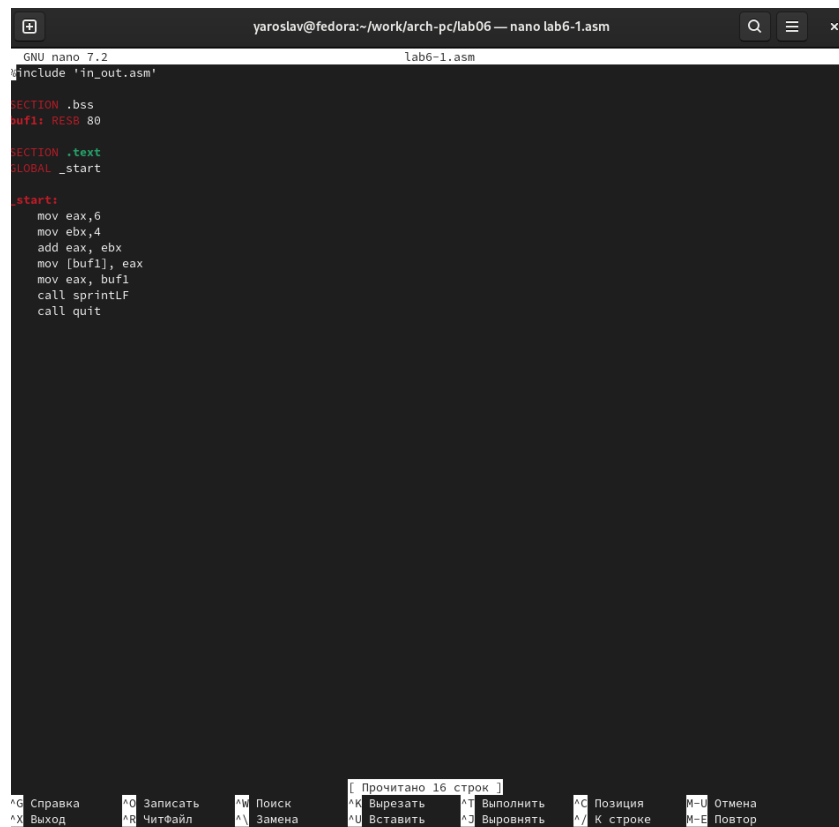
_start:
    mov eax, '6'
    mov ebx, '4'
    add eax, ebx
    mov [buf1], eax
    mov eax, buf1
    call sprintf
    call _exit
```

Рис. 5.2: Ввёл в файл lab6-1.asm текст программы из листинга 6.1



```
yaroslav@fedora:~/work/arch-pc/lab06
yaroslav@fedora:~/work/arch-pc/lab06
in_out.asm lab6-1.asm
yaroslav@fedora:~/work/arch-pc/lab06$ nano lab6-1.asm
yaroslav@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
yaroslav@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
yaroslav@fedora:~/work/arch-pc/lab06$ ./lab6-1
yaroslav@fedora:~/work/arch-pc/lab06$
```

Рис. 5.3: Создал исполняемый файл и запустил его



```
GNU nano 7.2 lab6-1.asm
#include 'in_out.asm'

SECTION .bss
buf1: RESB 80

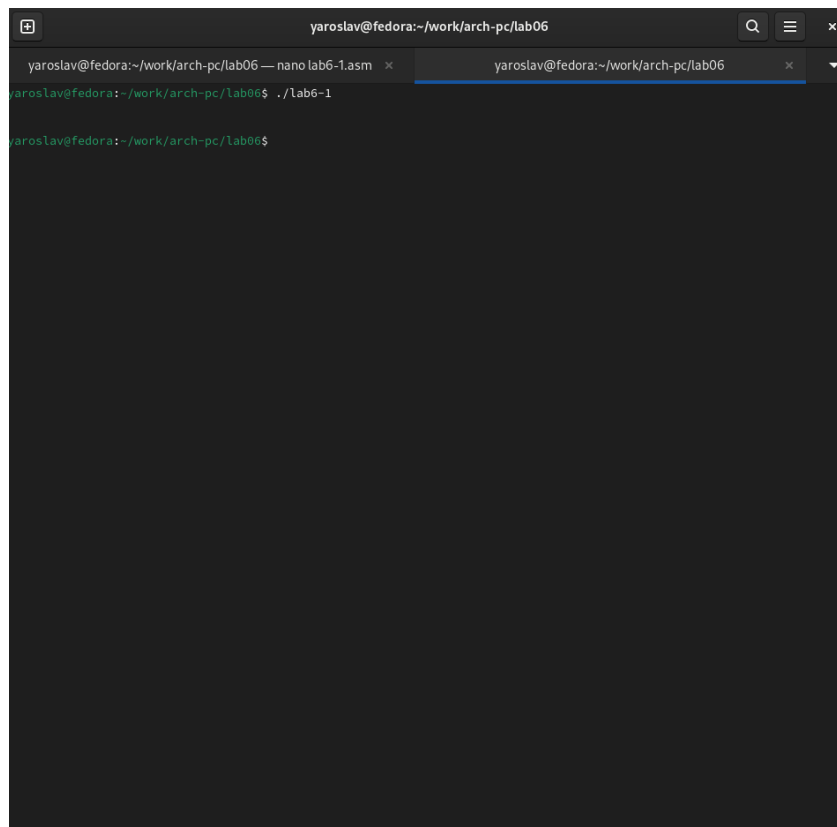
SECTION .text
GLOBAL _start

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

Прочитано 16 строк

Alt G	Справка	Alt O	Записать	Alt W	Поиск	Alt K	Вырезать	Alt T	Выполнить	Alt C	Позиция	Alt U	Отмена
Alt X	Выход	Alt R	Читфайл	Alt A	Замена	Alt V	Вставить	Alt J	Выровнять	Alt I	К строке	Alt E	Повтор

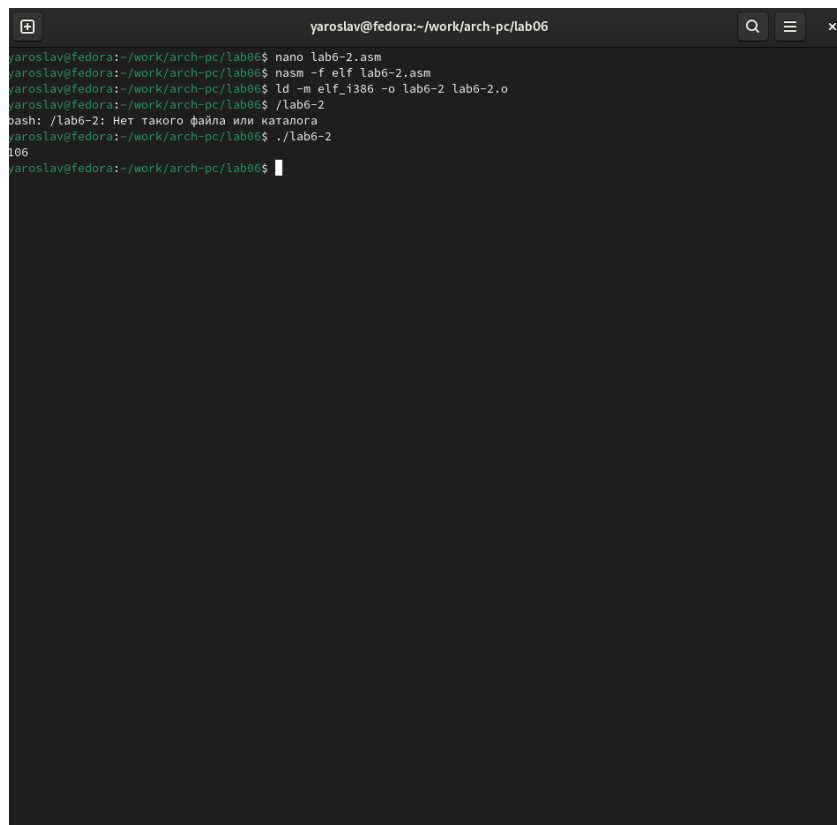
Рис. 5.4: Изменил текст программы



The image shows a terminal window with a dark background. The title bar at the top reads "yaroslav@fedora: ~/work/arch-pc/lab06". There are two tabs: "nano lab6-1.asm" and "yaroslav@fedora: ~/work/arch-pc/lab06". The terminal content shows the prompt "yaroslav@fedora: ~/work/arch-pc/lab06\$" followed by the command "./lab6-1" on the next line. The prompt then changes to "yaroslav@fedora: ~/work/arch-pc/lab06\$" again, indicating the command has been executed.

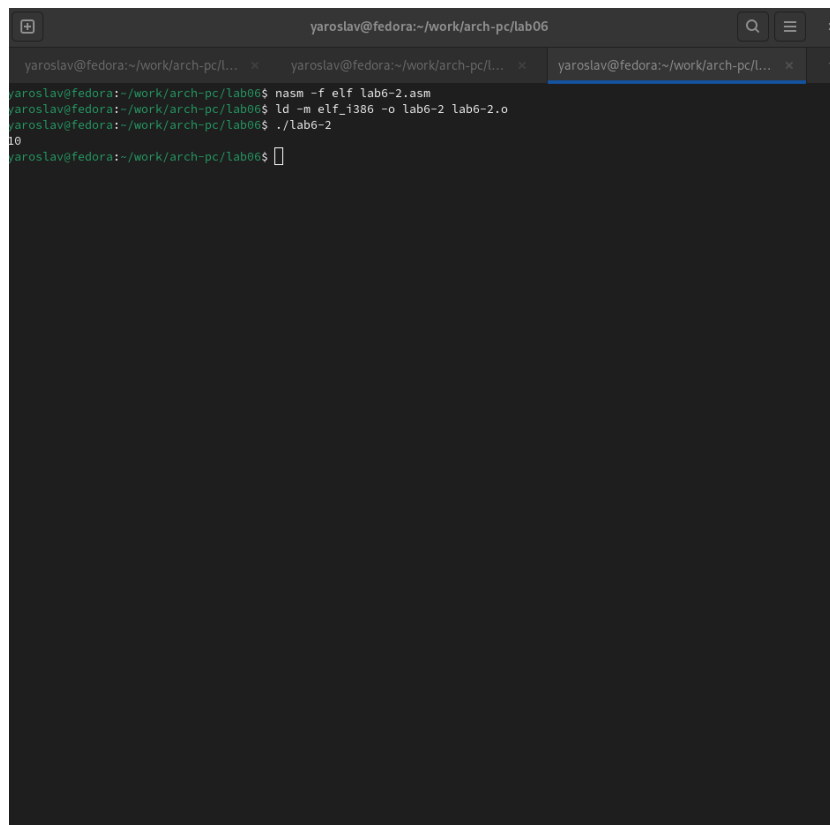
```
yaroslav@fedora: ~/work/arch-pc/lab06 — nano lab6-1.asm ×
yaroslav@fedora: ~/work/arch-pc/lab06 ×
yaroslav@fedora: ~/work/arch-pc/lab06$ ./lab6-1
yaroslav@fedora: ~/work/arch-pc/lab06$
```

Рис. 5.5: Повторно создал и запустил

A terminal window titled 'yaroslav@fedora:~/work/arch-pc/lab06' with search, menu, and close icons. The terminal shows a series of commands: 'nano lab6-2.asm' to create a file, 'nasm -f elf lab6-2.asm' to assemble it, and 'ld -m elf\_i386 -o lab6-2 lab6-2.o' to link it. An attempt to run './lab6-2' fails with the message 'bash: ./lab6-2: Нет такого файла или каталога'. The prompt returns to 'yaroslav@fedora:~/work/arch-pc/lab06\$'.

```
yaroslav@fedora:~/work/arch-pc/lab06$ nano lab6-2.asm
yaroslav@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
yaroslav@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
yaroslav@fedora:~/work/arch-pc/lab06$ ./lab6-2
bash: ./lab6-2: Нет такого файла или каталога
yaroslav@fedora:~/work/arch-pc/lab06$ ./lab6-2
106
yaroslav@fedora:~/work/arch-pc/lab06$
```

Рис. 5.6: Создал файл lab6-2.asm ввёл туда текст и запустил программу



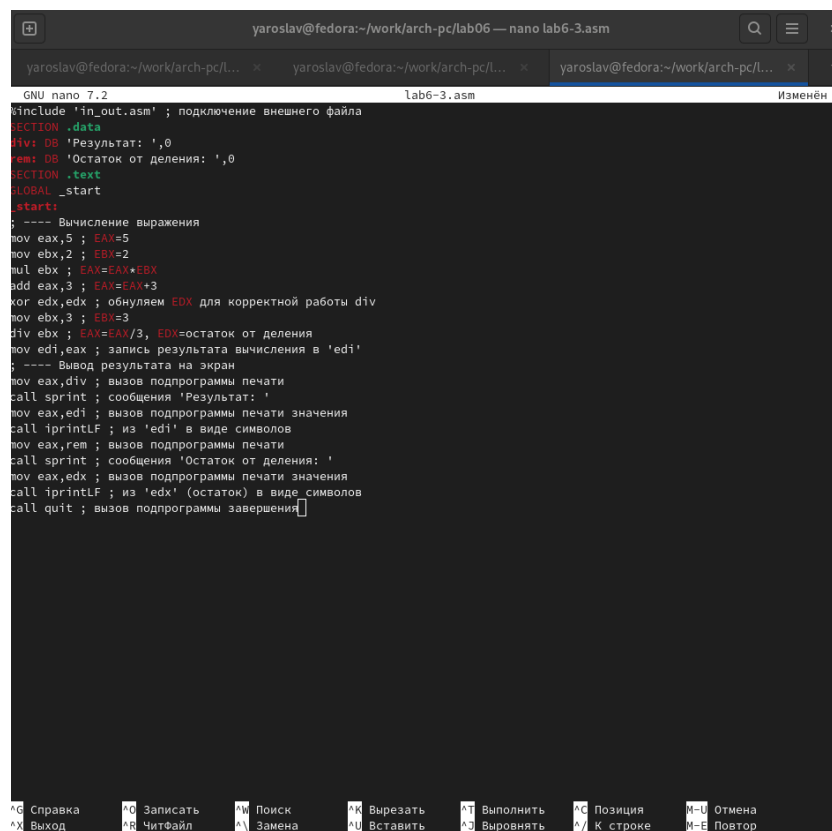
A terminal window with a dark background and light-colored text. The window title is 'yaroslav@fedora:~/work/arch-pc/lab06'. The terminal shows the following commands and output:

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

Рис. 5.7: Изменил программу и повторно запустил

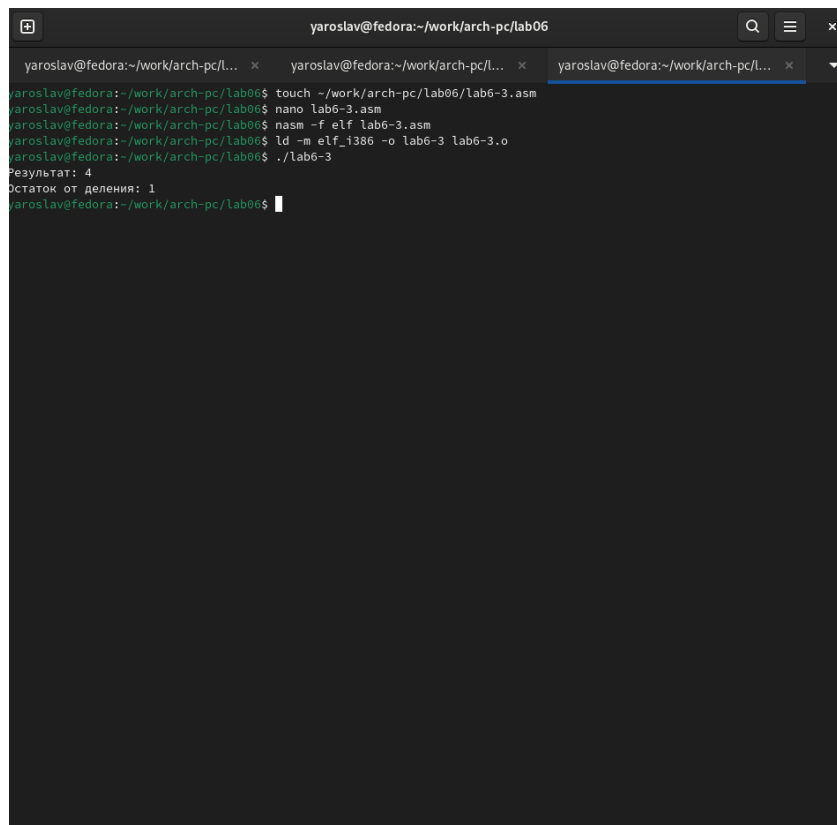


## 6 4.2 Выполнение арифметических операций в NASM



```
GNU nano 7.2 lab6-3.asm
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,5 ; EAX=5
mov ebx,2 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,3 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,3 ; EBX=3
div ebx ; EAX=EAX/3, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 6.1: Создал файл lab6-3.asm и скопировал туда программу



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

Рис. 6.2: Запустил файл lab6-3

```

GNU nano 7.2                                yaroslav@fedora:~/work/arch-pc/lab06 — nano lab6-3.asm
#include 'in_out.asm'

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

SECTION .text
GLOBAL _start

_start:
; ---- Вычисление выражения (4 * 6 + 2) / 5
mov eax, 4 ; EAX = 4
mov ebx, 6 ; EBX = 6
mul ebx ; EAX = EAX * EBX (4 * 6)
add eax, 2 ; EAX = EAX + 2 (4 * 6 + 2)
xor edx, edx ; обнуляем EDX для корректной работы div
mov ebx, 5 ; EBX = 5
div ebx ; EAX = EAX / 5, EDX = остаток от деления
mov edi, eax ; запись результата вычисления в 'edi'

; ---- Вывод результата на экран
mov eax, div ; вызов подпрограммы печати
call sprintf ; сообщения 'Результат: '
mov eax, edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax, rem ; вызов подпрограммы печати
call sprintf ; сообщения 'Остаток от деления: '
mov eax, edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

Рис. 6.3: Изменил текст для вычисления формулы  $f(x) = (4 \cdot 6 + 2)/5$ .

```

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

```

Рис. 6.4: Запустил программу

```

GNU nano 7.2                                variant.asm                                Изменён
#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 ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax=x'
xor edx, edx
mov ebx, 20
div ebx
inc ebx
mov eax, rem
call sprintf
mov eax, edx
call iprintLF
call quit

```

Рис. 6.5: Аналогично создал файл variant.asm и ввёл данные

```
yaroslav@fedora:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
yaroslav@fedora:~/work/arch-pc/lab06$ nano variant.asm
yaroslav@fedora:~/work/arch-pc/lab06$ nasm -f elf variant.asm
yaroslav@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
yaroslav@fedora:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
113232434
Ваш вариант: 15
yaroslav@fedora:~/work/arch-pc/lab06$
```

Рис. 6.6: Запустил программу

## 7 4.2.1 Ответы на вопросы:

1: `mov eax, rem call sprint`

2: Эти инструкции используются для чтения строки ввода пользователя и сохранения её в переменную `x`.

3: Инструкция `call atoi` используется для преобразования строки, хранящейся в `eax`, в целое число.

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

5: Остаток от деления записывается в регистр `edx`.

6: Инструкция `inc edx` используется для увеличения остатка от деления на 1, чтобы получить номер варианта.

7: `mov eax, edx call iprintLF`

## 8 4.3 Задания для самостоятельной работы:

Написать программу вычисления выражения  $y = 2x + 5$ . Программа должна вывести 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 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы.

```
GNU nano 7.2                               lab6-4.asm                               Изменён
%include 'in_out.asm'

SECTION .data
expr: DB 'Вычисление выражения y = 2x + 5', 0xA, 0
prompt: DB 'Введите значение x: ', 0
result: DB 'Результат: ', 0

SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start

_start:
; Вывод выражения для вычисления
mov eax, expr
call sprintf

; Запрос на ввод значения x
mov eax, prompt
call sprintf

; Чтение значения x
mov ecx, x
mov edx, 80
call sread

; Преобразование строки в число
mov eax, x
call atoi
mov ebx, eax ; сохраняем значение x в ebx

; Вычисление выражения y = 2x + 5
mov eax, 2
mul ebx      ; eax = 2 * x
add eax, 5   ; eax = 2 * x + 5

; Вывод результата
mov edi, eax ; сохраняем результат в edi
mov eax, result
call sprintf
mov eax, edi
call iprintf
```

Рис. 8.1: Создал программу

```
yaroslav@fedora: ~/work/arch-pc/lab06$ touch lab6-4.asm
yaroslav@fedora: ~/work/arch-pc/lab06$ nano lab6-4.asm
yaroslav@fedora: ~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
yaroslav@fedora: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
yaroslav@fedora: ~/work/arch-pc/lab06$ ./lab6-4
Вычисление выражения  $y = 2x + 5$ 
Введите значение x: 1
Результат: 7
```

Рис. 8.2: Проверил её работоспособность

## 9 Выводы

При выполнении данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.