

# **Отчет по лабораторной работе №6**

**Дисциплина: архитектура компьютера**

Гомазкова Алина

# 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>
4.1	Символьные и численные данные в NASM . . . . .	9
4.2	Выполнение арифметических операций в NASM . . . . .	13
4.3	Ответы на вопросы по программе . . . . .	16
4.4	Выполнение заданий для самостоятельной работы . . . . .	17
<b>5</b>	<b>Выводы</b>	<b>19</b>
	<b>Список литературы</b>	<b>20</b>

## Список иллюстраций

4.1	Рис. 1 Перехожу в каталог . . . . .	9
4.2	Рис. 2 Создание файла . . . . .	9
4.3	Рис. 3 Создание копии файла . . . . .	9
4.4	Рис. 4 Редактирование файла . . . . .	10
4.5	Рис. 5 Запуск исполняемого файла . . . . .	10
4.6	Рис. 6 Редактирование файла . . . . .	10
4.7	Рис. 7 Запуск исполняемого файла . . . . .	11
4.8	Рис. 8 Создание файла . . . . .	11
4.9	Рис. 9 Редактирование файла . . . . .	11
4.10	Рис. 10 Запуск исполняемого файла . . . . .	11
4.11	Рис. 11 Редактирование файла . . . . .	12
4.12	Рис. 12 Запуск исполняемого файла . . . . .	12
4.13	Рис. 13 Редактирование файла . . . . .	12
4.14	Рис. 14 Запуск исполняемого файла . . . . .	13
4.15	Рис. 15 Создание файла . . . . .	13
4.16	Рис. 16 Редактирование файла . . . . .	13
4.17	Рис. 17 Запуск исполняемого файла . . . . .	14
4.18	Рис. 18 Изменение программы . . . . .	14
4.19	Рис. 19 Запуск исполняемого файла . . . . .	14
4.20	Рис. 20 Создание файла . . . . .	15
4.21	Рис. 21 Редактирование файла . . . . .	15
4.22	Рис. 22 Запуск исполняемого файла . . . . .	15
4.23	Рис. 23 Создание файла . . . . .	17
4.24	Рис. 24 Написание программы . . . . .	17
4.25	Рис. 25 Запуск исполняемого файла . . . . .	18
4.26	Рис. 26 Запуск исполняемого файла . . . . .	18

## **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

### 4.1 Символьные и численные данные в NASM

Перехожу в каталог с помощью утилиты `cd` (рис. 1)

```
[alinagomazkova@fedora ~]$ cd ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab06
```

Рис. 4.1: Рис. 1 Перехожу в каталог

С помощью утилиты `touch` создаю файл `lab6-1.asm` (рис. 2)

```
[alinagomazkova@fedora lab06]$ touch lab6-1.asm
[alinagomazkova@fedora lab06]$ ls
lab6-1.asm  presentation  report
```

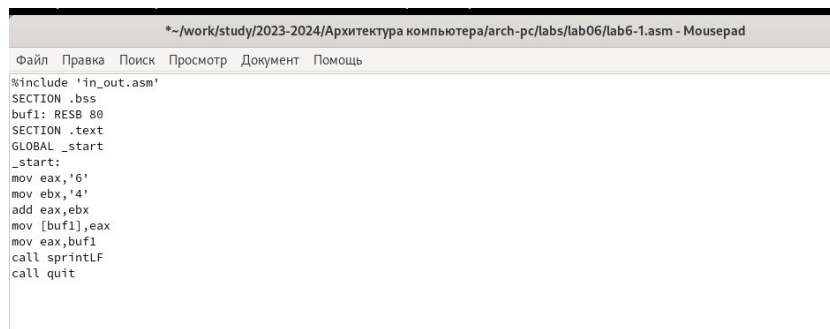
Рис. 4.2: Рис. 2 Создание файла

Копирую в текущий каталог файл `in_out.asm` с помощью утилиты `cp`, т.к. он будет использоваться в других программах (рис. 3)

```
[alinagomazkova@fedora lab06]$ cp ~/Загрузки/in_out.asm in_out.asm
[alinagomazkova@fedora lab06]$ ls
in_out.asm  lab6-1.asm  presentation  report
[alinagomazkova@fedora lab06]$
```

Рис. 4.3: Рис. 3 Создание копии файла

Открываю созданный файл `lab6-1.asm`, вставляю в него программу вывода значения регистра `eax` (рис. 4)

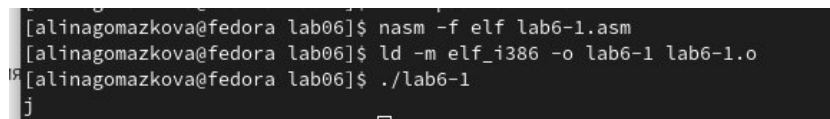


```
*~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/lab6-1.asm - Mousepad
Файл  Правка  Поиск  Просмотр  Документ  Помощь

%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. 4.4: Рис. 4 Редактирование файла

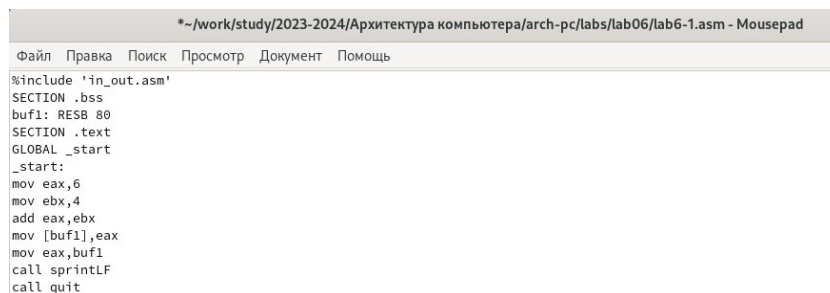
Создаю исполняемый файл программы и запускаю его. Вывод программы: символ j, потому что программа вывела символ, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6 (рис. 5)



```
[alinagomazkova@fedora lab06]$ nasm -f elf lab6-1.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[alinagomazkova@fedora lab06]$ ./lab6-1
j
```

Рис. 4.5: Рис. 5 Запуск исполняемого файла

Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4 (рис. 6)



```
*~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/lab6-1.asm - Mousepad
Файл  Правка  Поиск  Просмотр  Документ  Помощь

%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. 4.6: Рис. 6 Редактирование файла

Создаю новый исполняемый файл программы и запускаю его. Теперь вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран (рис. 7)

```

[alinagomazkova@fedora lab06]$ mousepad lab6-1.asm
[alinagomazkova@fedora lab06]$ nasm -f elf lab6-1.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[alinagomazkova@fedora lab06]$ ./lab6-1
[alinagomazkova@fedora lab06]$

```

Рис. 4.7: Рис. 7 Запуск исполняемого файла

Создаю новый файл lab6-2.asm с помощью утилиты touch (рис. 8)

```

[alinagomazkova@fedora lab06]$ touch lab6-2.asm
[alinagomazkova@fedora lab06]$

```

Рис. 4.8: Рис. 8 Создание файла

Ввожу в файл текст другой программы для вывода значения регистра eax (рис. 9)

```

*~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/lab6-2.asm - Mousepad
Файл  Правка  Поиск  Просмотр  Документ  Помощь
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit

```

Рис. 4.9: Рис. 9 Редактирование файла

Создаю и запускаю исполняемый файл lab6-2. Теперь вывод число 106, потому что программа позволяет вывести именно число, а не символ, хотя все еще происходит именно сложение кодов символов “6” и “4” (рис. 10)

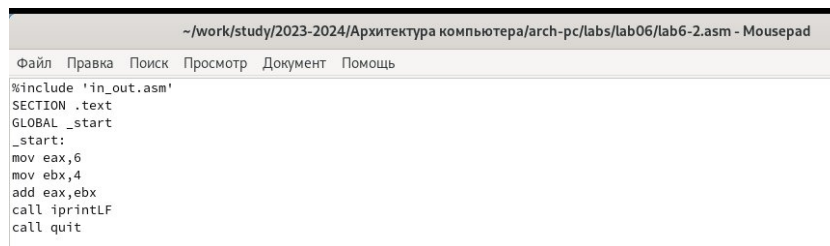
```

[alinagomazkova@fedora lab06]$ nasm -f elf lab6-2.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[alinagomazkova@fedora lab06]$ ./lab6-2
106
[alinagomazkova@fedora lab06]$

```

Рис. 4.10: Рис. 10 Запуск исполняемого файла

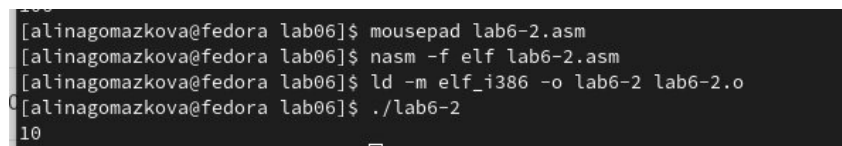
Заменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4 (рис. 11)



```
~/.work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/lab6-2.asm - Mousepad
Файл  Правка  Поиск  Просмотр  Документ  Помощь
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Рис. 4.11: Рис. 11 Редактирование файла

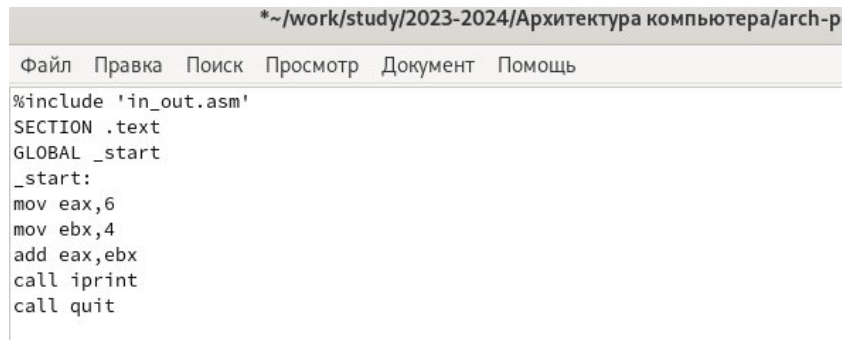
Создаю и запускаю новый исполняемый файл. Теперь программа складывает не соответствующие символам коды в системе ASCII, а сами числа, поэтому вывод 10 (рис. 12)



```
[alinagomazkova@fedora lab06]$ mousepad lab6-2.asm
[alinagomazkova@fedora lab06]$ nasm -f elf lab6-2.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[alinagomazkova@fedora lab06]$ ./lab6-2
10
```

Рис. 4.12: Рис. 12 Запуск исполняемого файла

Заменяю в тексте программы функцию iprintLF на iprint (рис. 13)



```
*~/.work/study/2023-2024/Архитектура компьютера/arch-p
Файл  Правка  Поиск  Просмотр  Документ  Помощь
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис. 4.13: Рис. 13 Редактирование файла

Создаю и запускаю новый исполняемый файл. Вывод не изменился, потому что символ переноса строки не отображался, когда программа исполнялась с функцией iprintLF, а iprint не добавляет к выводу символ переноса строки, в отличие от iprintLF (рис. 14)

```
[alinagomazkova@fedora lab06]$ nasm -f elf lab6-2.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[alinagomazkova@fedora lab06]$ ./lab6-2
10[alinagomazkova@fedora lab06]$ ls
```

Рис. 4.14: Рис. 14 Запуск исполняемого файла

## 4.2 Выполнение арифметических операций в NASM

Создаю файл lab6-3.asm с помощью утилиты touch (рис. 15)

```
[alinagomazkova@fedora lab06]$ touch lab6-3.asm
[alinagomazkova@fedora lab06]$
```

Рис. 4.15: Рис. 15 Создание файла

Ввожу в созданный файл текст программы для вычисления значения выражения  $f(x) = (5 * 2 + 3)/3$  (рис. 16)

```
~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/lab6-3.asm - Mousepad
Файл Правка Поиск Просмотр Документ Помощь
lab6-2.asm x 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. 4.16: Рис. 16 Редактирование файла

Создаю исполняемый файл и запускаю его (рис. 17)

```

[alinagomazkova@fedora lab06]$ nasm -f elf lab6-3.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-3 lab6-3.o
[alinagomazkova@fedora lab06]$ ./lab6-3
Результат: 4
Остаток от деления: 1

```

Рис. 4.17: Рис. 17 Запуск исполняемого файла

Изменяю программу так, чтобы она вычисляла значение выражения  $f(x) = (4 * 6 + 2)/5$  (рис. 18)

```

~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/lab6-3.asm - Mousepad
Файл  Правка  Поиск  Просмотр  Документ  Помощь
#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,4 ; EAX=4
mov ebx,6 ; EBX=6
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+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 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. 4.18: Рис. 18 Изменение программы

Создаю и запускаю новый исполняемый файл. Я посчитала для проверки правильности работы программы значение выражения самостоятельно, программа отработала верно (рис. 19)

```

[alinagomazkova@fedora lab06]$ nasm -f elf lab6-3.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-3 lab6-3.o
[alinagomazkova@fedora lab06]$ ./lab6-3
Результат: 5
Остаток от деления: 1
[alinagomazkova@fedora lab06]$

```

Рис. 4.19: Рис. 19 Запуск исполняемого файла

Создаю файл variant.asm с помощью утилиты touch (рис. 20)

```
[alinagomazkova@fedora lab06]$ nasm -f elf lab6-3.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-3 lab6-3.o
[alinagomazkova@fedora lab06]$ ./lab6-3
Результат: 5
Остаток от деления: 1
[alinagomazkova@fedora lab06]$
```

Рис. 4.20: Рис. 20 Создание файла

Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета (рис. 21)

```
*~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/variant.asm - Mousepad
Файл  Правка  Поиск  Просмотр  Документ  Помощь
%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 edx
mov eax, rem
call sprintf
mov eax, edx
call iprintf
call quit
```

Рис. 4.21: Рис. 21 Редактирование файла

Создаю и запускаю исполняемый файл. Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 9 (рис. 22)

```
[alinagomazkova@fedora lab06]$ nasm -f elf variant.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o variant variant.o
[alinagomazkova@fedora lab06]$ ./variant
Введите № студенческого билета:
1032235008
Ваш вариант: 9
[alinagomazkova@fedora lab06]$
```

Рис. 4.22: Рис. 22 Запуск исполняемого файла

## 4.3 Ответы на вопросы по программе

1. За вывод на экран сообщения 'Ваш вариант:' отвечают строки листинга 6.4 кода:

```
mov eax,rem
```

```
call sprint
```

2. Инструкция `mov ecx, x` используется, чтобы положить адрес вводимой строки `x` в регистр `ecx`. `mov edx, 80` - запись в регистр `edx` длины вводимой строки. `call sread` - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры.

3. `call atoi` используется для вызова подпрограммы из внешнего файла, которая преобразует `ascii`-код символа в целое число и записывает результат в регистр `eax`.

4. За вычисления варианта листинга 6.4 отвечают строки:

```
xor edx,edx
```

```
mov ebx,20
```

```
div ebx
```

```
inc edx
```

5. При выполнении инструкции `div ebx` остаток от деления записывается в регистр `edx`.

6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1.

7. За вывод на экран результата вычислений отвечают строки листинга 6.4:

```
mov eax,edx
```

```
call iprintLF
```



## 4.4 Выполнение заданий для самостоятельной работы

Создаю файл lab6-4.asm с помощью утилиты touch (рис. 23)

```
Баш вариант: 9  
[alinagomazkova@fedora lab06]$ touch lab6-4.asm  
[alinagomazkova@fedora lab06]$
```

Рис. 4.23: Рис. 23 Создание файла

Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения  $10 + (31x - 5)$ . Это выражение было под вариантом 9 (рис. 24)

```
*~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab06/lab6-4.asm - Mousepad  
Файл Правка Поиск Просмотр Документ Помощь  
lab6-4.asm lab6-3.asm  
%include 'in_out.asm' ; подключение внешнего файла 10+(31x-5)  
SECTION .data  
msg: DB 'Введите значение переменной x: ',0  
res: 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 sprint  
mov ecx, x  
mov edx, 80  
call sread  
mov eax, x ; вызов подпрограммы преобразования  
call atoi ; ASCII кода в число, 'eax=x'  
mov ebx, 31  
mul ebx; EAX=EAX*EBX = x*31  
add eax, -5; eax = eax-5 = 31x - 5  
add eax, 10; eax = eax+10 = (31x-5)+10  
mov edi, eax ; запись результата вычисления в 'edi'  
; ---- Вывод результата на экран  
mov eax, res ; вызов подпрограммы печати  
call sprint ; сообщения 'Результат: '  
mov eax, edi ; вызов подпрограммы печати значения  
call iprintLF ; из 'edi' в виде символов  
call quit ; вызов подпрограммы завершения
```

Рис. 4.24: Рис. 24 Написание программы

Создаю и запускаю исполняемый файл. При вводе значения 3, вывод - 98 (рис. 25)

```

[alinagomazkova@fedora lab06]$ mousepad lab6-3.asm
[alinagomazkova@fedora lab06]$ nasm -f elf lab6-4.asm
[alinagomazkova@fedora lab06]$ ld -m elf_i386 -o lab6-4 lab6-4.o
[alinagomazkova@fedora lab06]$ ./lab6-4
Введите значение переменной x: 3
Результат: 98
[alinagomazkova@fedora lab06]$ ./lab6-4

```

Рис. 4.25: Рис. 25 Запуск исполняемого файла

Провожу еще один запуск исполняемого файла для проверки работы программы с другим значением на входе. Программа отработала верно (рис. 26)

```

Результат: 98
[alinagomazkova@fedora lab06]$ ./lab6-4
Введите значение переменной x: 1
Результат: 36
[alinagomazkova@fedora lab06]$

```

Рис. 4.26: Рис. 26 Запуск исполняемого файла

## **5 Выводы**

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

# Список литературы

## 1. Архитектура ЭВМ