

# **Отчёт по лабораторной работе №6**

**Дисциплина: архитектура компьютера**

Кайнова Алина Андреевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>4</b>
<b>2</b>	<b>Задание</b>	<b>5</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>6</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>8</b>
4.1	Выполнение арифметических операций в NASM . . . . .	13
4.1.1	Ответы на вопросы по программе . . . . .	17
4.2	Выполнение заданий для самостоятельной работы . . . . .	17
<b>5</b>	<b>Выводы</b>	<b>20</b>
	<b>Список литературы</b>	<b>21</b>

## Список иллюстраций

4.1	Создание директории . . . . .	8
4.2	Создание файла . . . . .	8
4.3	Копирование файла . . . . .	8
4.4	Редактирование файла . . . . .	9
4.5	Запуск исполняемого файла . . . . .	9
4.6	Редактирование файла . . . . .	10
4.7	Запуск исполняемого файла . . . . .	10
4.8	Создание файла . . . . .	10
4.9	Запуск исполняемого файла . . . . .	11
4.10	Редактирование файла . . . . .	12
4.11	Запуск исполняемого файла . . . . .	12
4.12	Редактирование файла . . . . .	13
4.13	Запуск исполняемого файла . . . . .	13
4.14	Создание файла . . . . .	13
4.15	Редактирование файла . . . . .	14
4.16	Запуск исполняемого файла . . . . .	14
4.17	Изменение программы . . . . .	15
4.18	Запуск исполняемого файла . . . . .	15
4.19	Создание файла . . . . .	15
4.20	Редактирование файла . . . . .	16
4.21	Запуск исполняемого файла . . . . .	16
4.22	Создание файла . . . . .	17
4.23	Редактирование файла . . . . .	18
4.24	Запуск исполняемого файла . . . . .	18
4.25	Запуск исполняемого файла . . . . .	19

# 1 Цель работы

Освоить арифметические инструкции 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

##Символьные и численные данные в NASM

Создаю каталог для лабораторной работы №7 и перехожу в него

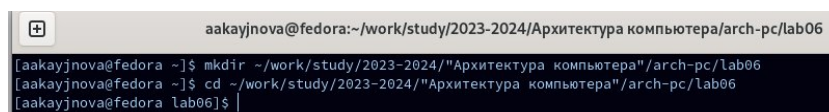
A terminal window with a title bar showing the user 'aakayjnova' and the current directory. The terminal displays three lines of commands and their output: 'mkdir ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06', 'cd ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab06', and the prompt 'lab06\$'.

Рис. 4.1: Создание директории

Создаю файл lab6-1.asm

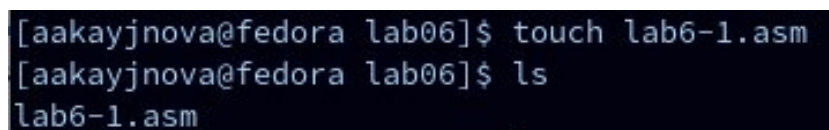
A terminal window showing two lines of commands: 'touch lab6-1.asm' and 'ls'. The output of 'ls' is 'lab6-1.asm'.

Рис. 4.2: Создание файла

Копирую файл in\_out.asm в текущий каталог

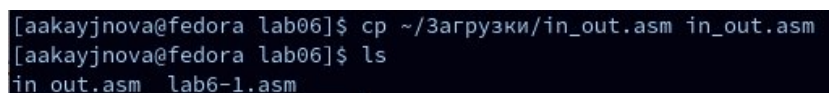
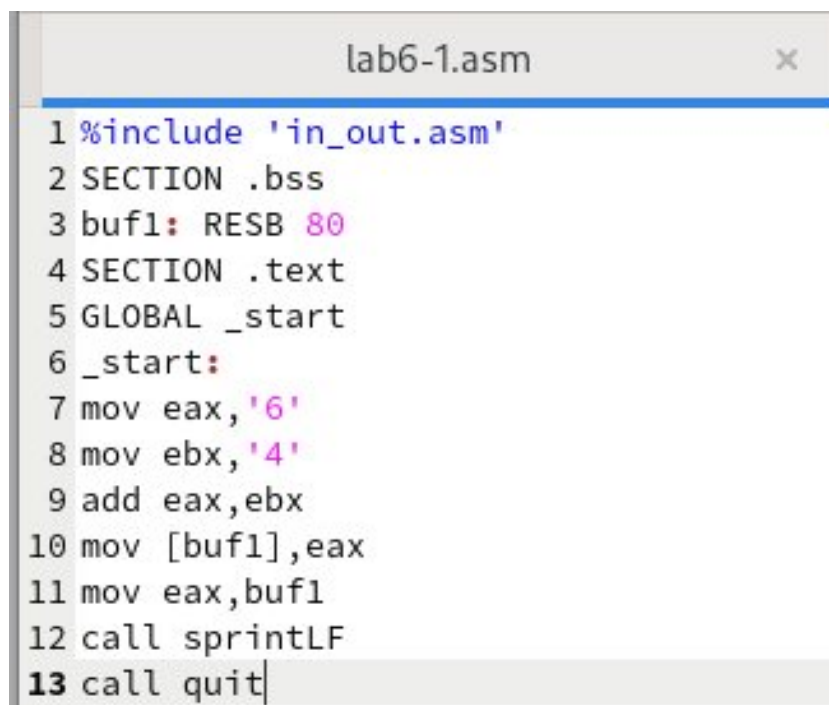
A terminal window showing two lines of commands: 'cp ~/Загрузки/in\_out.asm in\_out.asm' and 'ls'. The output of 'ls' is 'in\_out.asm lab6-1.asm'.

Рис. 4.3: Копирование файла

Открываю созданный файл и копирую в него программу вывода значения регистра eax

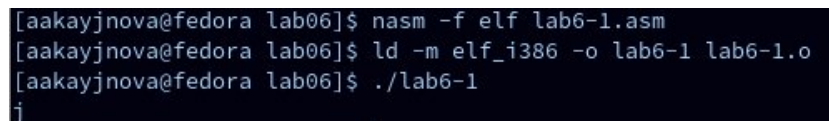




```
lab6-1.asm
1 %include 'in_out.asm'
2 SECTION .bss
3 buf1: RESB 80
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov eax, '6'
8 mov ebx, '4'
9 add eax, ebx
10 mov [buf1], eax
11 mov eax, buf1
12 call sprintLF
13 call quit
```

Рис. 4.4: Редактирование файла

Создаю исполняемый файл и запускаю его



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

Рис. 4.5: Запуск исполняемого файла

Изменяю в тексте программы в файле lab6-1.asm символы “6” и “4” на цифры 6 и 4 соответственно

```
*lab6-1.asm
1 %include 'in_out.asm'
2 SECTION .bss
3 buf1: RESB 80
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov eax,6
8 mov ebx,4
9 add eax,ebx
10 mov [buf1],eax
11 mov eax,buf1
12 call sprintLF
13 call quit
```

Рис. 4.6: Редактирование файла

Создаю новый исполняемый файл программы и запускаю его

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

Рис. 4.7: Запуск исполняемого файла

Создаю новый файл lab6-2.asm

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

Рис. 4.8: Создание файла

Вставляю в файл текст другой программы для вывода значения eax

## lab6-2.asm

```
1 %include 'in_out.asm'
2 SECTION .text
3 GLOBAL _start
4 _start:
5 mov eax,'6'
6 mov ebx,'4'
7 add eax,ebx
8 call iprintLF
9 call quit
```

<https://esystem.rudn.ru/pluginfile.php>

Создаю и запускаю исполняемый файл lab6-2

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

Рис. 4.9: Запуск исполняемого файла

Изменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на цифры 6 и 4 соответственно

```
lab6-2.asm
1 %include 'in_out.asm'
2 SECTION .text
3 GLOBAL _start
4 _start:
5 mov eax,6
6 mov ebx,4
7 add eax,ebx
8 call iprintLF
9 call quit
```

Рис. 4.10: Редактирование файла

Создаю и запускаю исполняемый файл

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

Рис. 4.11: Запуск исполняемого файла

Заменяю функцию `iprintLF` на `iprint` в тексте программы. Вывод не изменился, так как символ переноса строки не отображался, когда программа исполнялась с функцией `iprintLF`, а `iprint` не добавляет к выводу символ переноса строки, в отличие от `iprintLF`

```
*lab6-2.asm
1 %include 'in_out.asm'
2 SECTION .text
3 GLOBAL _start
4 _start:
5 mov eax,6
6 mov ebx,4
7 add eax,ebx
8 call iprint
9 call quit
```

Рис. 4.12: Редактирование файла

Создаю и запускаю новый исполняемый файл

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

Рис. 4.13: Запуск исполняемого файла

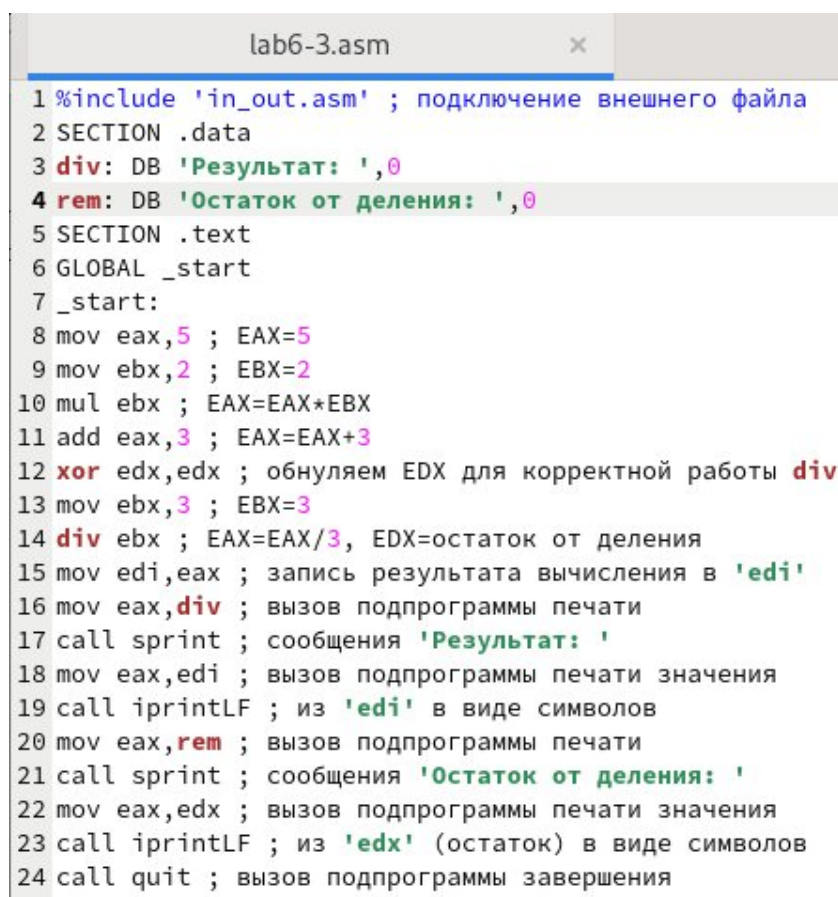
## 4.1 Выполнение арифметических операций в NASM

Создаю файл lab6-3.asm

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

Рис. 4.14: Создание файла

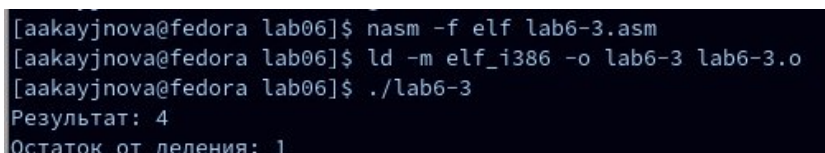
Ввожу в созданный файл текст программы для вычисления значения выражения  $f(x)=(5*2+3)/3$



```
lab6-3.asm
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8 mov eax,5 ; EAX=5
9 mov ebx,2 ; EBX=2
10 mul ebx ; EAX=EAX*EBX
11 add eax,3 ; EAX=EAX+3
12 xor edx,edx ; обнуляем EDX для корректной работы div
13 mov ebx,3 ; EBX=3
14 div ebx ; EAX=EAX/3, EDX=остаток от деления
15 mov edi,eax ; запись результата вычисления в 'edi'
16 mov eax,div ; вызов подпрограммы печати
17 call sprint ; сообщения 'Результат: '
18 mov eax,edi ; вызов подпрограммы печати значения
19 call iprintLF ; из 'edi' в виде символов
20 mov eax,rem ; вызов подпрограммы печати
21 call sprint ; сообщения 'Остаток от деления: '
22 mov eax,edx ; вызов подпрограммы печати значения
23 call iprintLF ; из 'edx' (остаток) в виде символов
24 call quit ; вызов подпрограммы завершения
```

Рис. 4.15: Редактирование файла

Создаю исполняемый файл и запускаю его



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

Рис. 4.16: Запуск исполняемого файла

Изменяю текст программы, чтобы вычислить значение выражения  $f(x)=(4*6+2)/5$

```
lab6-3.asm
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8 mov eax,4 ; EAX=4
9 mov ebx,6 ; EBX=6
10 mul ebx ; EAX=EAX*EBX
11 add eax,2 ; EAX=EAX+2
12 xor edx,edx ; обнуляем EDX для корректной работы div
13 mov ebx,5 ; EBX=5
14 div ebx ; EAX=EAX/5, EDX=остаток от деления
15 mov edi,eax ; запись результата вычисления в 'edi'
16 mov eax,div ; вызов подпрограммы печати
17 call sprint ; сообщения 'Результат: '
18 mov eax,edi ; вызов подпрограммы печати значения
19 call iprintLF ; из 'edi' в виде символов
20 mov eax,rem ; вызов подпрограммы печати
21 call sprint ; сообщения 'Остаток от деления: '
22 mov eax,edx ; вызов подпрограммы печати значения
23 call iprintLF ; из 'edx' (остаток) в виде символов
24 call quit ; вызов подпрограммы завершения
```

Рис. 4.17: Изменение программы

Создаю и запускаю новый исполняемый файл

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

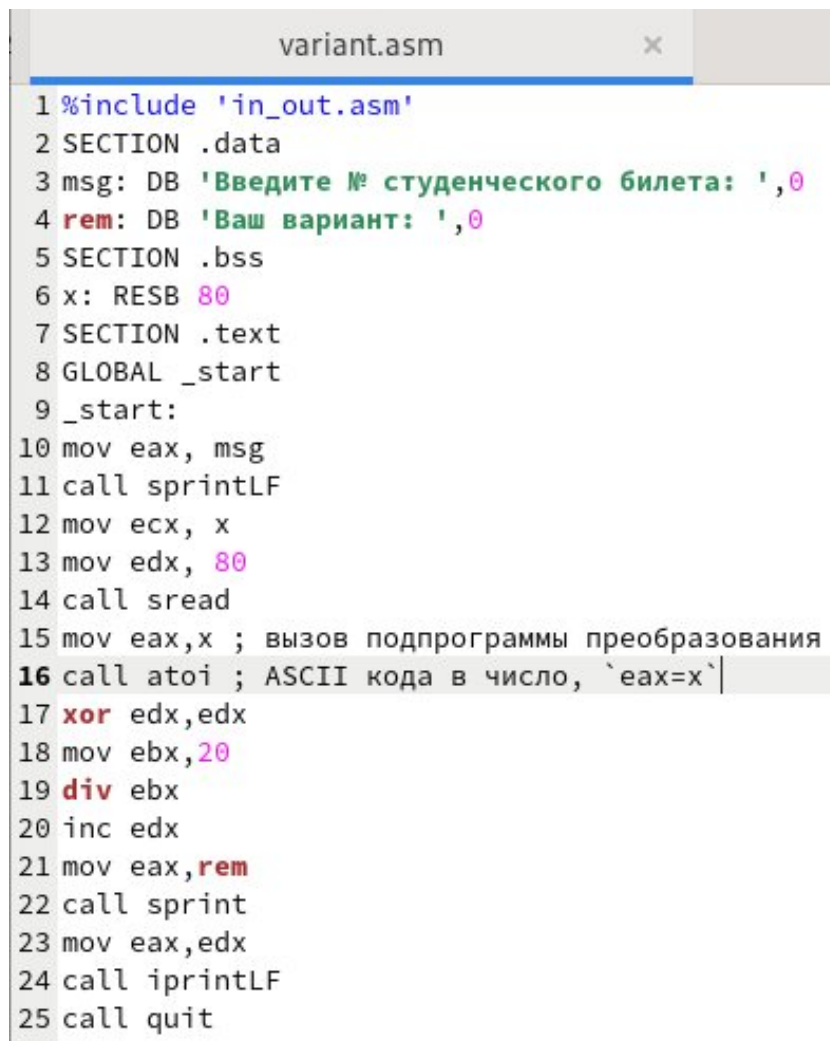
Рис. 4.18: Запуск исполняемого файла

Создаю файл variant.asm

```
[aakayjnova@fedora lab06]$ touch variant.asm
```

Рис. 4.19: Создание файла

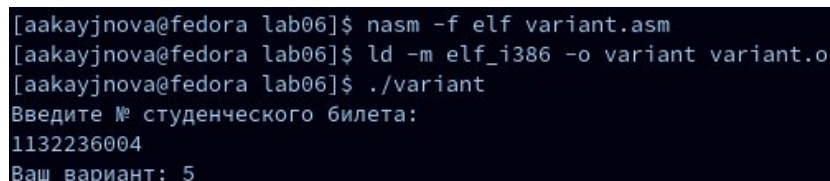
Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета



```
1 %include 'in_out.asm'
2 SECTION .data
3 msg: DB 'Введите № студенческого билета: ',0
4 rem: DB 'Ваш вариант: ',0
5 SECTION .bss
6 x: RESB 80
7 SECTION .text
8 GLOBAL _start
9 _start:
10 mov eax, msg
11 call sprintf
12 mov ecx, x
13 mov edx, 80
14 call sread
15 mov eax, x ; вызов подпрограммы преобразования
16 call atoi ; ASCII кода в число, `eax=x`
17 xor edx, edx
18 mov ebx, 20
19 div ebx
20 inc edx
21 mov eax, rem
22 call sprintf
23 mov eax, edx
24 call iprintLF
25 call quit
```

Рис. 4.20: Редактирование файла

Создаю и запускаю исполняемый файл, ввожу номер своего студ. билета



```
[aakayjnova@fedora lab06]$ nasm -f elf variant.asm
[aakayjnova@fedora lab06]$ ld -m elf_i386 -o variant variant.o
[aakayjnova@fedora lab06]$ ./variant
Введите № студенческого билета:
1132236004
Ваш вариант: 5
```

Рис. 4.21: Запуск исполняемого файла



### 4.1.1 Ответы на вопросы по программе

1. За вывод на экран сообщения “Ваш вариант” отвечают строки кода:

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

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

3. Инструкция `call atoi` используется для вызова из внешнего файла подпрограммы, преобразующей ASCII-код символа в целое число и записывающей результат в регистр `eax`.

4. За вычисление варианта отвечают строки листинга:

```
xor edx,edx ; обнуление edx для корректной работы div mov ebx,20 ; ebx = 20 div ebx ; eax = eax/20, edx - остаток от деления inc edx ; edx = edx + 1
```

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

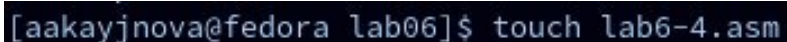
6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1.

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

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

## 4.2 Выполнение заданий для самостоятельной работы

Создаю файл `lab6-4.asm`



```
[aakay]nova@fedora lab06]$ touch lab6-4.asm
```

Рис. 4.22: Создание файла

Записываю в созданный файл текст программы для вычисления значения выражения под вариантом 5  $(9 \cdot x - 8) / 8$

```
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data ; секция инициированных данных
3 msg: DB 'Введите значение переменной x: ',0
4 rem: DB 'Результат: ',0
5 SECTION .bss ; секция не инициированных данных
6 x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры
7 SECTION .text ; Код программы
8 GLOBAL _start ; Начало программы
9 _start: ; Точка входа в программу
10 mov eax, msg ; запись адреса выводимого сообщения в eax
11 call sprint ; вызов подпрограммы печати сообщения
12 mov ecx, x ; запись адреса переменной в ecx
13 mov edx, 80 ; запись длины вводимого значения в edx
14 call sread ; вызов подпрограммы ввода сообщения
15 mov eax,x ; вызов подпрограммы преобразования
16 call atoi ; ASCII кода в число, `eax=x`
17 mov ebx,9; EBX = 9
18 mul ebx ; EAX=EAX*EBX = 9*x
19 add eax,-8 ; eax = eax-8 = (9*x-8)
20 mov ebx,8 ;
21 div ebx; EAX=EAX/EBX = (9*x-8)/8
22 mov edi,eax ; запись результата вычисления в 'edi'
23 mov eax,rem ; вызов подпрограммы печати
24 call sprint ; сообщения 'Результат: '
25 mov eax,edi ; вызов подпрограммы печати значения
26 call iprintLF ; из 'edi' в виде символов
27 call quit ; вызов подпрограммы завершения
```

Рис. 4.23: Редактирование файла

Создаю и запускаю исполняемый файл, ввожу значение 8

```
[aakayjnova@fedora lab06]$ nasm -f elf lab6-4.asm
[aakayjnova@fedora lab06]$ ld -m elf_i386 -o lab6-4 lab6-4.o
[aakayjnova@fedora lab06]$ ./lab6-4
Введите значение переменной x: 8
Результат: 8
```

Рис. 4.24: Запуск исполняемого файла

Ввожу другое значение x

```
[aakayjnova@fedora lab06]$ ./lab6-4  
Введите значение переменной x: 64  
Результат: 71
```

Рис. 4.25: Запуск исполняемого файла

## 5 Выводы

В ходе данной лабораторной работы мы научились работать с арифметическими инструкциями языка ассемблера NASM.

## Список литературы

- [illegible]