

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

дисциплина: Архитектура компьютера

Студент: Безходарнова А.В

Группа: НКАбд-01-25

МОСКВА

2025 г.

СОДЕРЖАНИЕ

1.Цель работы	3
2. Выполнение лабораторной работы.....	4
2.1 Символьные и численные данные в NASM.....	4
2.2 Выполнение арифметических операций в NASM	8
2.3 Ответы на контрольные вопросы.....	11
3. Задание для самостоятельной работы	12
4. Выводы	14

1.Цель работы

Целью данной лабораторной работы является освоение арифметических инструкций языка ассемблера NASM.

2. Выполнение лабораторной работы

2.1 Символьные и численные данные в NASM

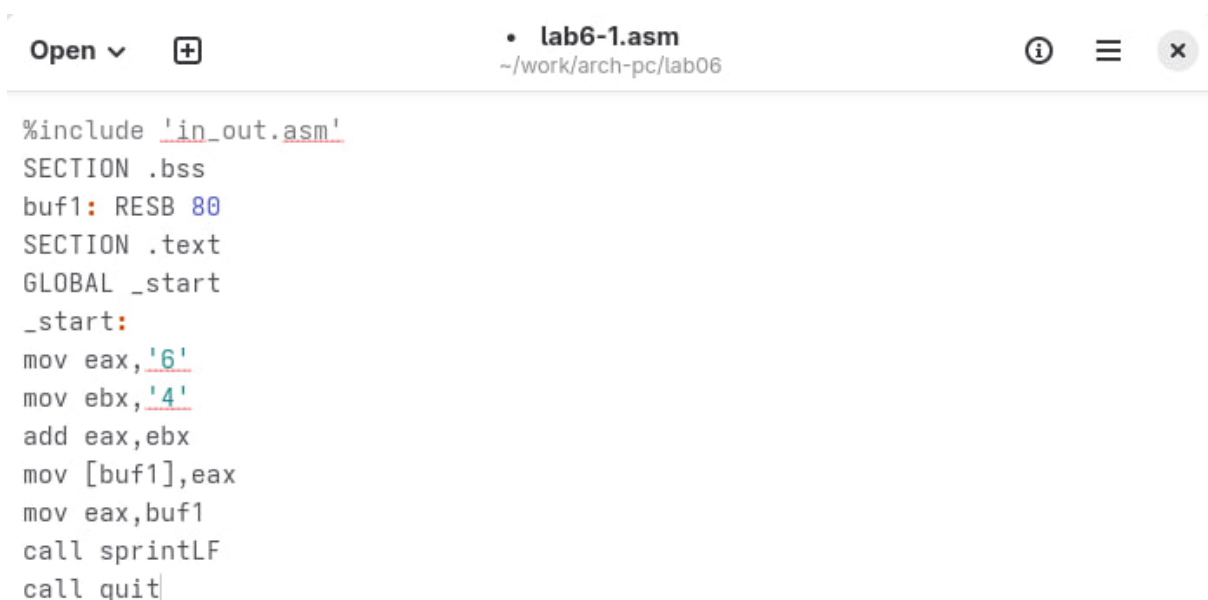
Создаю каталог для программ лабораторной работы № 6, перейдя в него создаю файл lab6-1.asm:.(рис.2.1.1)



```
avbezhodarnova1@avbezhodarnova1:~$ mkdir ~/work/arch-pc/lab06
avbezhodarnova1@avbezhodarnova1:~$ cd ~/work/arch-pc/lab06
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ touch lab6-1.asm
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$
```

Рис. 2.1.1. Создание файла

В созданном файле ввожу программу из листинга. (рис.2.1.3)



```
Open v [icon] • lab6-1.asm ~/work/arch-pc/lab06
%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. 2.1.2 Загрузка текста из листинга

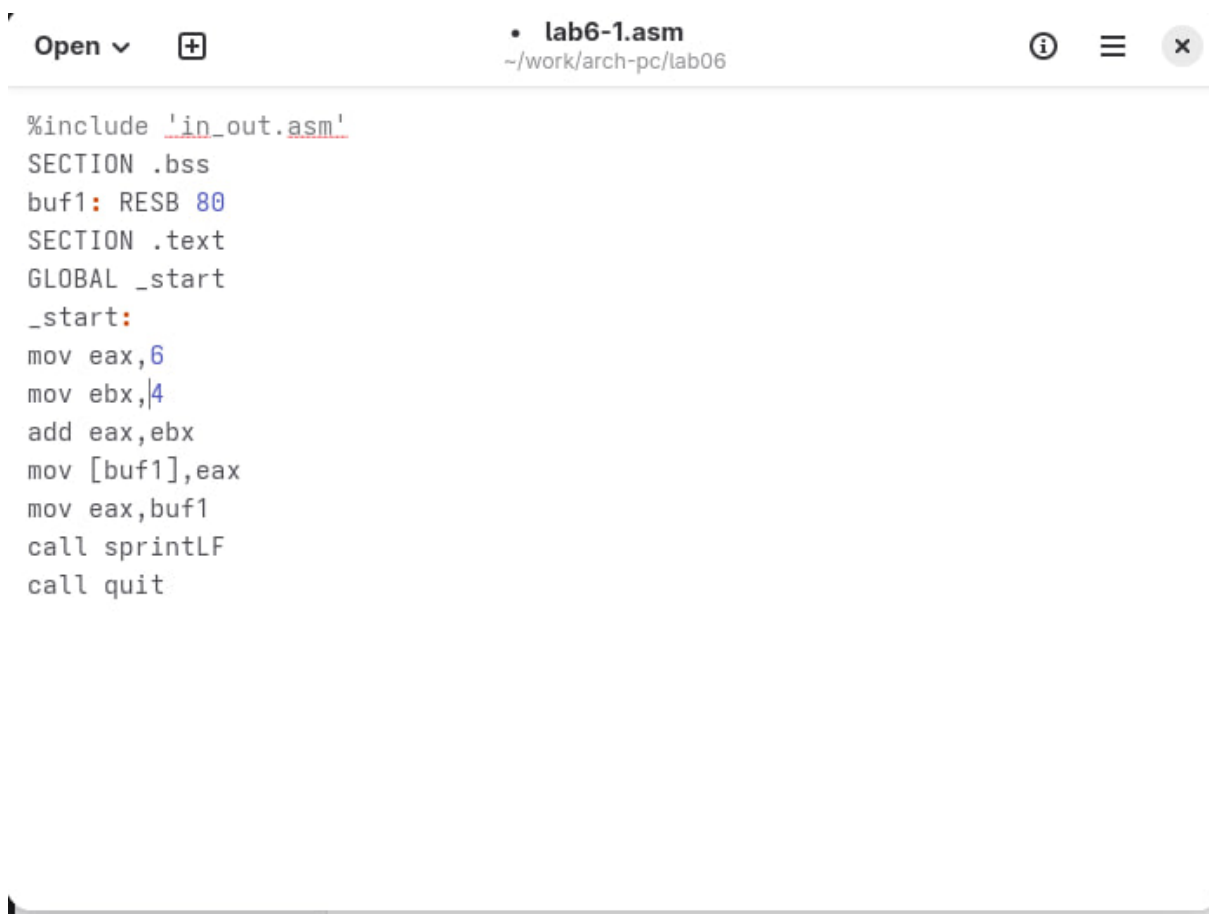
Далее запускаю исходный код. (рис.2.1.4)



```
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ./lab6-1
j
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$
```

Рис. 2.1.3 Запуск программы

Далее я убираю кавычки и снова запускаю файл. (Рис.2.1.4) и (рис.2.1.5)



The screenshot shows a text editor window titled 'lab6-1.asm' with the path '~/.work/arch-pc/lab06'. The code inside is as follows:

```
%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.2.1.4 Изменение текста



The screenshot shows a terminal window with the following commands and output:

```
avbezkhodarnova1@avbezkhodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
avbezkhodarnova1@avbezkhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
avbezkhodarnova1@avbezkhodarnova1:~/work/arch-pc/lab06$ ./lab6-1

```

Рис.2.1.5 Запуск программы

Программа выдала пустую строку, это связано с тем, что символ 10 означает переход на новую строку.

Далее создаю новый файл, загружаю туда код из листинга и запускаю программу. (рис.2.1.6)

```

avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ./lab6-2
106
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$

```

Рис.2.1.6 запуск программы

В результате запуска программы получилось число 106, программа, как и в первый раз, сложила коды символов, но вывела само число, а не его символ, благодаря замене функции вывода на `iprintLF`.

Я убираю кавычки (рис.2.1.7) и получаю нужный результат (рис.2.1.8)



```

Open ▾ +
• lab6-2.asm
~/work/arch-pc/lab06
lab6-1.asm
● lab6-2.asm
x

%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit

```

Рис.2.1.7 изменение текста программы

```

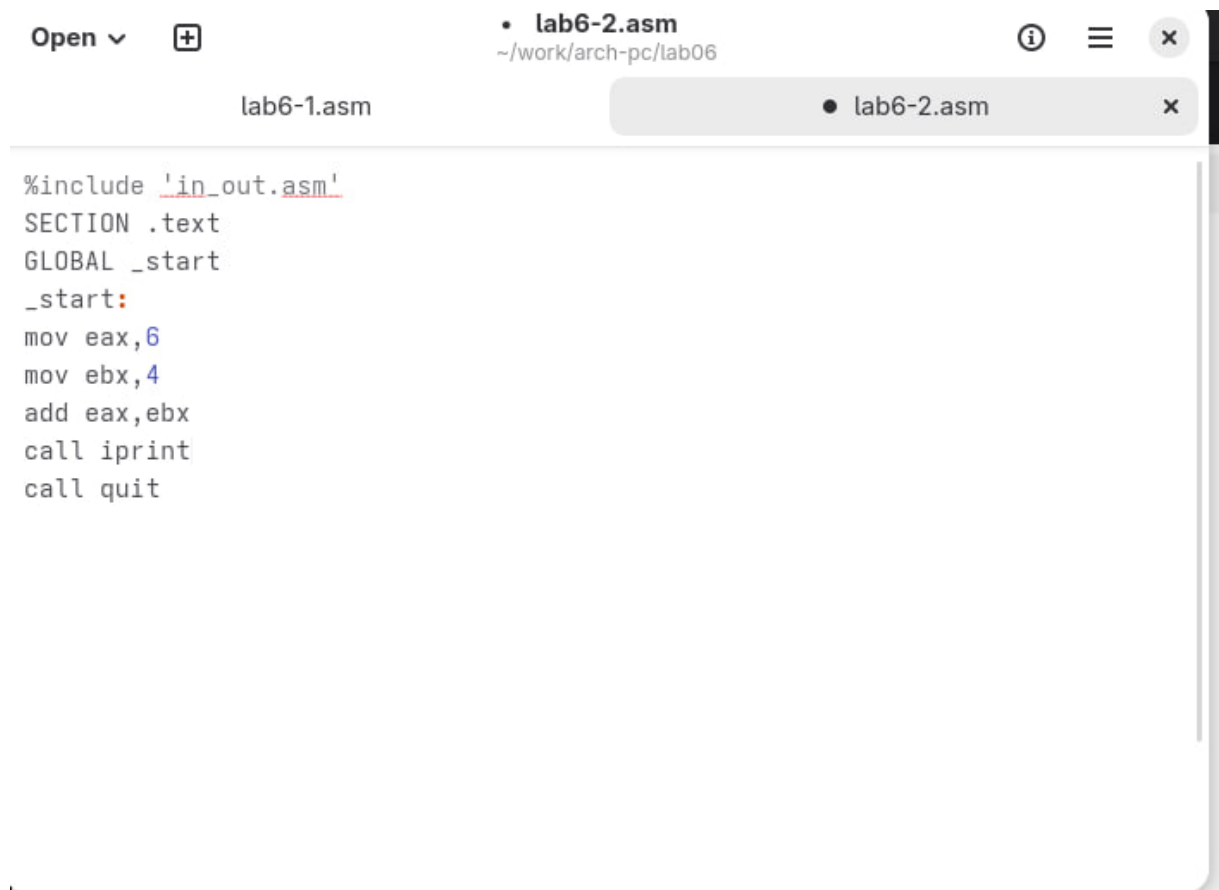
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ./lab6-2
10
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$

```

Рис.2.1.8 Вывод результата

Заменяю функцию `iprintLF` на `iprint` (рис.2.1.9) и запускаю программу.

(рис.2.1.10)



The screenshot shows a code editor with two tabs: 'lab6-1.asm' and 'lab6-2.asm'. The 'lab6-2.asm' tab is active and displays the following assembly code:

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис.2.1.9 Замена функции



```
avbezkhodarnova1@avbezkhodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
avbezkhodarnova1@avbezkhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
avbezkhodarnova1@avbezkhodarnova1:~/work/arch-pc/lab06$ ./lab6-2
10avbezkhodarnova1@avbezkhodarnova1:~/work/arch-pc/lab06$
```

Рис.2.1.10 запуск программы

Отличие состоит в том, что во втором случае не произошел переход на новую строку.

2.2 Выполнение арифметических операций в NASM

Создаю новый файл и загружаю туда текст из листинга. (рис. 2.2.1)



```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
66 Демидова А. В.
Архитектура ЭВМ
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 iprintfLF ; 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 iprintfLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершенияS
```

Рис.2.2.1 Ввод кода.

Запускаю программу. Она выдает результат вычислений, а также остаток от деления. (Рис.2.2.2)

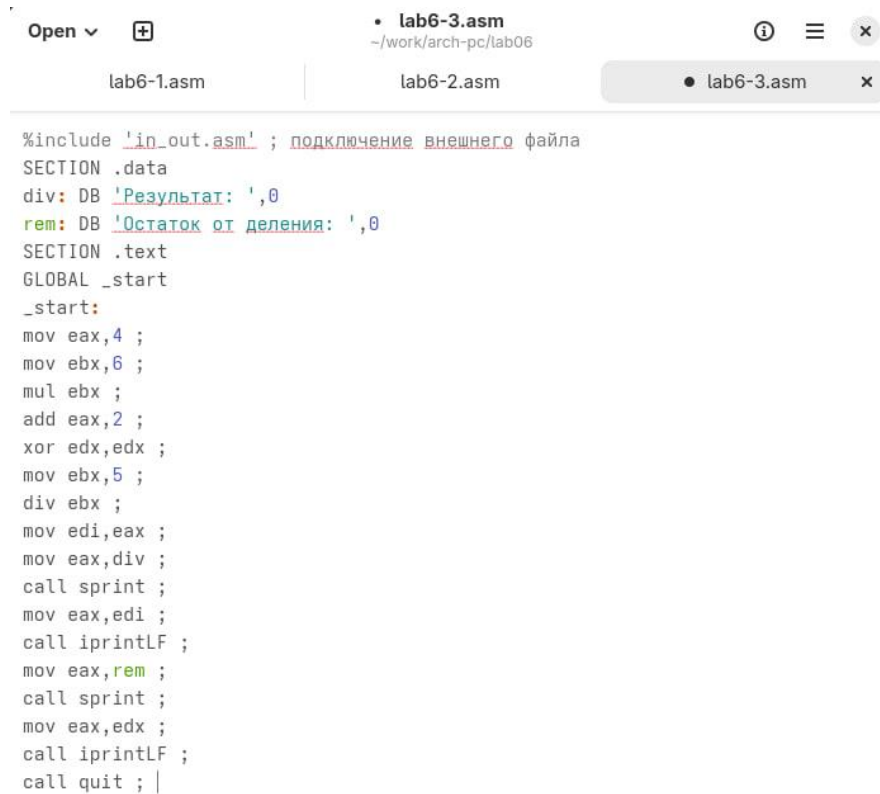

```

avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$

```

Рис. 2.2.2 Запуск

Изменяю данные на $f(x) = (4*6+2)/5$ для вычисления. (Рис.2.2.3)



```

Open ▾ ⊞ lab6-3.asm ~/work/arch-pc/lab06
lab6-1.asm lab6-2.asm lab6-3.asm x

%include 'in_out.asm' ; подключение внешнего файла
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 iprintLF ;
mov eax,rem ;
call sprint ;
mov eax,edx ;
call iprintLF ;
call quit ; |

```

Рис. 2.2.3 Изменения кода

Запускаю программу и получаю ответ (рис.2.2.4)

```

avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
avbezhkodarnova1@avbezhkodarnova1:~/work/arch-pc/lab06$

```

Рис.2.2.4 Запуск программы

Далее я создаю новый файл, в который загружаю текст из листинга.

(рис.2.2.5)

```
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$
```

Рис.2.2.5 Создание файла

Запускаю, ввожу номер своего студенческого билета и получаю ответ.

(Рис.2.2.6)

```
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ nasm -f elf variant.asm
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant.asm variant.asm.o
ld: cannot find variant.asm.o: No such file or directory
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032253545
Ваш вариант: 6
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$
```

Рис.2.2.6 Запуск программы

2.3 Ответы на контрольные вопросы

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

```
mov eax  
rem call sprint
```

2. Эти инструкции читают строку ввода от пользователя. Команда `mov esx, x` указывает, куда сохранить введенные данные, а `mov edx, 80` задает максимальную длину ввода 80 символов.

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 отвечают строки:

```
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`.

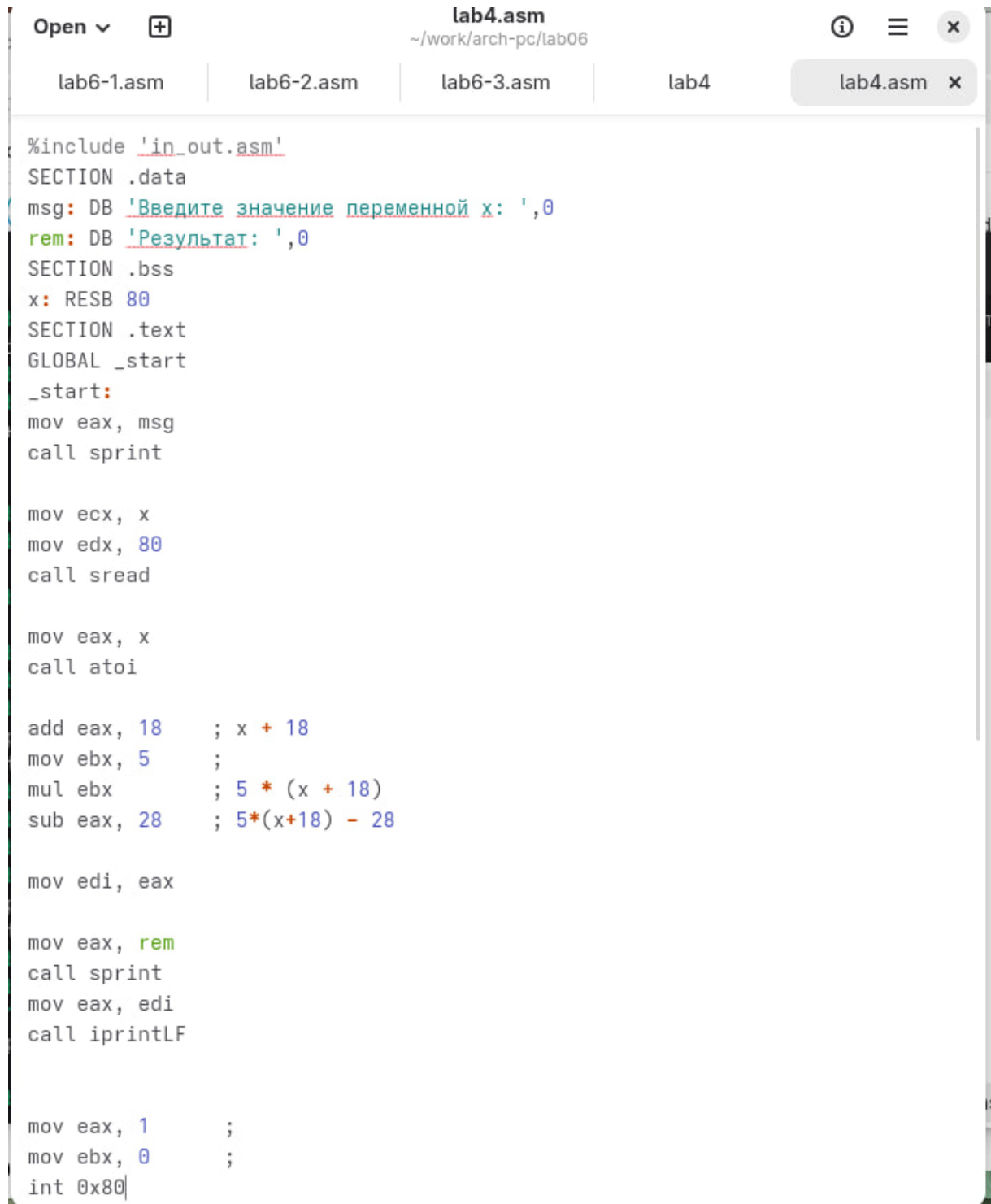
6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1.

7. За вывод на экран результатов вычислений отвечают строки:

```
mov eax edx  
call iprintLF
```

3. Задание для самостоятельной работы

Из предложенных вариантов, я выбираю свое выражение: $f(x) = 5 \cdot (x + 18) - 28$ и изменяю код так, чтобы программа считала именно мое выражение. Рис(3.1) и рис(3.2)



```
lab4.asm
~/work/arch-pc/lab06

lab6-1.asm | lab6-2.asm | lab6-3.asm | lab4 | lab4.asm x

#include 'in_out.asm'
SECTION .data
msg: DB 'Введите значение переменной x: ',0
rem: DB 'Результат: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprint

mov ecx, x
mov edx, 80
call sread

mov eax, x
call atoi

add eax, 18 ; x + 18
mov ebx, 5 ;
mul ebx ; 5 * (x + 18)
sub eax, 28 ; 5*(x+18) - 28

mov edi, eax

mov eax, rem
call sprint
mov eax, edi
call iprintLF

mov eax, 1 ;
mov ebx, 0 ;
int 0x80
```

Рис.3.1 Редактирование файла.

```
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ nasm -f elf lab4.asm
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab4 lab4.o
^[[Aavbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$ ./lab4
Введите значение переменной x: 1
Результат: 67
avbezhodarnova1@avbezhodarnova1:~/work/arch-pc/lab06$
```

Рис.3.2 Запуск программы.

4. Выводы

В ходе выполнения данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM