

# **Лабораторная работа №6**

**Арифметические операции в NASM**

Приспешкин Андрей Андреевич

# Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Ответы на вопросы	19
5	Задания для самостоятельной работы	20
6	Выводы	23
	Список литературы	24

## Список иллюстраций

3.1	Рис.1 Создание рабочего каталога и файла lab6-1.asm . . . . .	7
3.2	Рис.2 Код из листинга 6.1 . . . . .	8
3.3	Рис.3 Создание и запуск исполняемого файла . . . . .	8
3.4	Рис.4 Замена символов на числа . . . . .	9
3.5	Рис.5 Результат работы файла после замены символов на числа .	9
3.6	Рис.6 Создание файла lab6-2.asm . . . . .	9
3.7	Рис.7 Код из листинга 6.2 . . . . .	10
3.8	Рис.9 Код после замены символов на числа . . . . .	11
3.9	Рис.10 Результат работы программы после замены символов на числа	11
3.10	Рис.11 Замена функции iprintLF на функцию iprint . . . . .	12
3.11	Рис.12 Результат работы программы после замены iprintLF на iprint	12
3.12	Рис.13 Создание файла lab6-3.asm . . . . .	13
3.13	Рис.14 Код из листинга 6.3 . . . . .	13
3.14	Рис.15 Работа файла lab6-3.asm . . . . .	14
3.15	Рис.16 Код после замены чисел . . . . .	15
3.16	Рис.17 Результат работы программы после замены чисел . . . . .	16
3.17	Рис.18 Создание программы variant.asm . . . . .	16
3.18	Рис.19 Код из листинга 6.4 . . . . .	17
3.19	Рис.20 Результат работы программы . . . . .	18
5.1	Рис.21 Код для вычисления выражения . . . . .	21
5.2	Рис.22 Проверка работы программы . . . . .	22

## Список таблиц

# 1 Цель работы

Цель данной лабораторной работы – Освоение арифметических инструкций в языке ассемблера NASM

## 2 Задание

1. Численные и символьные данные в NASM
2. Выполнение арифметических операций
3. Задания для самостоятельной работы

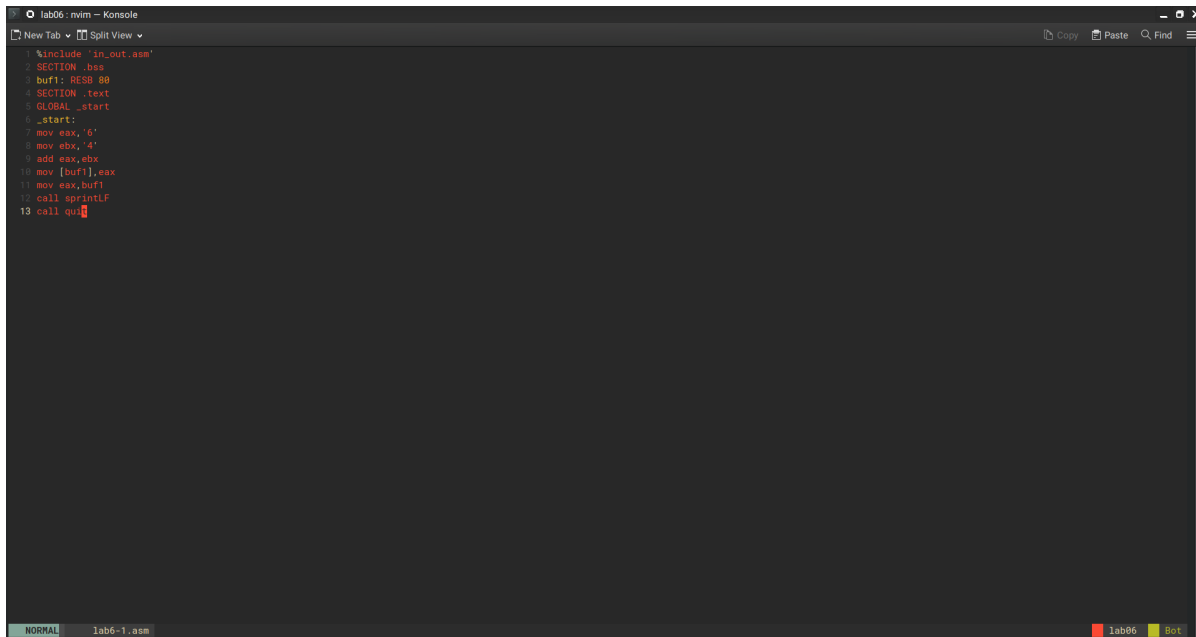
### 3 Выполнение лабораторной работы

Утилитой `mkdir` создадим каталог для выполнения лабораторной работы, утилитой `touch` создаём в этом каталоге файл `lab6-1.asm`, проверим утилитой `ls`(Рис.1).

```
aaprispeshkin:[aaprispeshkin]:~$ mkdir ~/work/arch-pc/lab06
aaprispeshkin:[aaprispeshkin]:~$ cd ~/work/arch-pc/lab06
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ touch lab6-1.asm
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ls
lab6-1.asm
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$
```

Рис. 3.1: Рис.1 Создание рабочего каталога и файла `lab6-1.asm`

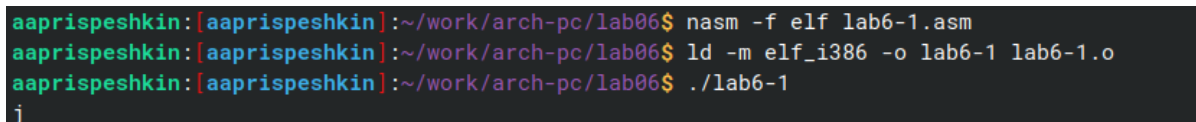
В текстовом редакторе `neovim` вставим в файл код из листинга 6.1(Рис.2).



```
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 sprintf
13 call quit
```

Рис. 3.2: Рис.2 Код из листинга 6.1

Создадим исполняемый файл lab6-1 и запустим его, заметим что вместо желаемого результата мы получаем символ j(Рис.3).

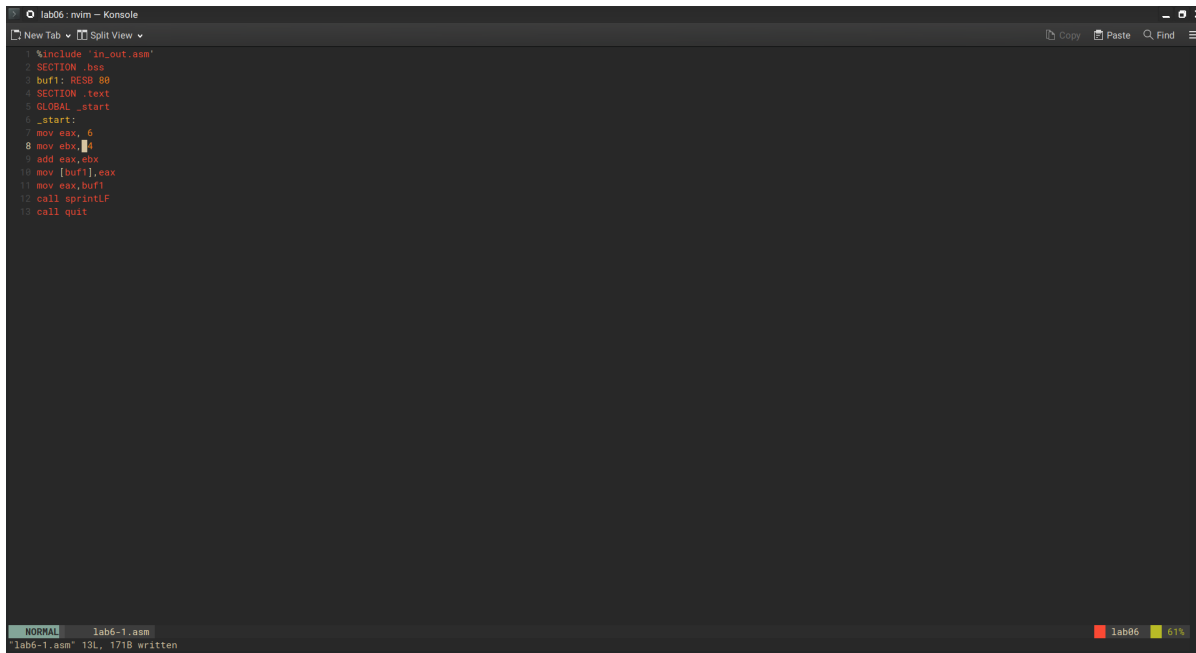


```
aapripeshkin:[aapripeshkin]:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
aapripeshkin:[aapripeshkin]:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
aapripeshkin:[aapripeshkin]:~/work/arch-pc/lab06$ ./lab6-1
j
```

Рис. 3.3: Рис.3 Создание и запуск исполняемого файла

Поменяем в коде '6' и '4' на 6 и 4(Рис.4).

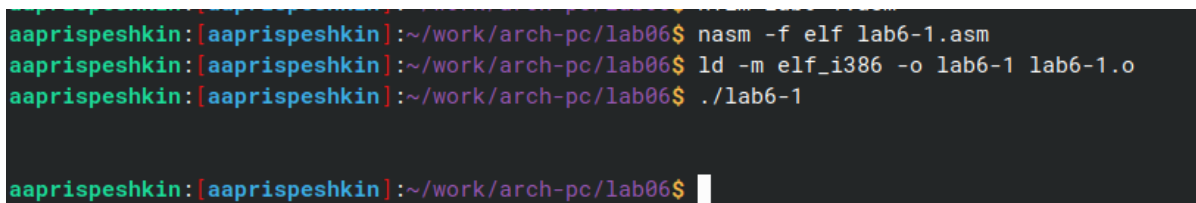




```
1 %include 'in_out.asm'
2 SECTION .bss
3 buf1: RESB 80
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov eax, 6
8 mov ebx, 1
9 add eax, ebx
10 mov [buf1], eax
11 mov eax, buf1
12 call sprintf
13 call quit
```

Рис. 3.4: Рис.4 Замена символов на числа

Создадим исполняемый файл и запустим его, заметим что в этот раз мы получили символ перевода строки(Рис.5).

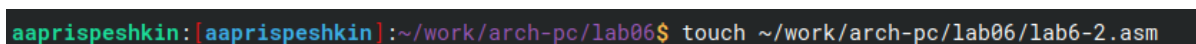


```
aaprispeshkin: [aaprispeshkin]: ~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
aaprispeshkin: [aaprispeshkin]: ~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
aaprispeshkin: [aaprispeshkin]: ~/work/arch-pc/lab06$ ./lab6-1

aaprispeshkin: [aaprispeshkin]: ~/work/arch-pc/lab06$
```

Рис. 3.5: Рис.5 Результат работы файла после замены символов на числа

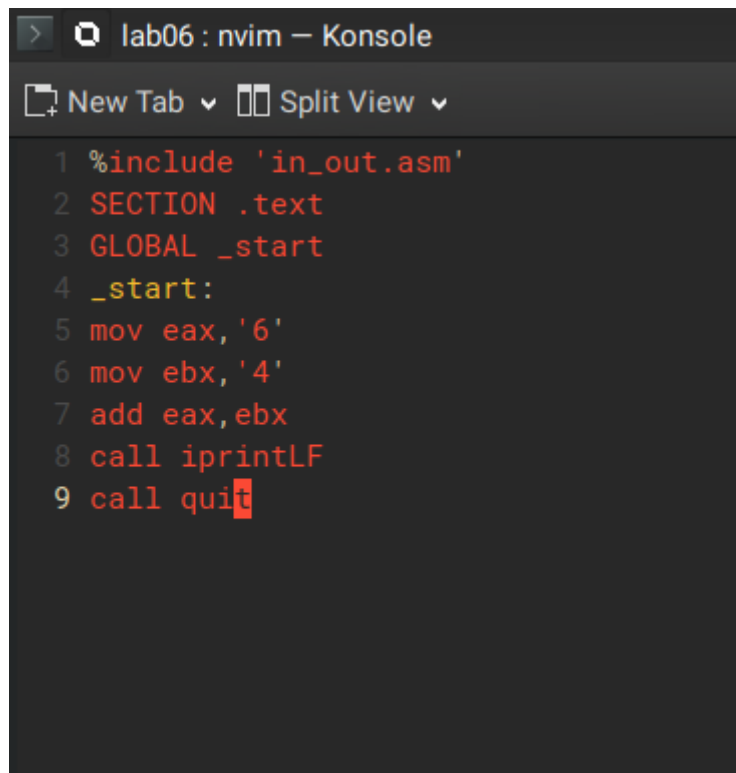
Утилитой touch создадим файл lab6-2.asm(Рис.6).



```
aaprispeshkin: [aaprispeshkin]: ~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-2.asm
```

Рис. 3.6: Рис.6 Создание файла lab6-2.asm

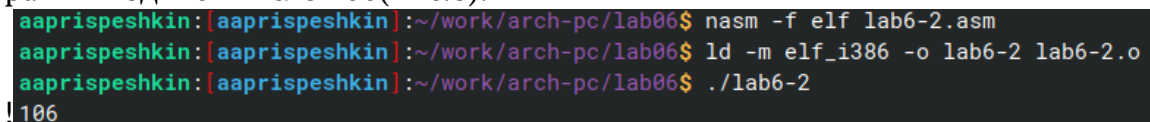
Вставим туда код из листинга 6.2(Рис.7).

A screenshot of a terminal window titled 'lab06 : nvim — Konsole'. The window shows a text editor with assembly code. The code is as follows:

```
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
```

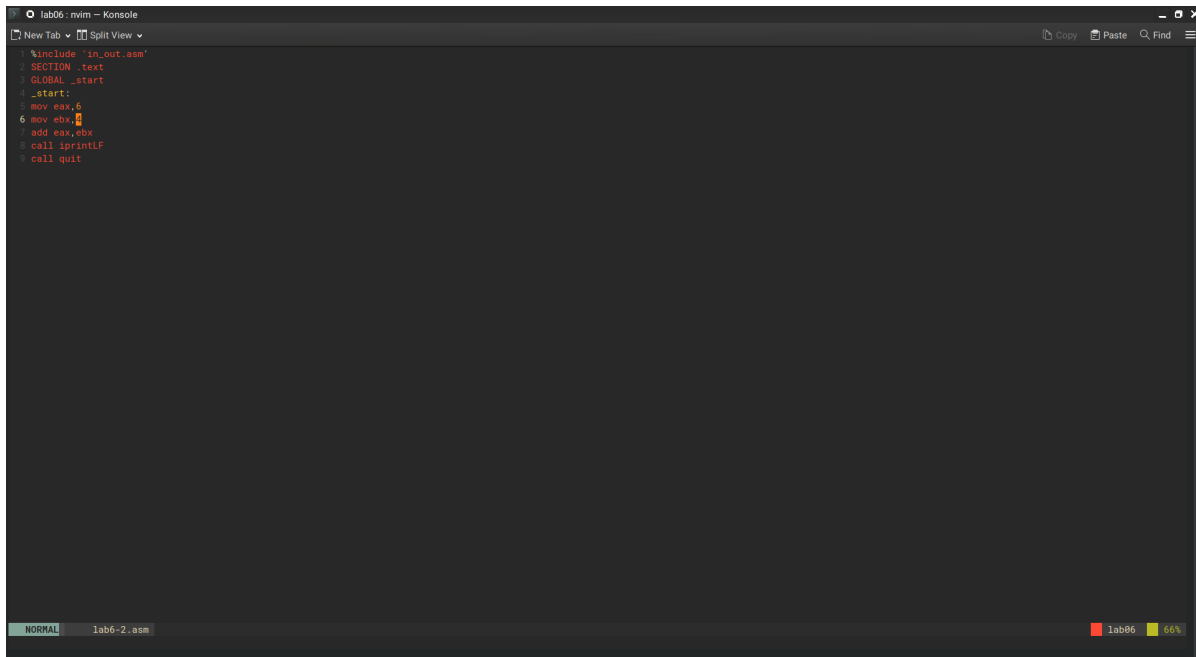
Рис. 3.7: Рис.7 Код из листинга 6.2

Создадим исполняемый файл и проверим результат его работы, увидим что на экран выводится число 106(Рис.8).

A screenshot of a terminal window showing the compilation and execution of the assembly code. The output is as follows:

```
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ./lab6-2
!106
```

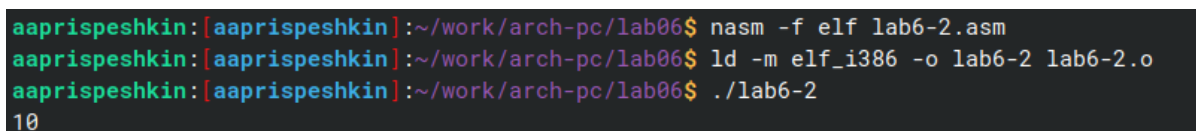
Заменим символы '6' и '4' на числа 6 и 4(Рис.9).



```
1 %include 'in_out.asm'
2 SECTION .text
3 GLOBAL _start
4 _start:
5 mov eax, 6
6 mov ebx, 0
7 add eax, ebx
8 call sprintf
9 call quit
```

Рис. 3.8: Рис.9 Код после замены символов на числа

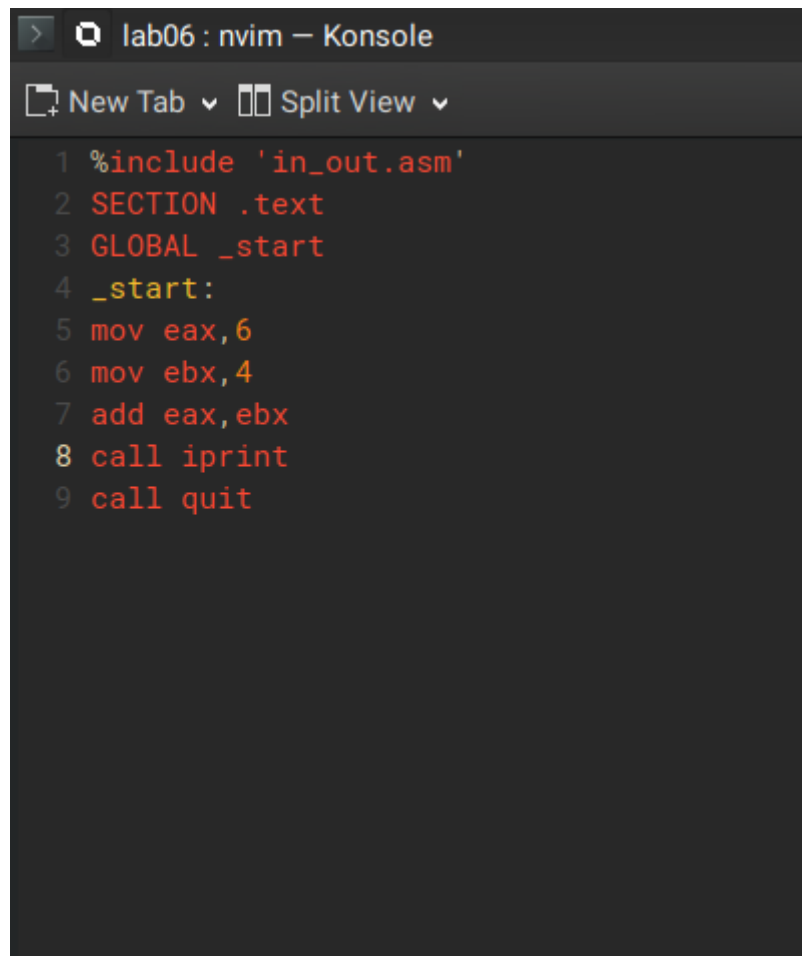
Создадим исполняемый файл и проверим его работу. Увидим, что в результате мы получаем число 10(Рис.10).



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

Рис. 3.9: Рис.10 Результат работы программы после замены символов на числа

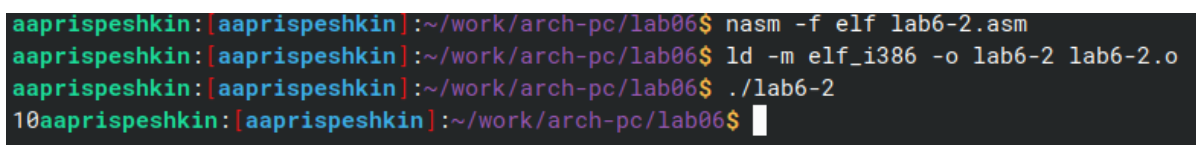
Заменяем функцию iprintLF на iprint(Рис.11).

A screenshot of a terminal window titled 'lab06: nvim — Konsole'. The window shows a text editor with assembly code. The code is as follows:

```
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. 3.10: Рис.11 Замена функции iprintLF на функцию iprint

Создадим исполняемый файл и посмотрим на результат. Увидим, что функция iprint не добавляет перенос строки после ответа вывода результата(Рис.12).

A screenshot of a terminal window showing the execution of assembly code. The commands and their outputs are as follows:

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

Рис. 3.11: Рис.12 Результат работы программы после замены iprintLF на iprint

Утилитой touch создадим файл lab6-3.asm, проверим утилитой ls(Рис.13).

```
10aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ls
in_out.asm lab6-1 lab6-1.asm lab6-1.o lab6-2 lab6-2.asm lab6-2.o lab6-3.asm
```

Рис. 3.12: Рис.13 Создание файла lab6-3.asm

Вставим туда код из листинга 6.3, для вычисления выражения  $f(x) = (5 * 2 + 3) / 3$  (Рис.14).

```
lab06: nvim — Konsole
New Tab ▾ Split View ▾
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 ; ---- Вычисление выражения
9 mov eax,5 ; EAX=5
10 mov ebx,2 ; EBX=2
11 mul ebx ; EAX=EAX*EBX
12 add eax,3 ; EAX=EAX+3
13 xor edx,edx ; обнуляем EDX для корректной работы div
14 mov ebx,3 ; EBX=3
15 div ebx ; EAX=EAX/3, EDX=остаток от деления
16 mov edi,eax ; запись результата вычисления в 'edi'
17 ; ---- Вывод результата на экран
18 mov eax,div ; вызов подпрограммы печати
19 call sprint ; сообщения 'Результат: '
20 mov eax,edi ; вызов подпрограммы печати значения
21 call iprintLF ; из 'edi' в виде символов
22 mov eax,rem ; вызов подпрограммы печати
23 call sprint ; сообщения 'Остаток от деления: '
24 mov eax,edx ; вызов подпрограммы печати значения
25 call iprintLF ; из 'edx' (остаток) в виде символов
26 call quit ; вызов подпрограммы завершения
```

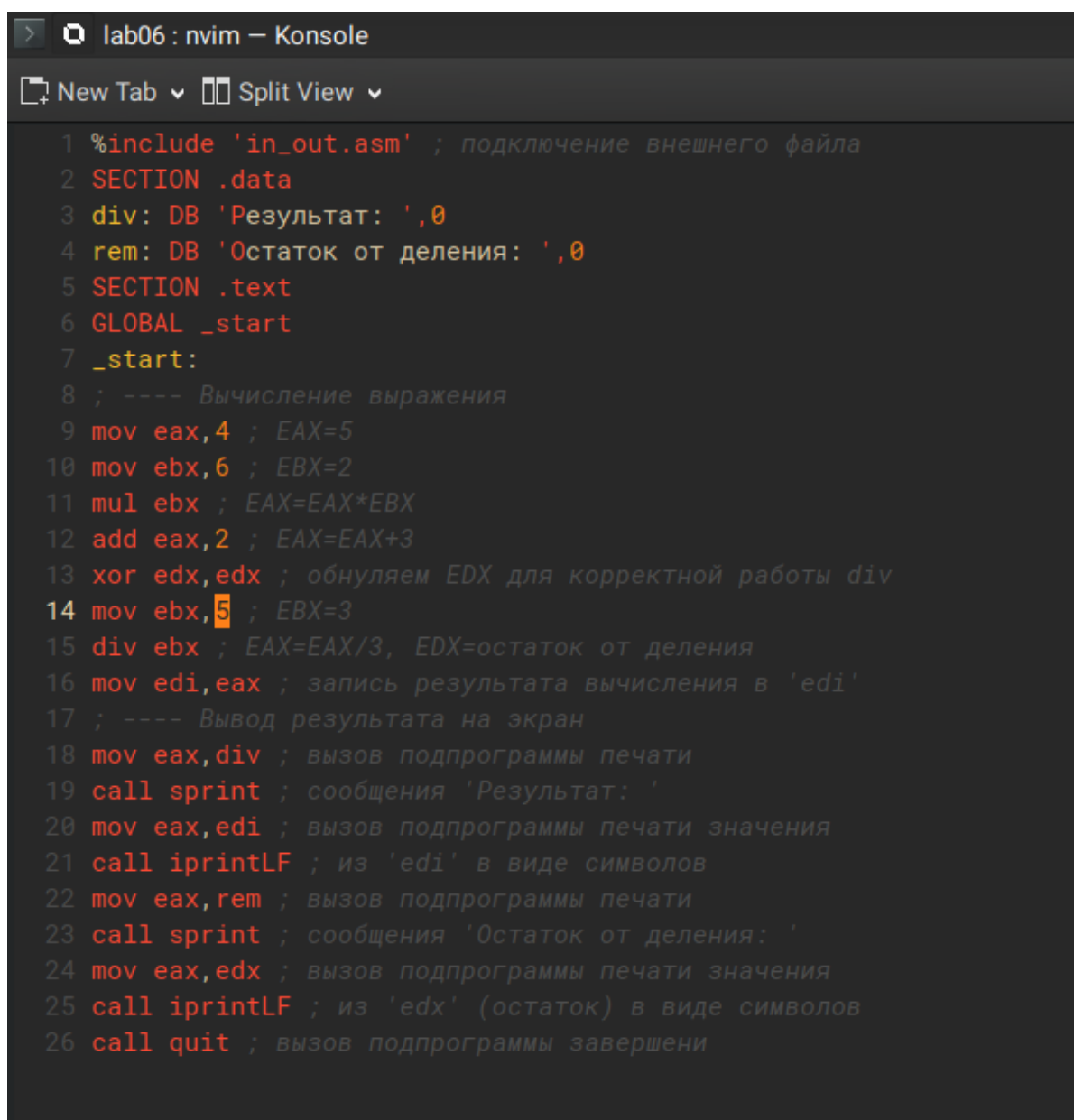
Рис. 3.13: Рис.14 Код из листинга 6.3

Создадим исполняемый файл и удостоверимся в правильности его работы(Рис.15).

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

Рис. 3.14: Рис.15 Работа файла lab6-3.asm

Заменим в нашей программе числа, для вычисления выражения  $f(x) = (4 * 6 + 2)/5$  (Рис.16).



```
lab06 : nvim — Konsole
New Tab ▾ Split View ▾
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 ; ---- Вычисление выражения
9 mov eax,4 ; EAX=5
10 mov ebx,6 ; EBX=2
11 mul ebx ; EAX=EAX*EBX
12 add eax,2 ; EAX=EAX+3
13 xor edx,edx ; обнуляем EDX для корректной работы div
14 mov ebx,5 ; EBX=3
15 div ebx ; EAX=EAX/3, EDX=остаток от деления
16 mov edi,eax ; запись результата вычисления в 'edi'
17 ; ---- Вывод результата на экран
18 mov eax,div ; вызов подпрограммы печати
19 call sprint ; сообщения 'Результат: '
20 mov eax,edi ; вызов подпрограммы печати значения
21 call iprintLF ; из 'edi' в виде символов
22 mov eax,rem ; вызов подпрограммы печати
23 call sprint ; сообщения 'Остаток от деления: '
24 mov eax,edx ; вызов подпрограммы печати значения
25 call iprintLF ; из 'edx' (остаток) в виде символов
26 call quit ; вызов подпрограммы завершения
```

Рис. 3.15: Рис.16 Код после замены чисел

Проверим результат работы программы(Рис.17).

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

Рис. 3.16: Рис.17 Результат работы программы после замены чисел

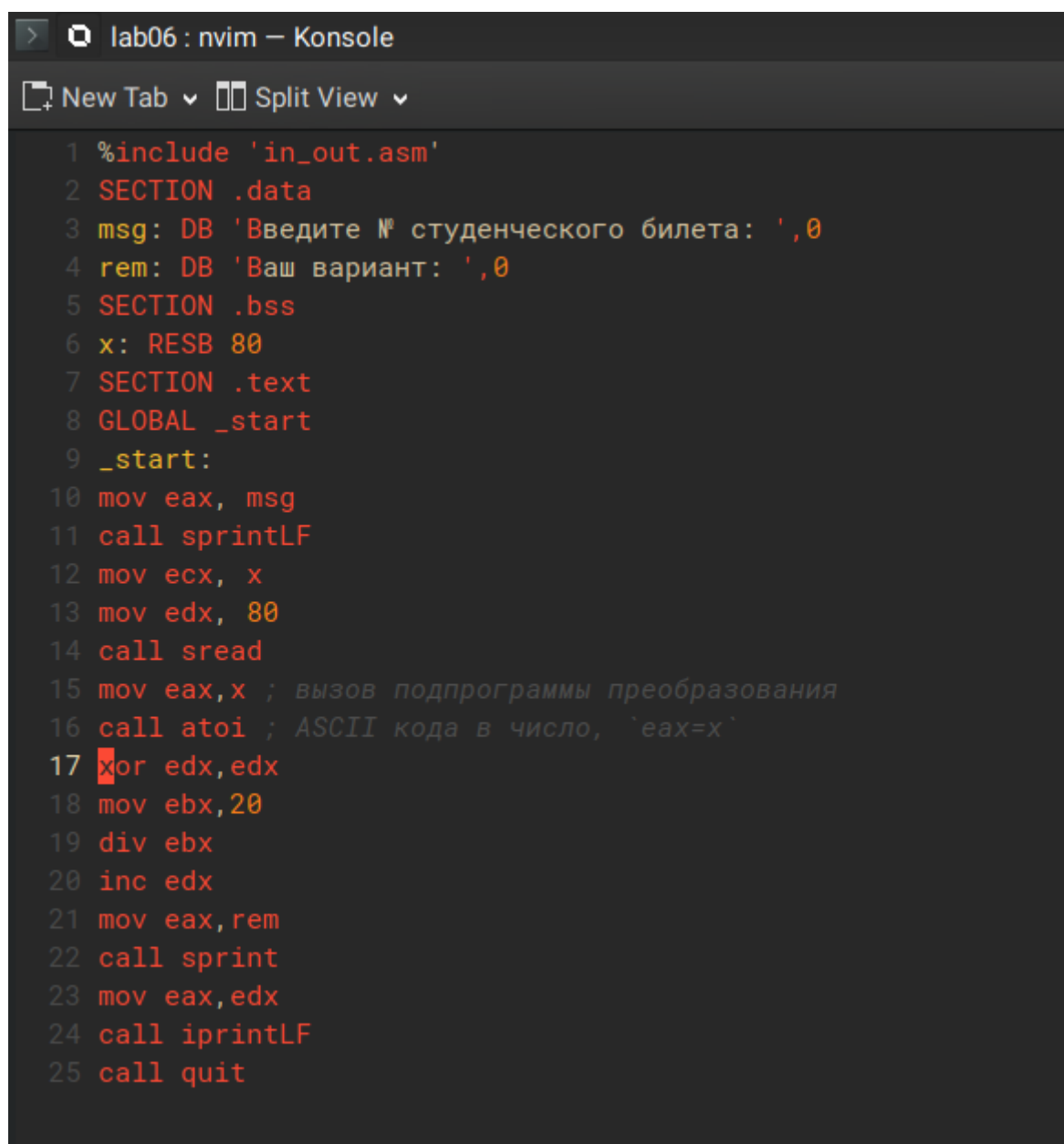
Утилитой touch создадим программу variant.asm(Рис.18).

```
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm
```

Рис. 3.17: Рис.18 Создание программы variant.asm

Вставим в программу variant.asm код из листинга 6.4 для вычисления выражения  $(S_n \bmod 20) + 1$ , где  $S_n$  – номер студенческого билета (В данном случае  $a \bmod b$  – это остаток от деления  $a$  на  $b$ )(Рис.19).



A screenshot of a terminal window titled 'lab06 : nvim — Konsole'. The window shows a text editor with assembly code. The code is as follows:

```
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. 3.18: Рис.19 Код из листинга 6.4

Создадим исполняемый файл и введём туда номер своего студенческого билета, получим вариант 1(Рис.20)

```
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ nasm -f elf variant.asm
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132239660
Ваш вариант: 1
```

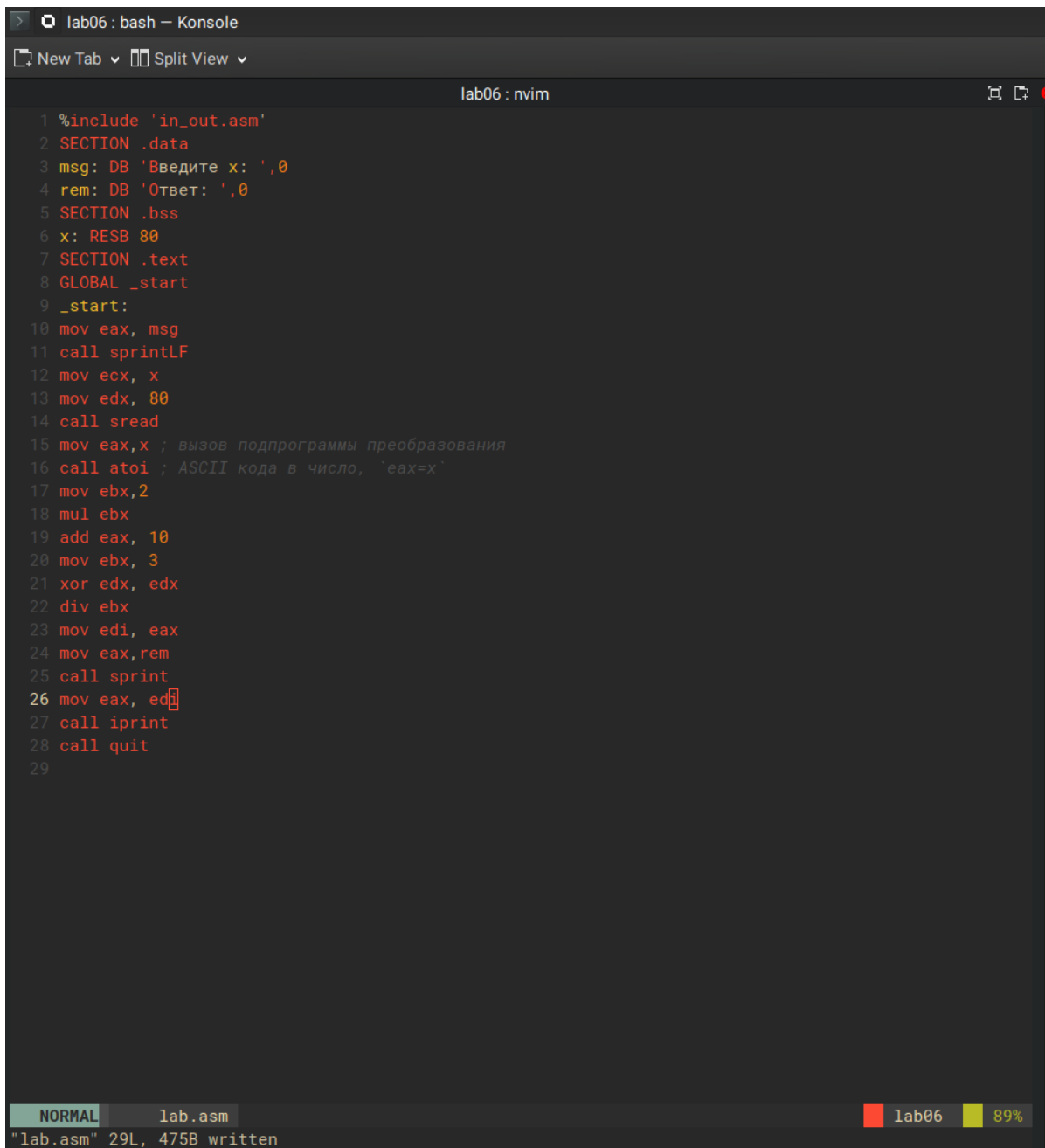
Рис. 3.19: Рис.20 Результат работы программы

## 4 Ответы на вопросы

1. За вывод сообщения отвечают строки: `mov eax, rem call sprint`
2. `mov ecx, x` - перевод адреса вводимой строки в регистр `ecx` `mov edx, 80` - запись в регистр `edx` длины вводимой строки `call sread` - вызов подпрограммы, отвечающей за ввод сообщения с клавиатуры
3. `call atoi` - вызов подпрограммы, для преобразования `ascii` символа в число
4. `xor edx, edx mov ebx, 20 div ebx inc edx`
5. `edx`
6. Увеличение значения в регистре `eax` на 1
7. `mov eax, edx call iprintLF`

## 5 Задания для самостоятельной работы

Создадим файл в котором будем работать, и напишем там код для вычисления выражения  $f(x) = (10 + 2x)/3$  (Рис.21).



```
1 %include 'in_out.asm'
2 SECTION .data
3 msg: DB 'Введите x: ',0
4 rem: DB 'Ответ: ',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 mov ebx,2
18 mul ebx
19 add eax, 10
20 mov ebx, 3
21 xor edx, edx
22 div ebx
23 mov edi, eax
24 mov eax, rem
25 call sprintf
26 mov eax, edi
27 call iprint
28 call quit
29
```

NORMAL lab.asm 89%  
"lab.asm" 29L, 475B written

Рис. 5.1: Рис.21 Код для вычисления выражения

Создадим исполняемый файл и проверим правильность написания кода со значениями x 1 и 10(Рис.22).

```
Ответ: 0aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ nasm -f elf lab.asm
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab lab.o
aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ./lab
Введите x:
1
Ответ: 4aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$ ./lab
Введите x:
10
Ответ: 10aaprispeshkin:[aaprispeshkin]:~/work/arch-pc/lab06$
```

Рис. 5.2: Рис.22 Проверка работы программы

## 6 Выводы

Я научился проводить арифметические операции на языке ассемблера NASM.

## Список литературы

[illegible]