

# **Отчёта по лабораторной работе 7**

**Освоение арифметических инструкций языка ассемблера NASM.**

Ела Абого Мигель Анхель Ндонг

# Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	21

## Список иллюстраций

2.1	Пример программы . . . . .	7
2.2	Работа программы . . . . .	7
2.3	Пример программы . . . . .	8
2.4	Работа программы . . . . .	9
2.5	Пример программы . . . . .	10
2.6	Работа программы . . . . .	10
2.7	Пример программы . . . . .	11
2.8	Работа программы . . . . .	11
2.9	Работа программы . . . . .	12
2.10	Пример программы . . . . .	13
2.11	Работа программы . . . . .	13
2.12	Пример программы . . . . .	14
2.13	Работа программы . . . . .	15
2.14	Пример программы . . . . .	16
2.15	Работа программы . . . . .	17
2.16	Пример программы . . . . .	19
2.17	Работа программы . . . . .	20

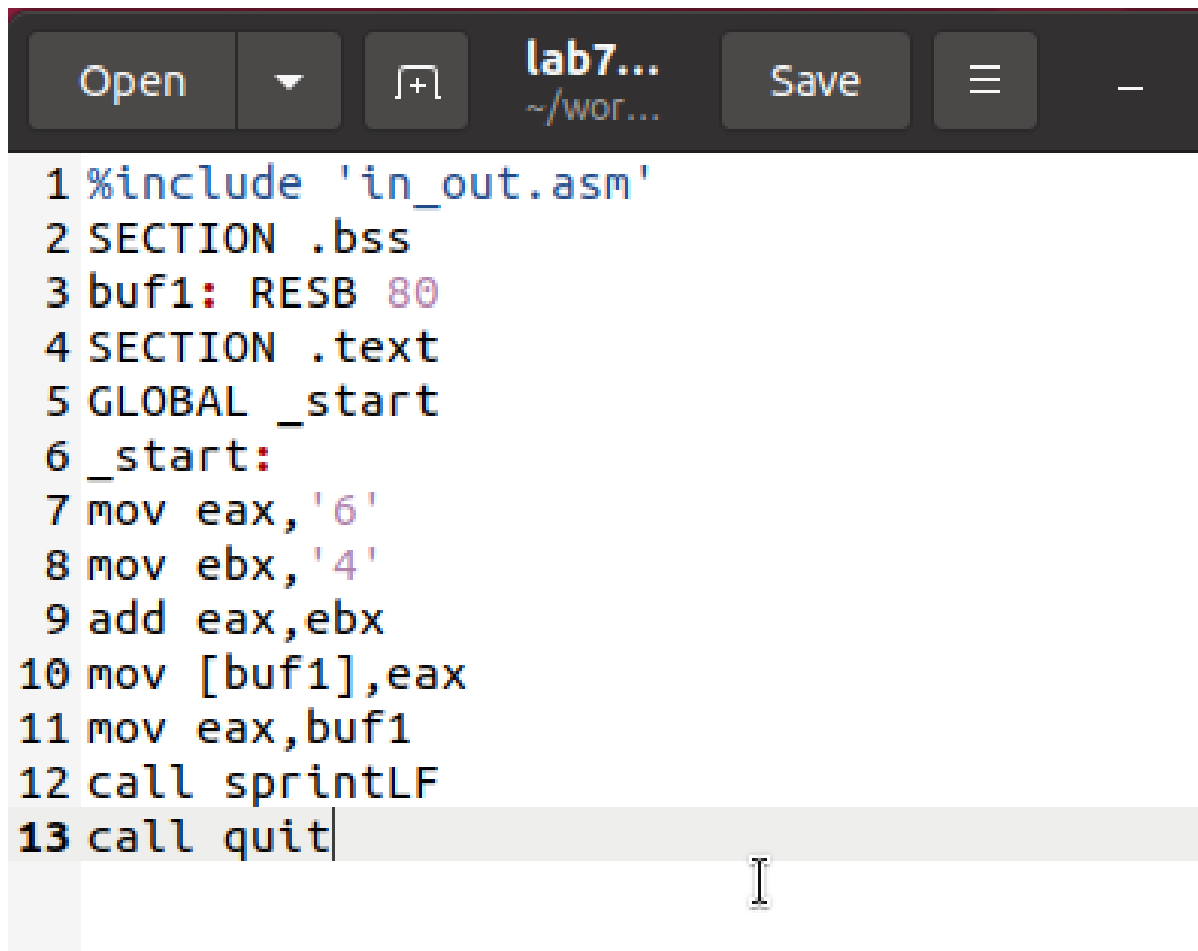
## **Список таблиц**

# 1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

## 2 Выполнение лабораторной работы

1. Создайте каталог для программ лабораторной работы № 6, перейдите в него и создайте файл lab7-1.asm:
2. Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения, записанные в регистр eax. (рис. 2.1, 2.2)

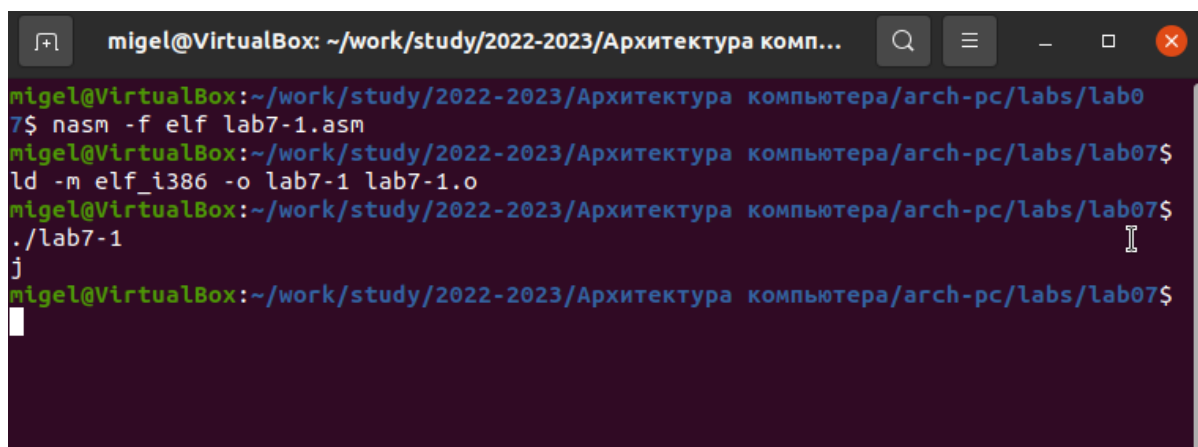


The screenshot shows a code editor window with a dark theme. The title bar includes buttons for 'Open', a dropdown menu, a file icon, the filename 'lab7...' with the path '~/.work...', a 'Save' button, and window control icons. The code is written in assembly and is as follows:

```
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
```

The cursor is positioned at the end of line 13.

Рис. 2.1: Пример программы



The screenshot shows a terminal window with a dark theme. The title bar shows the user 'migel' at 'VirtualBox' with the path '~/.work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07'. The terminal output is as follows:

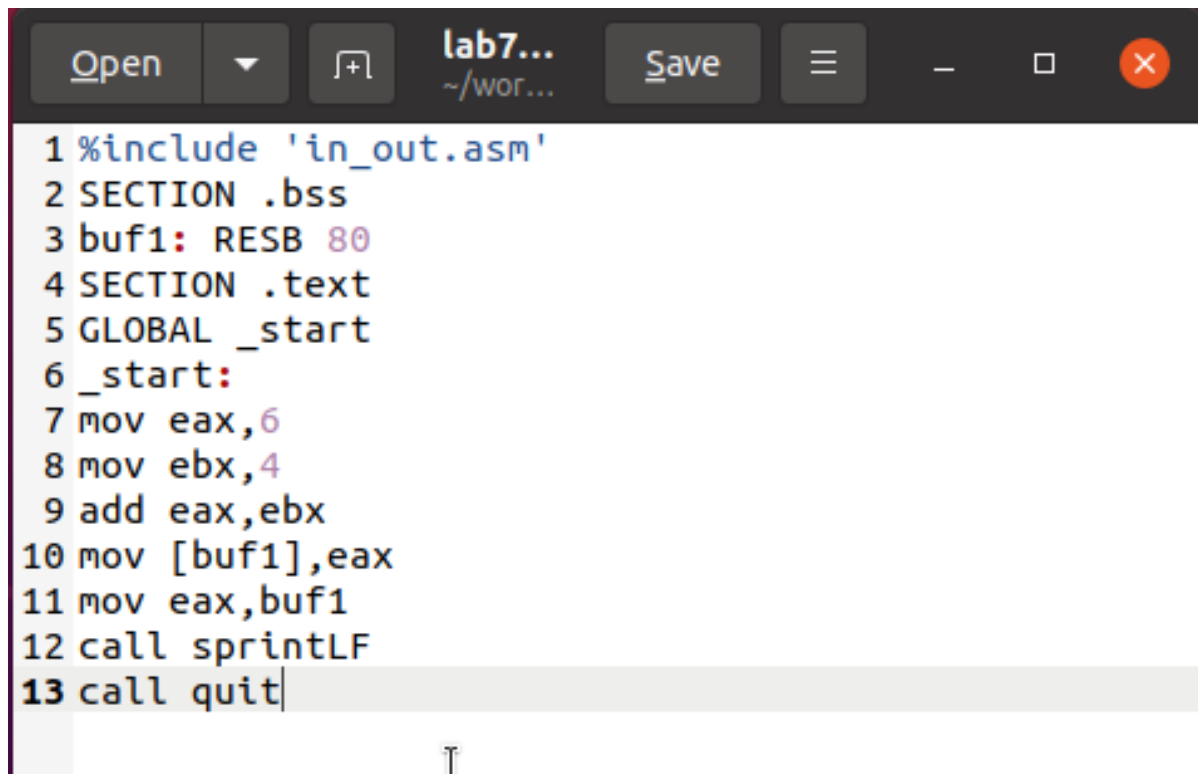
```
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-1.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-1
6
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
```

The cursor is positioned at the end of the last line.

Рис. 2.2: Работа программы

3. Далее изменим текст программы и вместо символов, запишем в регистры

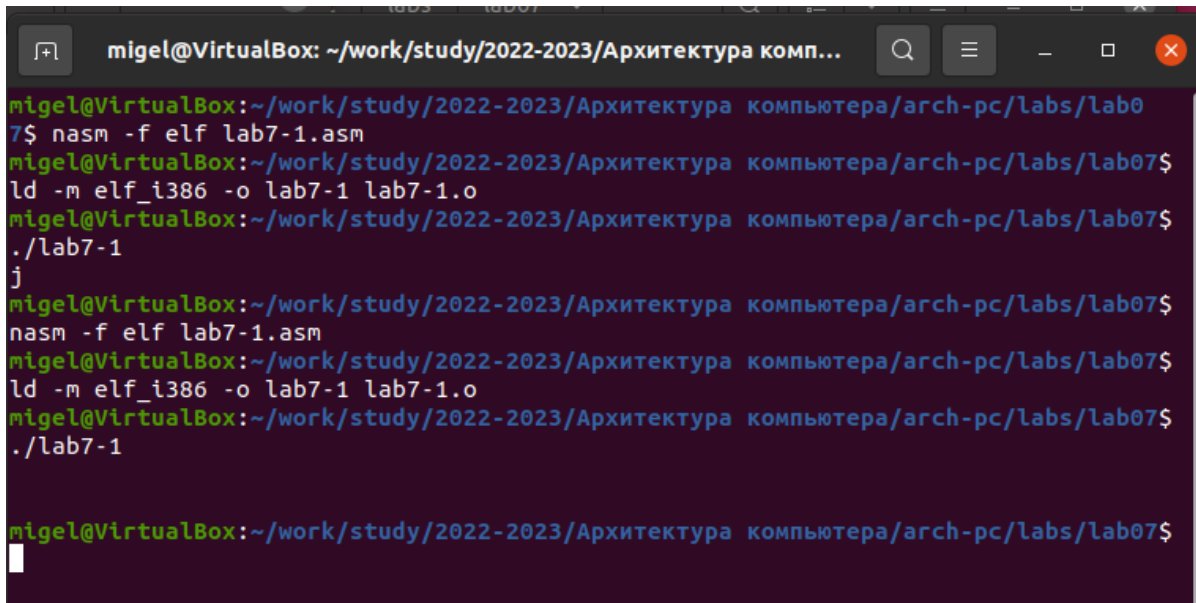
числа. Исправьте текст программы (Листинг 1) следующим образом: (рис. 2.3, 2.4)

A screenshot of a code editor window with a dark theme. The window has a title bar with 'lab7...' and a path '~\wor...'. Below the title bar are buttons for 'Open', 'Save', and a menu icon. The main area contains assembly code with line numbers 1 through 13. The code includes a header file, defines a buffer, and performs some arithmetic and system calls. The last line is highlighted with a light gray background.

```
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. 2.3: Пример программы





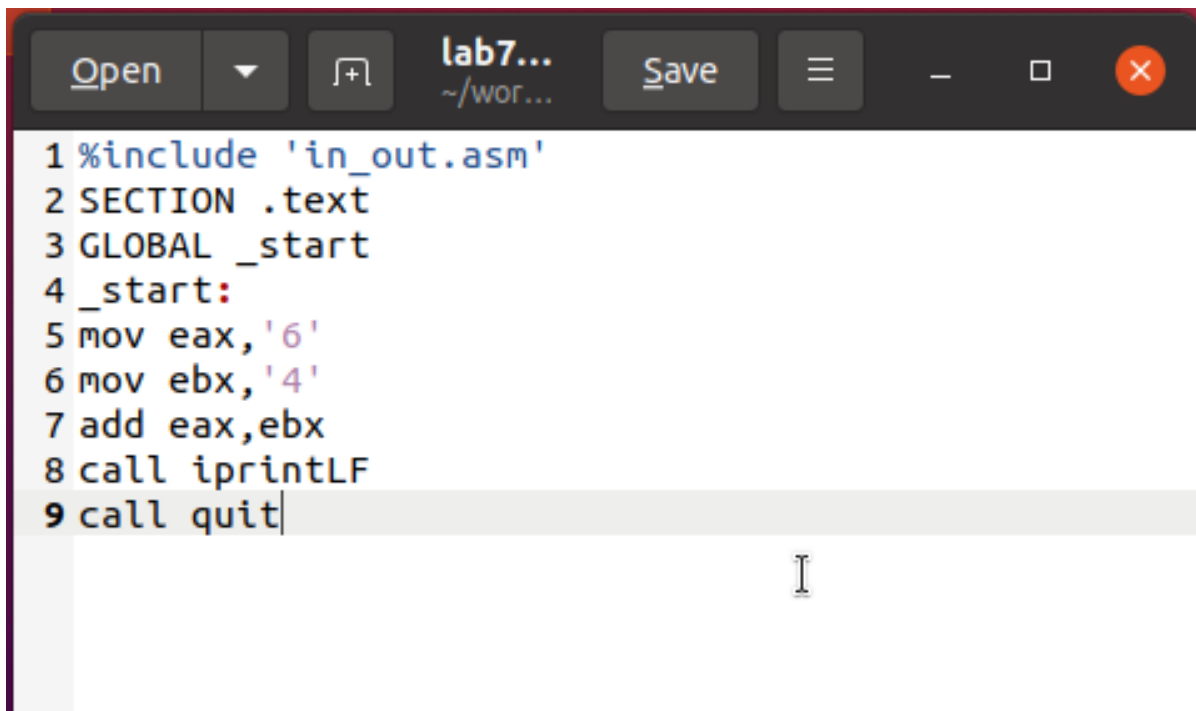
```
migel@VirtualBox: ~/work/study/2022-2023/Архитектура комп...
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-1.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-1
j
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ nasm -f elf lab7-1.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$ ./lab7-1

migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
```

Рис. 2.4: Работа программы

Никакой символ не виден, но он есть. Это возврат каретки LF.

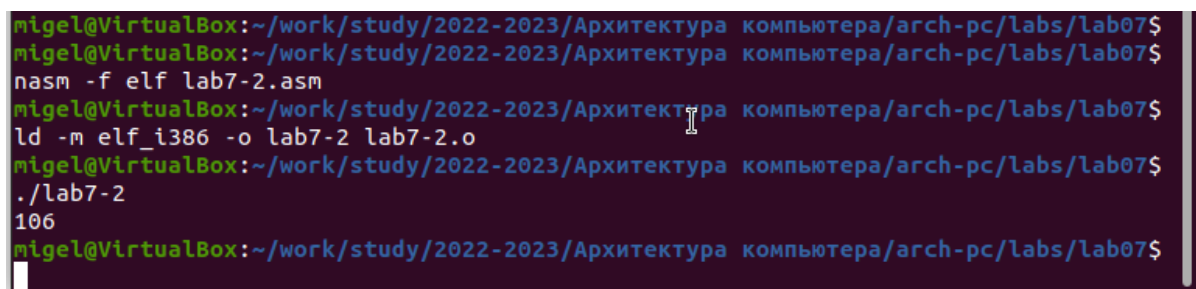
4. Как отмечалось выше, для работы с числами в файле `in_out.asm` реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Преобразуем текст программы из Листинга 7.1 с использованием этих функций. (рис. 2.5, 2.6)

A screenshot of a text editor window titled 'lab7...' with a dark theme. The editor contains assembly code for a program. 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
```

The cursor is positioned at the end of line 9.

Рис. 2.5: Пример программы

A screenshot of a terminal window with a dark background. It shows the execution of the assembly program. The commands and their outputs are as follows:

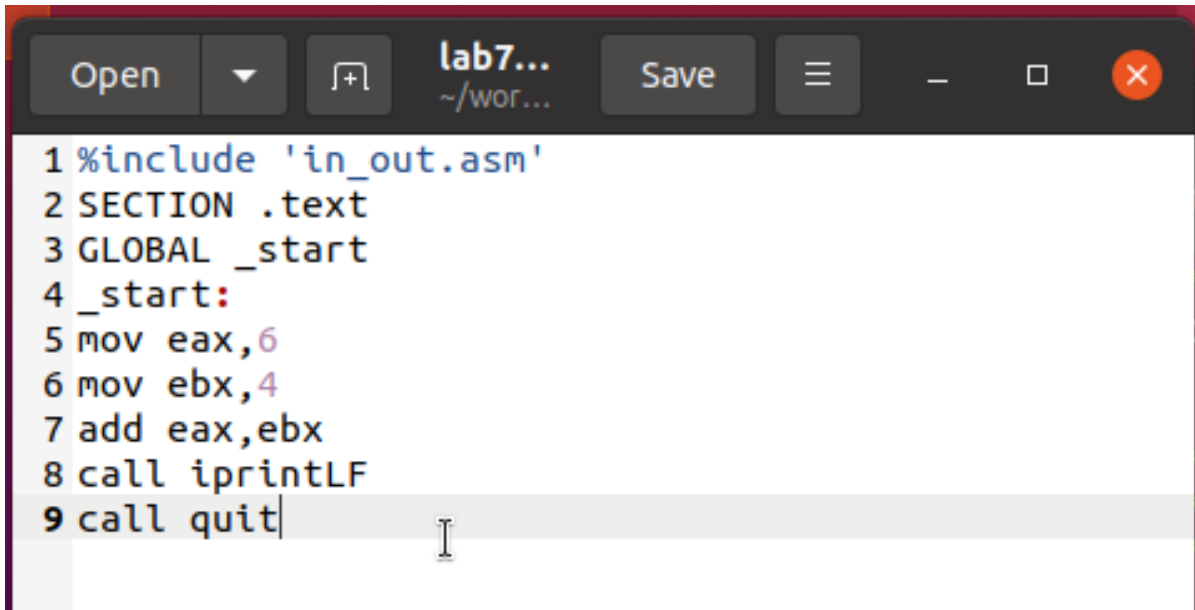
```
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-2.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-2 lab7-2.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-2
106
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
```

Рис. 2.6: Работа программы

В результате работы программы мы получим число 106. В данном случае, как и в первом, команда `add` складывает коды символов '6' и '4' ( $54+52=106$ ). Однако, в отличие от программы из листинга 7.1, функция `iprintLF` позволяет вывести число, а не символ, кодом которого является это число.

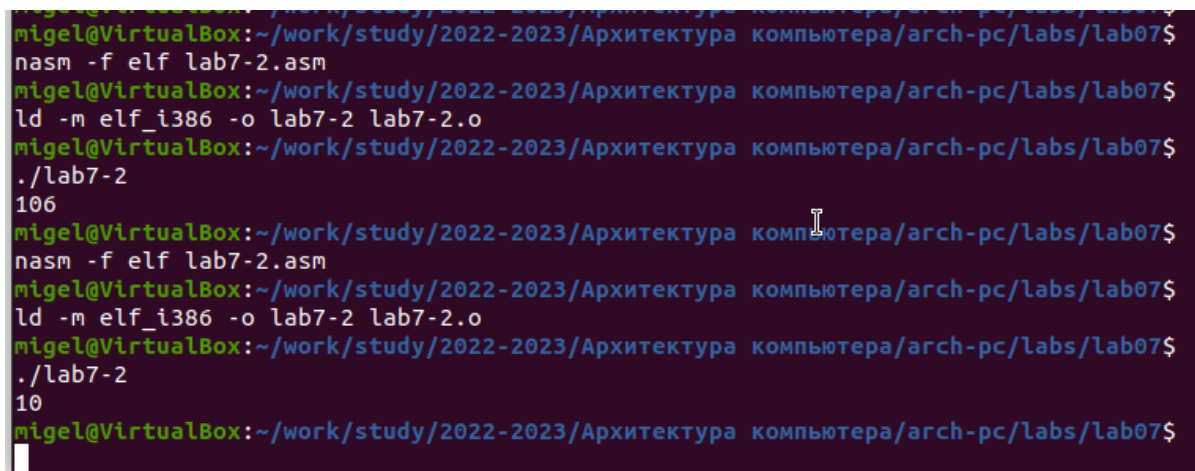
5. Аналогично предыдущему примеру изменим символы на числа. (рис. 2.7, 2.8)

Создайте исполняемый файл и запустите его. Какой результат будет получен при исполнении программы? – получили число 10



```
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. 2.7: Пример программы



```
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-2.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-2 lab7-2.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-2
106
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-2.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-2 lab7-2.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-2
10
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
```

Рис. 2.8: Работа программы

Замените функцию `iprintLF` на `iprint`. Создайте исполняемый файл и запустите его. Чем отличается вывод функций `iprintLF` и `iprint`? - Вывод отличается что нет переноса строки. (рис. 2.9)

```

migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-2.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-2 lab7-2.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-2
106
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-2.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-2 lab7-2.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-2
10
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-2.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-2 lab7-2.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-2
10migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
$

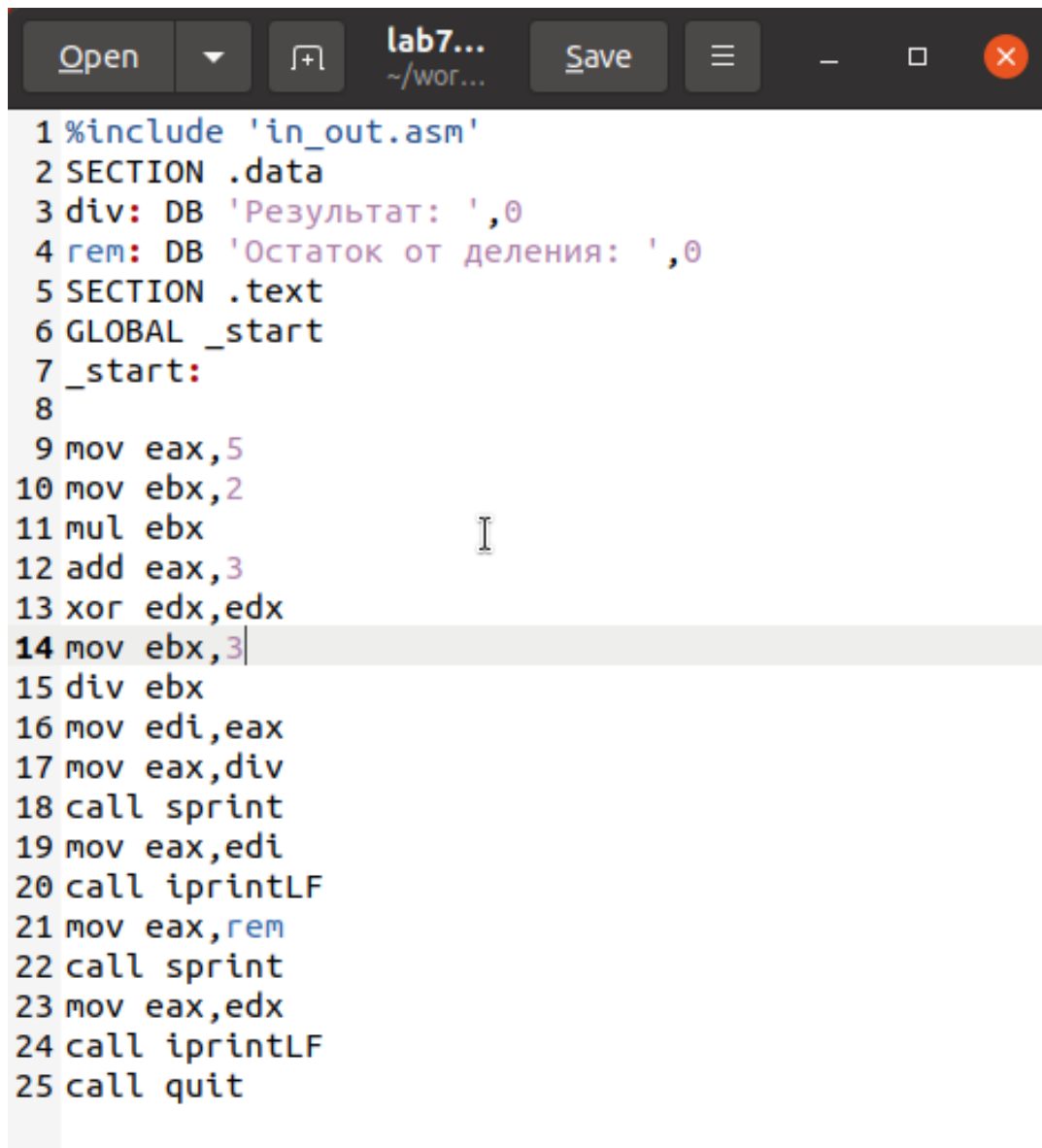
```

Рис. 2.9: Работа программы

6. В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения

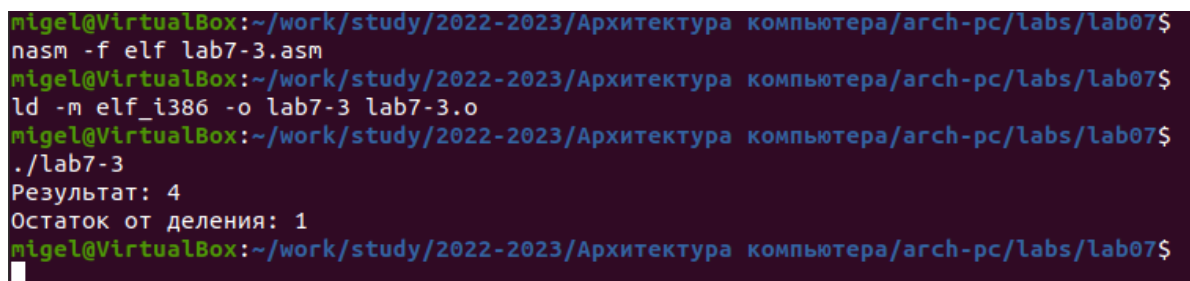
$$f(x) = (5 * 2 + 3) / 3$$

. (рис. 2.10, рис. 2.11)



```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8
9 mov eax,5
10 mov ebx,2
11 mul ebx
12 add eax,3
13 xor edx,edx
14 mov ebx,3
15 div ebx
16 mov edi,eax
17 mov eax,div
18 call sprint
19 mov eax,edi
20 call iprintLF
21 mov eax,rem
22 call sprint
23 mov eax,edx
24 call iprintLF
25 call quit
```

Рис. 2.10: Пример программы



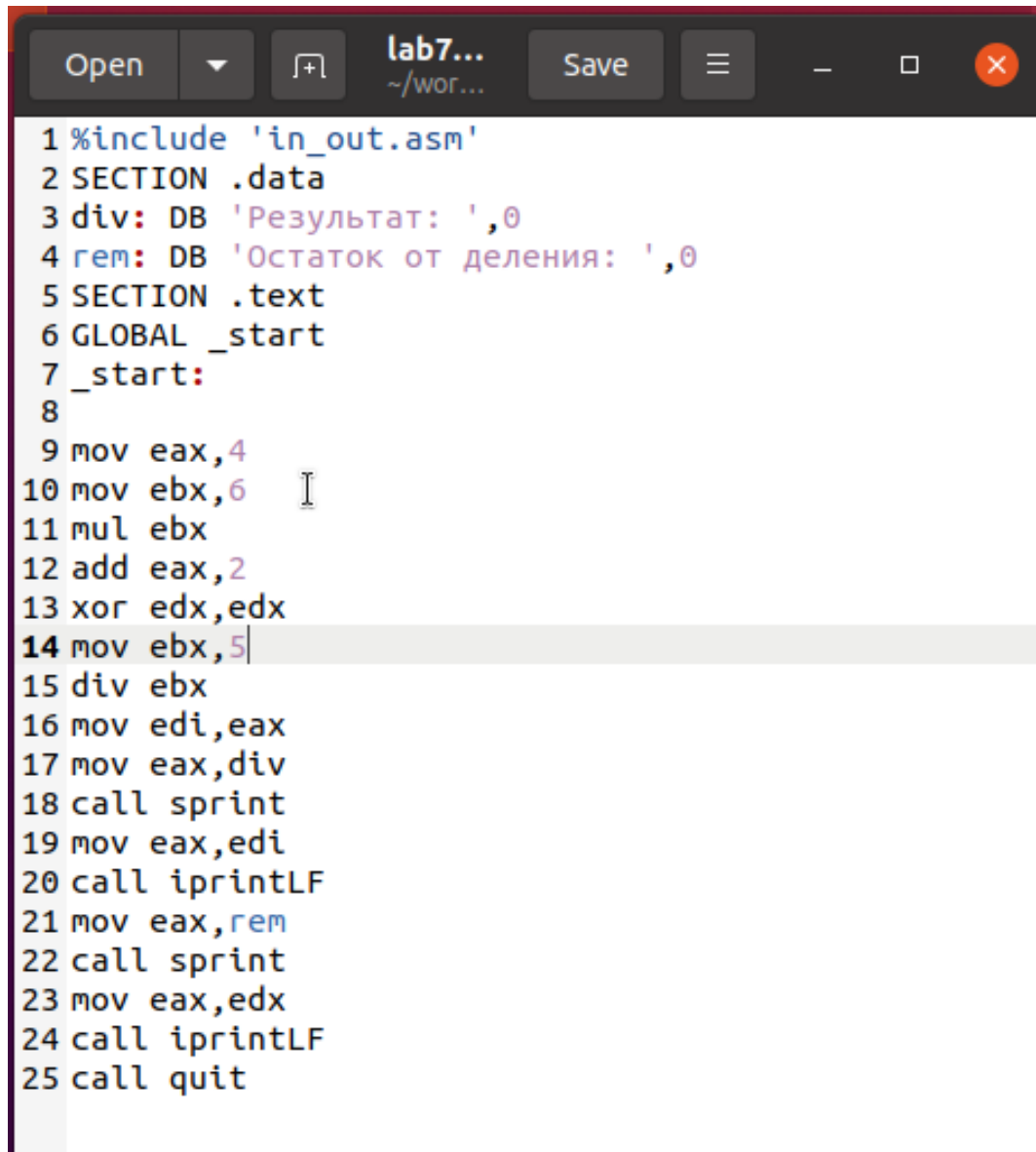
```
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-3.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-3 lab7-3.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-3
Результат: 4
Остаток от деления: 1
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
```

Рис. 2.11: Работа программы

Измените текст программы для вычисления выражения

$$f(x) = (4 * 6 + 2) / 5$$

. Создайте исполняемый файл и проверьте его работу. (рис. 2.12, рис. 2.13)



```
1 %include 'in_out.asm'
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8
9 mov eax,4
10 mov ebx,6
11 mul ebx
12 add eax,2
13 xor edx,edx
14 mov ebx,5
15 div ebx
16 mov edi,eax
17 mov eax,div
18 call sprint
19 mov eax,edi
20 call iprintLF
21 mov eax,rem
22 call sprint
23 mov eax,edx
24 call iprintLF
25 call quit
```

Рис. 2.12: Пример программы

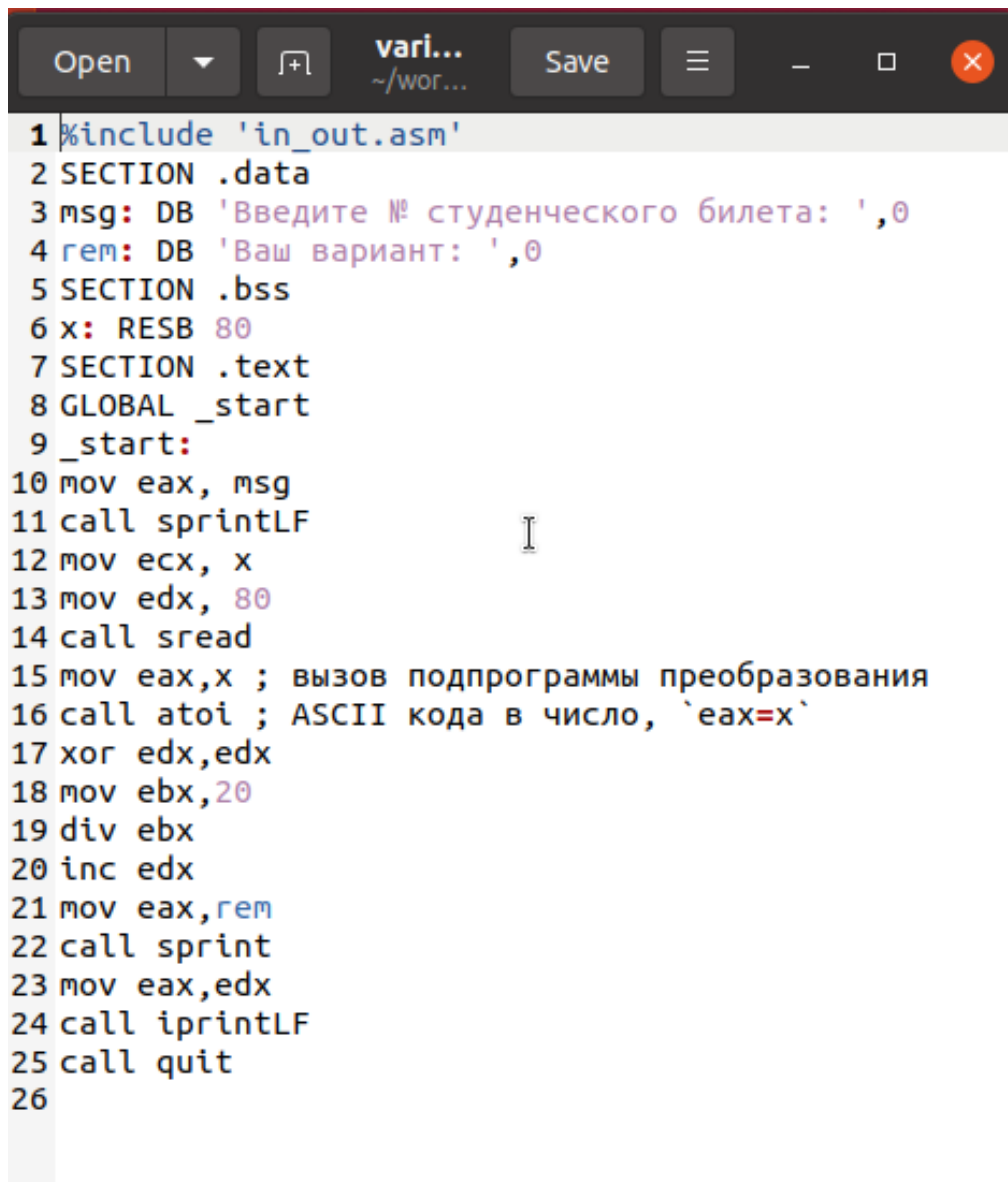
```

migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-3.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-3 lab7-3.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-3
Результат: 4
Остаток от деления: 1
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
nasm -f elf lab7-3.asm
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
ld -m elf_i386 -o lab7-3 lab7-3.o
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
./lab7-3
Результат: 5
Остаток от деления: 1
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$

```

Рис. 2.13: Работа программы

7. В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму: (рис. 2.14, рис. 2.15)



```
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
26
```

Рис. 2.14: Пример программы



```
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
nasm -f elf variant.asm  
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
ld -m elf_i386 -o variant variant.o  
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
./variant  
Введите № студенческого билета:  
1032215713  
Ваш вариант: 14  
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$
```

Рис. 2.15: Работа программы

- Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? – `mov eax,rem` – перекладывает в регистр значение переменной с фразой ‘Ваш вариант:’ `call sprint` – вызов подпрограммы вывода строки
- Для чего используются следующие инструкции? `nasm mov ecx, x mov edx, 80 call sread`

Считывает значение студбилета в переменную X из консоли

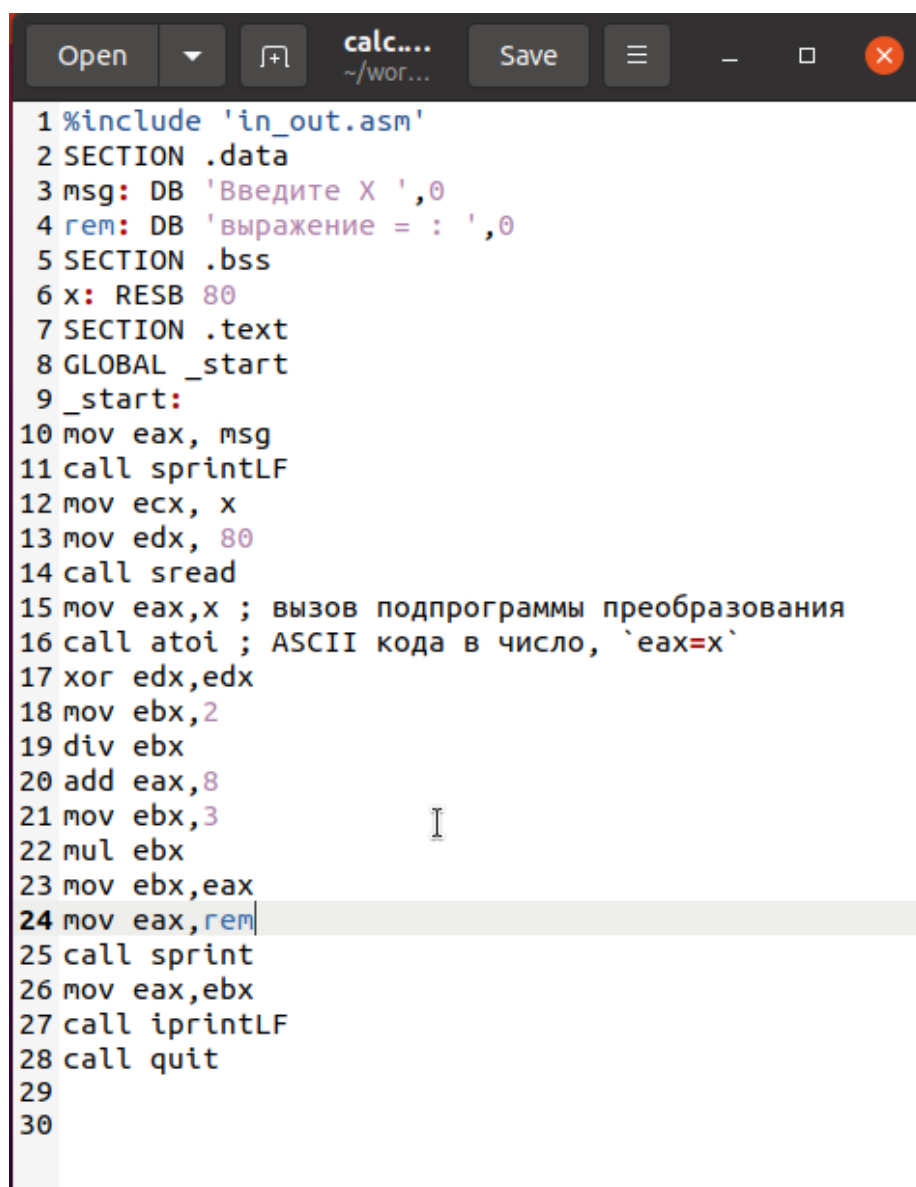
- Для чего используется инструкция “`call atoi`”? – эта подпрограмма переводит введенные символы в числовой формат
- Какие строки листинга 7.4 отвечают за вычисления варианта? `xor edx,edx mov ebx,20 div ebx`
- В какой регистр записывается остаток от деления при выполнении инструкции “`div ebx`”? 1 байт AH 2 байта DX 4 байта EDX – наш случай
- Для чего используется инструкция “`inc edx`”? по формуле вычисления варианта нужно прибавить единицу
- Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений `mov eax,edx` – результат перекладывается в регистр `eax call iprintLF` – вызов подпрограммы вывода

8. Написать программу вычисления выражения  $y = f(x)$ . Программа должна выводить 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. Создайте исполняемый файл и проверьте его работу для значений  $x_1$  и  $x_2$  из 6.3. (рис. 2.16, рис. 2.17)

Получили вариант 14 -

$$(x/2 + 8) * 3$$

для  $x=1$  и  $4$



```
1 %include 'in_out.asm'
2 SECTION .data
3 msg: DB 'Введите X ',0
4 rem: DB 'выражение = : ',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,2
19 div ebx
20 add eax,8
21 mov ebx,3
22 mul ebx
23 mov ebx,eax
24 mov eax,rem
25 call sprintf
26 mov eax,ebx
27 call iprintLF
28 call quit
29
30
```

Рис. 2.16: Пример программы

```
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
nasm -f elf calc.asm  
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
ld -m elf_i386 -o calc calc.o  
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
./calc  
Введите X  
1  
выражение = : 24  
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
./calc  
Введите X  
4  
выражение = : 30  
migel@VirtualBox:~/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab07$  
█
```

Рис. 2.17: Работа программы

## **3 Выводы**

Изучили работу с арифметическими операциями