

Лабораторная работа №6

Отчет

Устинова Виктория Вадимовна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	16

Список иллюстраций

3.1	Используем команды <code>mkdir</code> , <code>touch</code> , создаем и переходим туда с помощью команды <code>cd</code>	7
3.2	Заполняем файл	7
3.3	Запускаем файл и смотрим на его работу	8
3.4	Убираем кавычки и сохраняем	8
3.5	Наблюдаем за произошедшими изменениями	8
3.6	Используем команду <code>touch</code> и создаем файл	8
3.7	Заполняем файл	9
3.8	Запускаем файл и смотрим на его работу	9
3.9	Убираем кавычки и сохраняем	9
3.10	Запускаем файл и смотрим на его работу	9
3.11	Редактируем файл	10
3.12	Запускаем файл и смотрим на его работу	10
3.13	Создаем файл	10
3.14	Заполняем файл	11
3.15	Запускаем файл и смотрим на его работу	11
3.16	Редактируем наш файл	12
3.17	Запускаем файл и смотрим на его работу	12
3.18	Заполняем этот файл	13
3.19	Запускаем файл и смотрим на его работу	13
3.20	Заполняем этот файл, чтобы решалось уравнение $(8x-6)/2$	15
3.21	Программа работает корректно	15
3.22	Программа работает корректно	15

Список таблиц

1 Цель работы

Освоить арифметические инструкции языка ассемблера NASM и написать программы для вычисления арифметических действий с помощью неизвестной.

2 Задание

Выполнить лабораторную и написать программу для вычисления уравнения.

3 Выполнение лабораторной работы

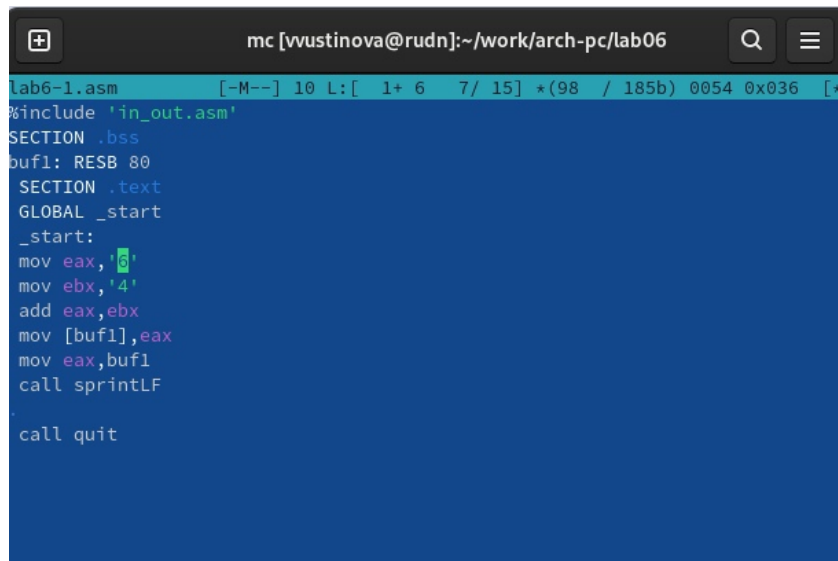
Символьные и численные данные в NASM

Создаем каталог lab6, а в нем файл lab6-1.asm(рис. 3.1).

```
vvustinova@rudn:~$ cd ~/work/arch-pc/lab06
vvustinova@rudn:~/work/arch-pc/lab06$
```

Рис. 3.1: Используем команды mkdir, touch, создаем и переходим туда с помощью команды cd

Открываем файл в Midnight Commander и заполняем его в соответствии с листингом 6.1(рис. 3.2).



```
mc [vvustinova@rudn]:~/work/arch-pc/lab06
lab6-1.asm [-M--] 10 L: [ 1+ 6 7/ 15] *(98 / 185b) 0054 0x036 [*]
%include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, '0'
mov ebx, '4'
add eax, ebx
mov [buf1], eax
mov eax, buf1
call sprintf
call quit
```

Рис. 3.2: Заполняем файл

Необходимо запустить файл(рис. 3.3).

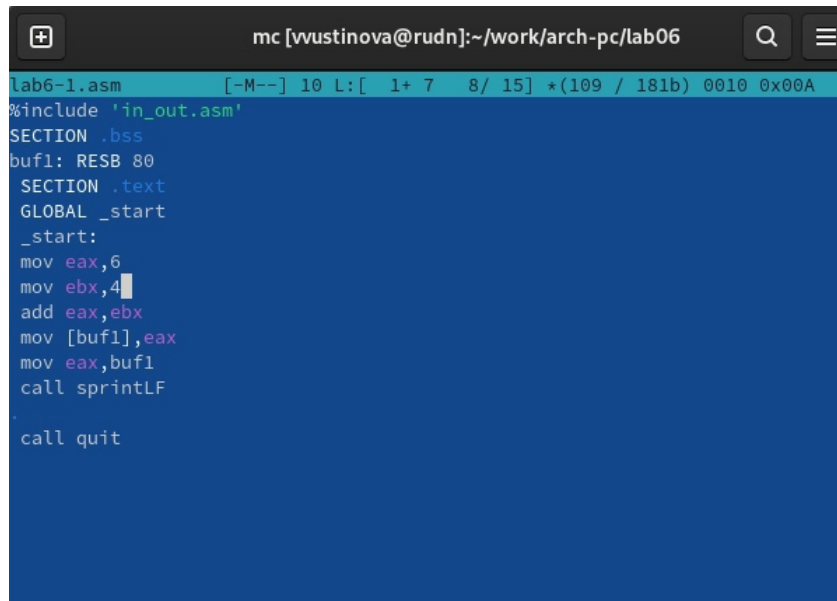
```

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

```

Рис. 3.3: Запускаем файл и смотрим на его работу

Открываем файл и редактируем его(рис. 3.4).



```

lab6-1.asm  [-M--] 10 L:[ 1+ 7 8/ 15] *(109 / 181b) 0010 0x00A
#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 printf
call quit

```

Рис. 3.4: Убираем кавычки и сохраняем

Запускаем наш файл(рис. 3.5).

```

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

```

Рис. 3.5: Наблюдаем за произошедшими изменениями

Создаем новый файл lab6-2.asm(рис. 3.6).

```

vvustinova@rudn:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-2.asm
vvustinova@rudn:~/work/arch-pc/lab06$ mc

```

Рис. 3.6: Используем команду touch и создаем файл

Заполняем файл в соответствии с листингом 6.2(рис. 3.7).


```

lab6-2.asm      [----]  9 L:[ 1+10 11/ 11] *(119 / 119b) <EOF>
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:

mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF

call quit

```

Рис. 3.7: Заполняем файл

Необходимо запустить файл(рис. 3.8).

```

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

```

Рис. 3.8: Запускаем файл и смотрим на его работу

Открываем файл и редактируем его(рис. 3.9).

```

lab6-2.asm      [-M--]  8 L:[ 1+ 6 7/ 11] *(77 / 115b) 0052 0x034
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
call iprintLF

call quit

```

Рис. 3.9: Убираем кавычки и сохраняем

```

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

```

Рис. 3.10: Запускаем файл и смотрим на его работу

Снова открываем файл для редактирования и меняем `iprintLF` на `iprint`(рис. 3.11).

```
lab6-2.asm [-M--] 11 L:[ 1+ 8 9/ 11] *(102 / 113b) 0010 0x00
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
call iprint

call quit
```

Рис. 3.11: Редактируем файл

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

Рис. 3.12: Запускаем файл и смотрим на его работу

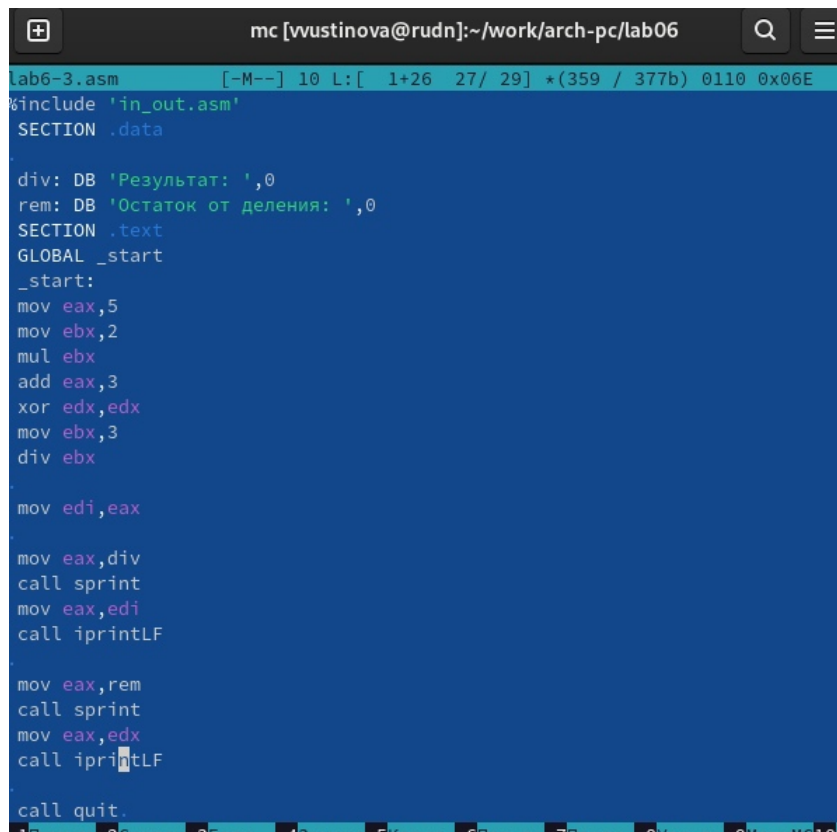
Выполнение арифметических операций в NASM

Создаем новый файл `lab6-3.asm`(рис. 3.13).

```
10vvustinova@rudn:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm
vvustinova@rudn:~/work/arch-pc/lab06$
```

Рис. 3.13: Создаем файл

Открываем файл и редактируем в соответствии с листингом 6.3(рис. 3.14).



```
lab6-3.asm [-M--] 10 L:[ 1+26 27/ 29] *(359 / 377b) 0110 0x06E
#include 'in_out.asm'
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
mov eax,5
mov ebx,2
mul ebx
add eax,3
xor edx,edx
mov ebx,3
div ebx

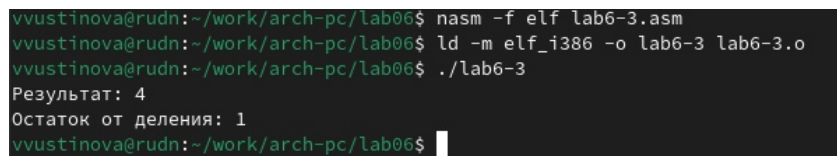
mov edi,eax

mov eax,div
call sprint
mov eax,edi
call iprintLF

mov eax,rem
call sprint
mov eax,edx
call iprintLF

call quit.
```

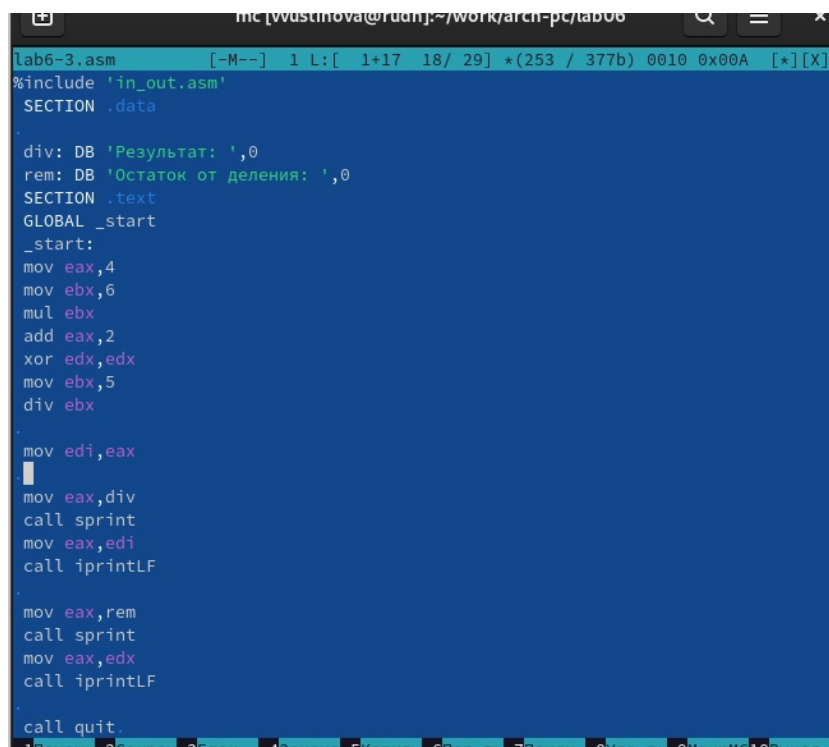
Рис. 3.14: Заполняем файл



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

Рис. 3.15: Запускаем файл и смотрим на его работу

Открываем файл и редактируем его для вычисления выражения $f(x) = (4 * 6 + 2)/5$ (рис. 3.16).



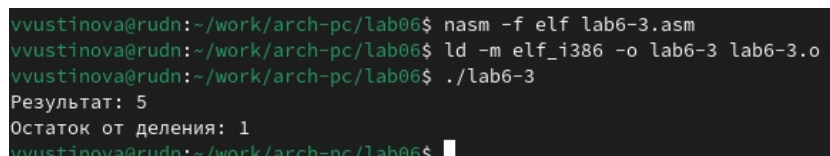
```
lab6-3.asm [-M--] 1 L: [ 1+17 18/ 29] *(253 / 377b) 0010 0x00A [*][X]
%include 'in_out.asm'
SECTION .data
.
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
mov eax,4
mov ebx,6
mul ebx
add eax,2
xor edx,edx
mov ebx,5
div ebx

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
```

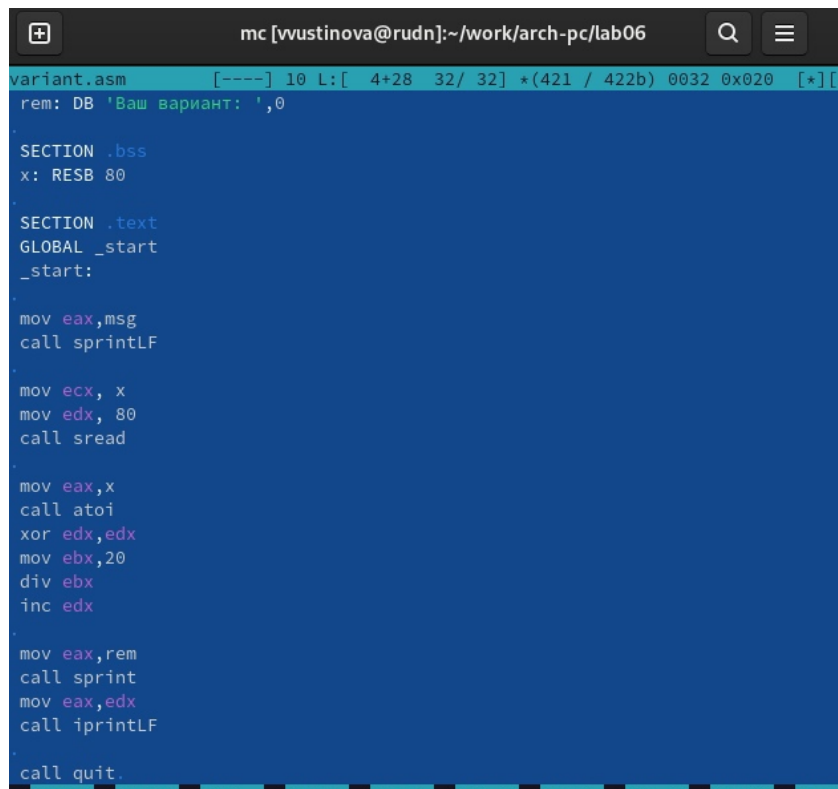
Рис. 3.16: Редактируем наш файл



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

Рис. 3.17: Запускаем файл и смотрим на его работу

Создаем новый файл variant.asm и открываем его(рис. 3.18).



```
variant.asm [----] 10 L:[ 4+28 32/ 32] *(421 / 422b) 0032 0x020 [*][
rem: DB 'Ваш вариант: ',0

SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax,msg
call sprintLF

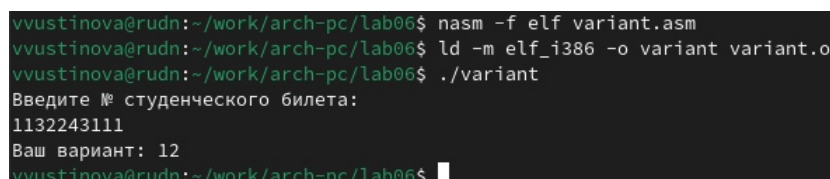
mov ecx,x
mov edx,80
call sread

mov eax,x
call atoi
xor edx,edx
mov ebx,20
div ebx
inc edx

mov eax,rem
call sprint
mov eax,edx
call iprintLF

call quit.
```

Рис. 3.18: Заполняем этот файл



```
vvustinova@rudn:~/work/arch-pc/lab06$ nasm -f elf variant.asm
vvustinova@rudn:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
vvustinova@rudn:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132243111
Ваш вариант: 12
vvustinova@rudn:~/work/arch-pc/lab06$
```

Рис. 3.19: Запускаем файл и смотрим на его работу

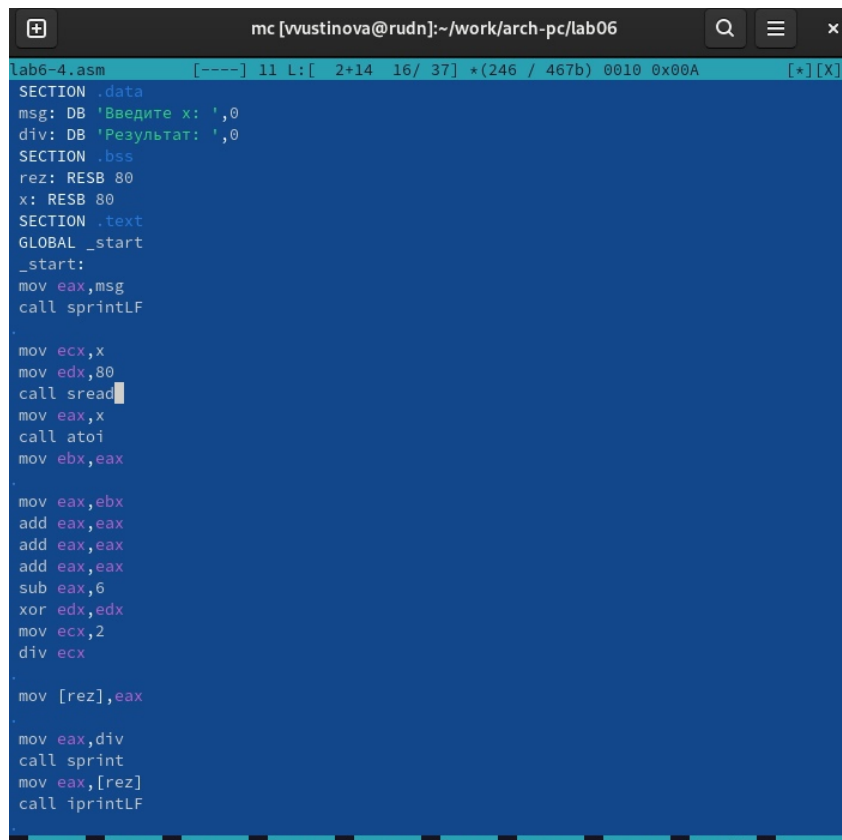
Ответы на вопросы

1. Строка “mov eax,rem” и строка “call sprint” отвечают за вывод на экран сообщения ‘Ваш вариант:’.
2. Эти инструкции используются для чтения строки с вводом данных от пользователя. Начальный адрес строки сохраняется в регистре ecx, а количество символов в строке (максимальное количество символов, которое может быть считано) сохраняется в регистре edx. Затем вызывается процедура sread, которая выполняет чтение строки.

3. Инструкция “call atoi” используется для преобразования строки в целое число. Она принимает адрес строки в регистре еах и возвращает полученное число в регистре еах.
4. Строка “xor edx,edx” обнуляет регистр edx перед выполнением деления. Строка “mov ebx,20” загружает значение 20 в регистр ebx. Строка “div ebx” выполняет деление регистра еах на значение регистра ebx с сохранением частного в регистре еах и остатка в регистре edx.
5. Остаток от деления записывается в регистр edx.
6. Инструкция “inc edx” используется для увеличения значения в регистре edx на 1. В данном случае, она увеличивает остаток от деления на 1.
7. Строка “mov еах,edx” отправляет значение остатка от деления в регистр еах. Строка “call iprintLF” вызывает процедуру iprintLF для вывода значения на экран вместе с переводом строки.

Задание для самостоятельной работы

Создаем новый файл lab6-4.asm и открываем его (рис. 3.20).



```
lab6-4.asm [----] 11 L:[ 2+14 16/ 37] *(246 / 467b) 0010 0x00A [*] [X]
SECTION .data
msg: DB 'Введите x: ',0
div: DB 'Результат: ',0
SECTION .bss
rez: RESB 80
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,msg
call sprintLF

mov ecx,x
mov edx,80
call sread
mov eax,x
call atoi
mov ebx,eax

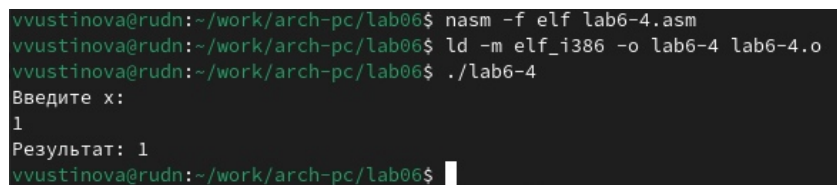
mov eax,ebx
add eax,eax
add eax,eax
add eax,eax
sub eax,6
xor edx,edx
mov ecx,2
div ecx

mov [rez],eax

mov eax,div
call sprint
mov eax,[rez]
call iprintLF
```

Рис. 3.20: Заполняем этот файл, чтобы решалось уравнение $(8x-6)/2$

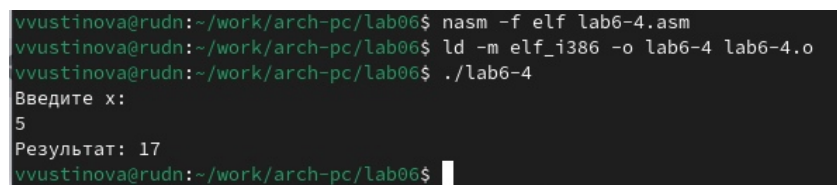
Проверяем программу для $x=1$ (рис. 3.21).



```
vvustinova@rudn:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
vvustinova@rudn:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
vvustinova@rudn:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
1
Результат: 1
vvustinova@rudn:~/work/arch-pc/lab06$
```

Рис. 3.21: Программа работает корректно

Проверяем программу для $x=5$ (рис. 3.22).



```
vvustinova@rudn:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
vvustinova@rudn:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
vvustinova@rudn:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
5
Результат: 17
vvustinova@rudn:~/work/arch-pc/lab06$
```

Рис. 3.22: Программа работает корректно

4 Выводы

Мы приобрели навыки создания исполнительных файлов для решения выражений и освоили арифметические инструкции в NASM.