

Отчёта по лабораторной работе №6

Арифметические операции в NASM.

Шакер Альсалем

Содержание

Список иллюстраций

3.1	Создаем каталог с помощью команды mkdir и файл с помощью команды touch .	6
3.2	Заполняем файл	7
3.3	Запускаем файл и смотрим на его работу	7 3.4
	Изменяем файл	8
3.5	Запускаем файл и смотрим на его работу	8 3.6
	Создаем файл	8 3.7
	Заполняем файл	9
3.8	Смотрим на работу программы	9 3.9
	Изменяем файл.....	10
•	Смотрим на работу программы.....	10
•	Изменяем файл.....	11
•	Смотрим на работу программы.....	11
•	Создаем файл.....	11

• Заполняем файл	12
• Смотрим на результат работы программы	12
• Редактируем файл	13
• Смотрим на результат работы программы	13
• Создаем файл	13
• Заполняем файл	14
• Проверяемс результат работы программы.....	14
• Создаем файл	15
• Заполняем файл	16
• Проверяем работу программы.....	16
• Проверяем работу программы.....	17

• Цель работы

Освоить арифметических инструкций языка ассемблера NASM и написать программы для вычисления арифметических выражений с неизвестной.

• Задание

Написать программы для решения выражений.

- **Выполнение лабораторной работы**
- **Символьные и численные данные в NASM**

Создаем каталог для программ ЛБ6, и в нем создаем файл (рис. 3.1).

```
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ mkdir ~/work/arch-pc/lab06
mkdir: cannot create directory '/home/alsalemsaker/work/arch-pc/lab06': File exists
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ cd ~/work/arch-pc/lab06
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ touch lab6-1.asm
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$
```

Рис. 3.1: Создаем каталог с помощью команды `mkdir` и файл с помощью команды `touch`

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



Рис. 3.2: Заполняем файл

Создаем исполняемый файл и запускаем его (рис. 3.3).

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

Рис. 3.3: Запускаем файл и смотрим на его работу

Снова открываем файл для редактирования и убираем кавычки с числовых значений (рис. 3.4).

```
GNU nano 6.2 /home/alsalemsaker/work/arch-pc/lab06/lab6-1.asm *
#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

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute  ^C Location  M-U Undo
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify  ^/ Go To Line M-E Redo
```

Рис. 3.4: Изменяем файл

Создаем исполняемый файл и запускаем его (рис. 3.5).

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

alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$
```

Рис. 3.5: Запускаем файл и смотрим на его работу

Создаем новый файл в каталоге (рис. 3.6).

```
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab-2.asm
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$
```

Рис. 3.6: Создаем файл

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

```
GNU nano 6.2 /home/alsalemshaker/work/arch-pc/lab06/lab-2.asm *
%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 iprintLF

call quit

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location  M-U Undo
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line M-E Redo
```

Рис. 3.7: Заполняем файл

Создаем исполняемый файл и запускаем его (рис. 3.8).

```
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ ./lab6-2
134520832
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$
```

Рис. 3.8: Смотрим на работу программы

Снова открываем файл для редактирования и убираем кавычки с числовых значений (рис. 3.9).

```
GNU nano 6.2 /home/alsalemshaker/work/arch-pc/lab06/lab6-2.asm
%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 iprint

call quit

[ Read 14 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location  M-U Undo
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line M-E Redo
```

Рис. 3.9: Изменяем файл

Создаем исполняемый файл и запускаем его (рис. 3.10).mc

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

Рис. 3.10: Смотрим на работу программы

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

```
+ alsalemshaker@alsalemshaker-VirtualBox: ~/work/arch-pc/lab06
GNU nano 6.2 /home/alsalemshaker/work/arch-pc/lab06/lab6-2.asm
%include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint

call quit
```

Рис. 3.11: Изменяем файл

Создаем исполняемый файл и запускаем его (рис. 3.12).

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

Рис. 3.12: Смотрим на работу программы

Вывод функций `iprintLF` и `iprint` отличаются только тем, что `LF` переносит новую строку.

- **Выполнение арифметических операций в NASM**

Создаем новый файл в каталоге (рис. 3.13).

```
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/lab6-3.asm
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$
```

Рис. 3.13: Создаем файл

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

```
alsalemsaker@alsalemsaker-VirtualBox: /home/alsalemsaker/work/arch-pc
GNU nano 6.2
%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
```

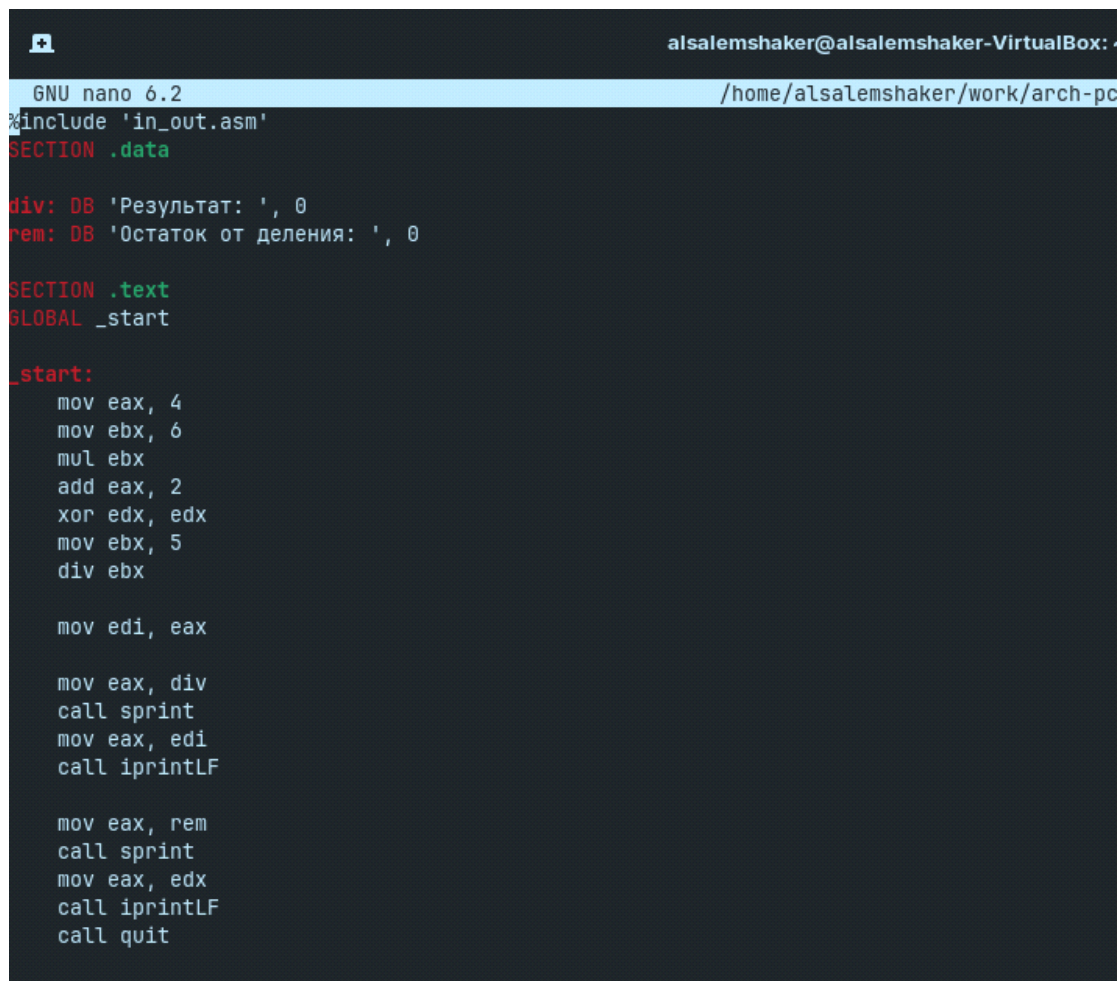
Рис. 3.14: Заполняем файл

Создаем исполняемый файл и запускаем его (рис. 3.15).

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

Рис. 3.15: Смотрим на результат работы программы

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



```
alsalemsaker@alsalemsaker-VirtualBox: ~  
GNU nano 6.2 /home/alsalemsaker/work/arch-pc  
%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 файл

Компилируем файл и запускаем программу (рис. 3.17).



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

Рис. 3.17: Смотрим на результат работы программы

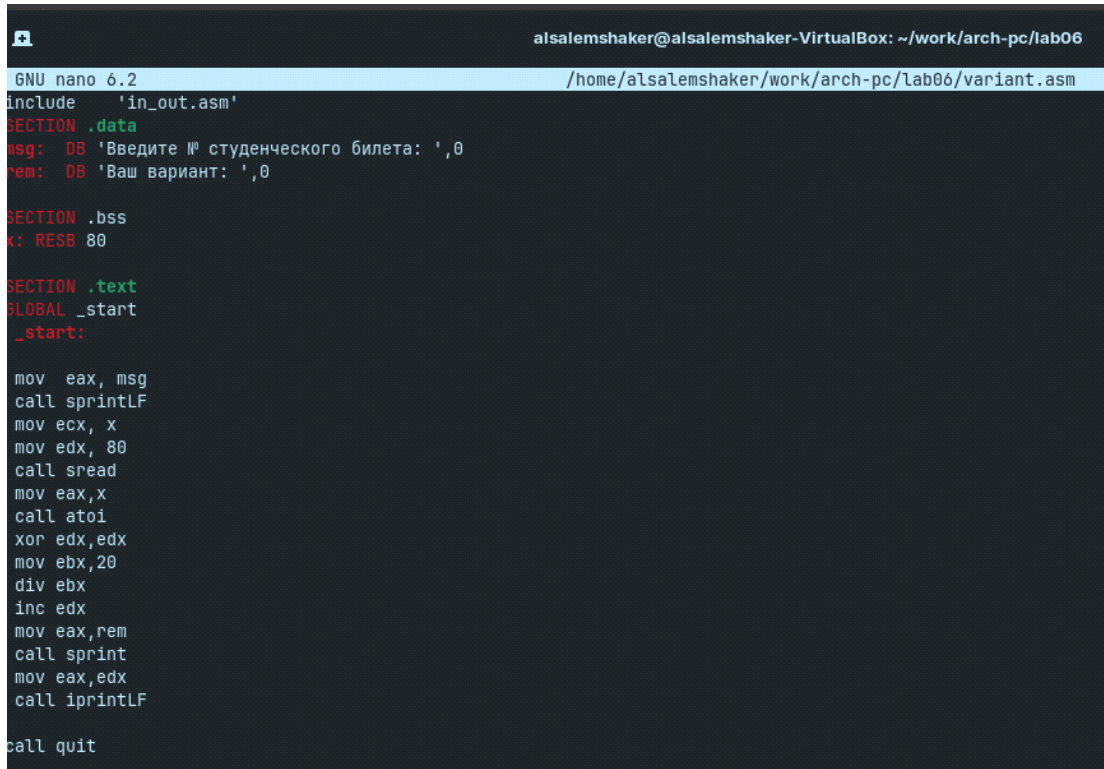
Создаем новый файл в каталоге (рис. 3.18).



```
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ touch ~/work/arch-pc/lab06/variant.asm  
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ mc
```

Рис. 3.18: Создаем файл

Открываем файл и редактируем в соответствии с листингом 6.4 (рис. 3.19).



```
alsalemsaker@alsalemsaker-VirtualBox: ~/work/arch-pc/lab06
GNU nano 6.2 /home/alsalemsaker/work/arch-pc/lab06/variant.asm
include 'in_out.asm'
SECTION .data
msg: DB 'Введите № студенческого билета: ',0
rem: DB 'Ваш вариант: ',0

SECTION .bss
x: RESB 80

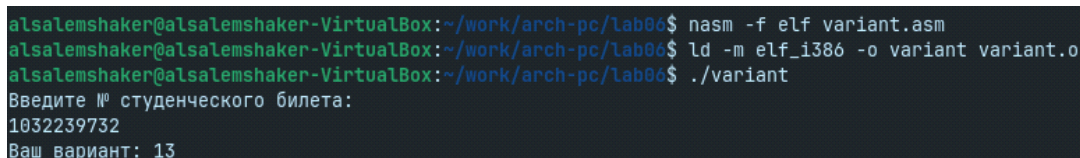
SECTION .text
GLOBAL _start
_start:

mov eax, msg
call sprintf
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 iprintf

call quit
```

Рис. 3.19: Заполняем файл

Компилируем файл и запускаем его (рис. 3.20).



```
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ nasm -f elf variant.asm
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1032239732
Ваш вариант: 13
```

Рис. 3.20: Проверяем результат работы программы

- **Ответы на вопросы по программе**

- Строка “mov eax,rem” и строка “call sprint” отвечают за вывод на экран сообщения ‘Ваш вариант:’.

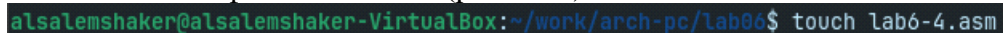
- Эти инструкции используются для чтения строки с вводом данных от пользователя. Начальный адрес строки сохраняется в регистре `ecx`, а количество символов в строке (максимальное количество символов, которое может

быть считано) сохраняется в регистре `edx`. Затем вызывается процедура `sread`, которая выполняет чтение строки.

- Инструкция `call atoi` используется для преобразования строки в целое число. Она принимает адрес строки в регистре `eax` и возвращает полученное число в регистре `eax`.
- 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 `eax` на значение регистра `ebx` с сохранением частного в регистре `eax` и остатка в регистре `edx`.
- Остаток от деления записывается в регистр `edx`.
- Инструкция `inc edx` используется для увеличения значения в регистре `edx` на 1. В данном случае, она увеличивает остаток от деления на 1.
- Строка `mov eax,edx` передает значение остатка от деления в регистр `eax`. Строка `call iprintLF` вызывает процедуру `iprintLF` для вывода значения на экран вместе с переводом строки.

• Задание для самостоятельной работы

Создаем новый файл в каталоге (рис. 3.21).

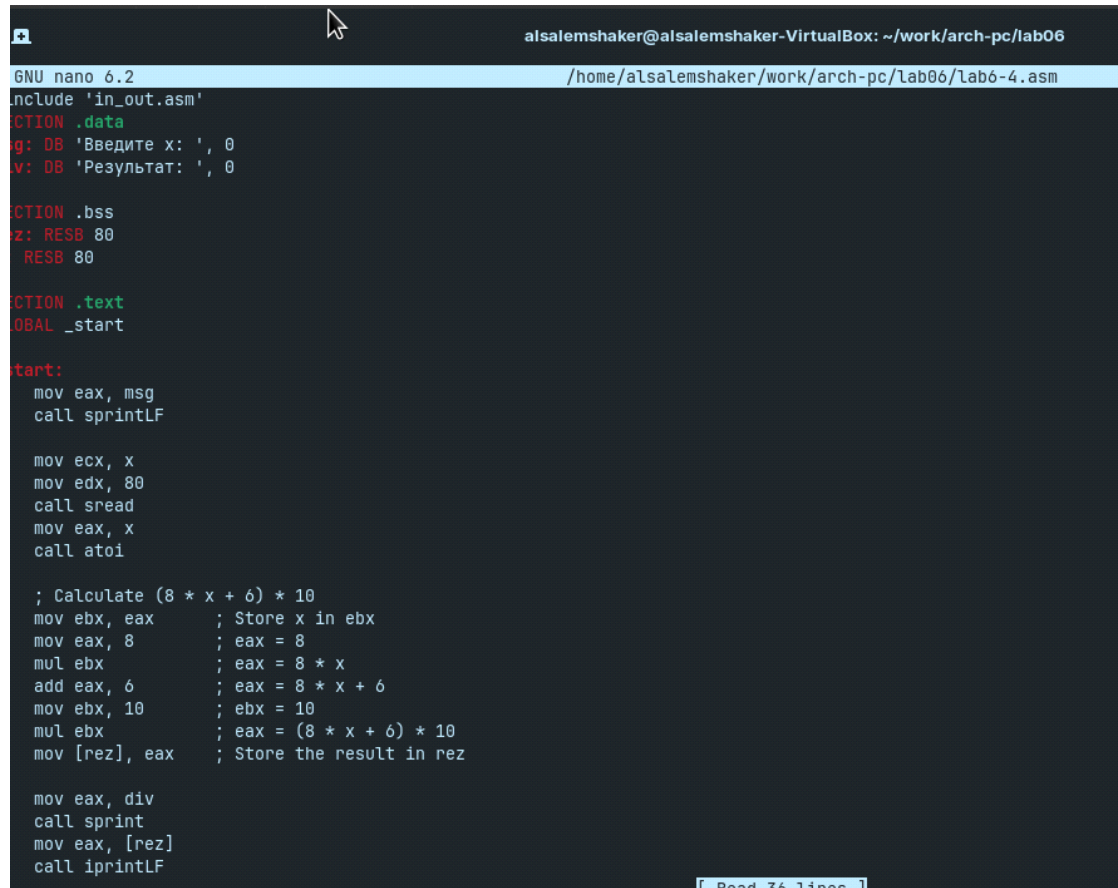


```
alsalemsaker@alsalemsaker-VirtualBox:~/work/arch-pc/lab06$ touch lab6-4.asm
```

Рис. 3.21: Создаем файл

Открываем его и заполняем, чтобы решалось выражение $f(x)=(8*x+6)*10$ (рис.

3.22)



```
GNU nano 6.2 /home/alsalemshaker/work/arch-pc/lab06/lab6-4.asm
include 'in_out.asm'
SECTION .data
msg: DB 'Введите x: ', 0
v: DB 'Результат: ', 0

SECTION .bss
rez: RESB 80
      RESB 80

SECTION .text
GLOBAL _start

_start:
    mov eax, msg
    call sprintf

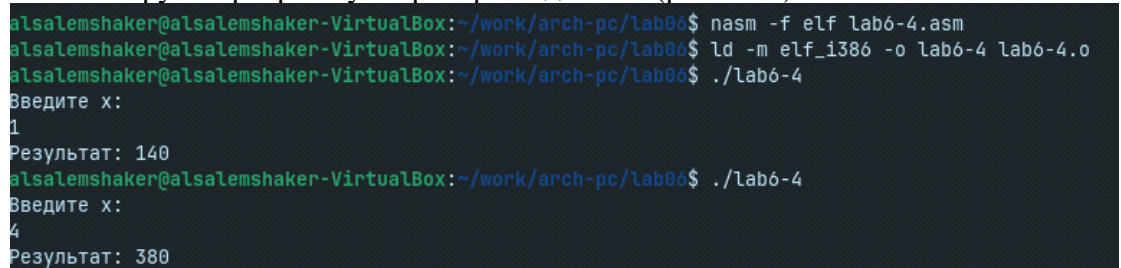
    mov ecx, x
    mov edx, 80
    call sread
    mov eax, x
    call atoi

    ; Calculate (8 * x + 6) * 10
    mov ebx, eax      ; Store x in ebx
    mov eax, 8         ; eax = 8
    mul ebx           ; eax = 8 * x
    add eax, 6         ; eax = 8 * x + 6
    mov ebx, 10        ; ebx = 10
    mul ebx           ; eax = (8 * x + 6) * 10
    mov [rez], eax     ; Store the result in rez

    mov eax, div
    call sprintf
    mov eax, [rez]
    call iprintf
```

Рис. 3.22: Заполняем файл

Компилируем программу и проверяем для $x=1$ (рис. 3.23).



```
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
1
Результат: 140
alsalemshaker@alsalemshaker-VirtualBox:~/work/arch-pc/lab06$ ./lab6-4
Введите x:
4
Результат: 380
```

Рис. 3.24: Проверяем работу программы

- **Выводы**

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