

Отчёт по лабораторной работе №6

Дисциплина: архитектура компьютера

Чернятьева Олеся Олеговна

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
2.1	Символьные и численные данные в NASM	5
2.2	Выполнение арифметических операций в NASM	10
2.2.1	Ответы на вопросы по программе	13
2.3	Выполнение заданий для самостоятельной работы	14
3	Выводы	17

Список иллюстраций

2.1	Создание директории	5
2.2	Создание и копирование файла для дальнейшей работы	5
2.3	Редактирование файла	6
2.4	Редактирование файла	6
2.5	Запуск исполняемого файла	6
2.6	Редактирование файла	7
2.7	Запуск исполняемого файла	7
2.8	Создание файла	7
2.9	Редактирование файла	8
2.10	Запуск исполняемого файла	8
2.11	Редактирование файла	8
2.12	Запуск исполняемого файла	9
2.13	Редактирование файла	9
2.14	Запуск исполняемого файла	10
2.15	Создание файла	10
2.16	Редактирование файла	10
2.17	Запуск исполняемого файла	11
2.18	Изменение программы	11
2.19	Запуск исполняемого файла	11
2.20	Создание файла	12
2.21	Редактирование файла	12
2.22	Запуск исполняемого файла	12
2.23	Создание файла	14
2.24	Написание программы	14
2.25	Запуск исполняемого файла	14
2.26	Запуск исполняемого файла	15

1 Цель работы

Цель данной лабораторной работы - освоение арифметических инструкций низкоуровневого языка ассемблера NASM.

2 Выполнение лабораторной работы

2.1 Символьные и численные данные в NASM

1

С помощью `mkdir` создала директорию `lab06`, перехожу в нее и создаю файл для работы. (рис. [2.1])

```
[oochernyatjeva@fedora ~]$ mkdir -p ~/work/arch-pc/lab06
[oochernyatjeva@fedora ~]$ cd ~/work/arch-pc/lab06
[oochernyatjeva@fedora lab06]$
```

Рис. 2.1: Создание директории

2

копирую файл `in_out.asm` в новый созданный каталог, т.к. он будет использоваться в других программах (рис. [2.2]).

.

```
oochernyatjeva@fedora:~/Загрузки
[oochernyatjeva@fedora Загрузки]$ cp in_out.asm ~/work/arch-pc/lab06
[oochernyatjeva@fedora Загрузки]$
```

Рис. 2.2: Создание и копирование файла для дальнейшей работы

3

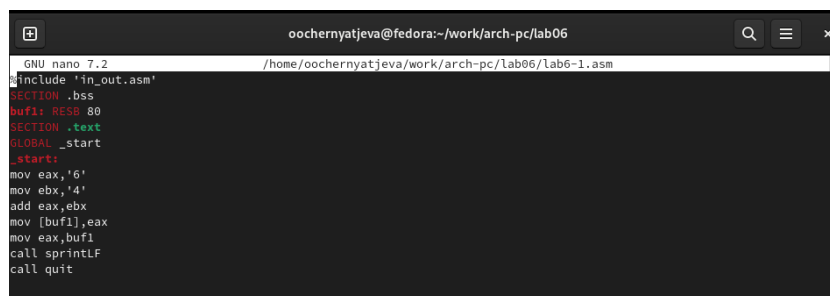
Создаю файл lab6-1.asm и вставляю в него программу для вывода значения записанные в регистр eax (рис. [2.3]).

```
[oochernyatjeva@fedora lab06]$ touch lab6-1.asm  
[oochernyatjeva@fedora lab06]$ mc
```

Рис. 2.3: Редактирование файла

4

Открываю созданный файл lab6-1.asm, вставляю в него программу (рис. [2.4]).

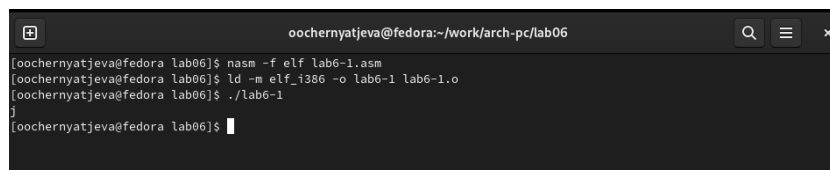


```
GNU nano 7.2 /home/oochernyatjeva/work/arch-pc/lab06/lab6-1.asm  
#include "in_out.asm"  
SECTION .bss  
buf1: RESB 80  
SECTION .text  
GLOBAL _start  
_start:  
mov eax,'6'  
mov ebx,'4'  
add eax,ebx  
mov [buf1],eax  
mov eax,buf1  
call sprintf  
call quit
```

Рис. 2.4: Редактирование файла

5

Создаю исполняемый файл программы и запускаю его (рис. [2.5]). Вывод программы: символ j, потому что программа вывела символ, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6.



```
[oochernyatjeva@fedora lab06]$ nasm -f elf lab6-1.asm  
[oochernyatjeva@fedora lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o  
[oochernyatjeva@fedora lab06]$ ./lab6-1  
j  
[oochernyatjeva@fedora lab06]$
```

Рис. 2.5: Запуск исполняемого файла

6

Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4 (рис. [2.6]).

```
GNU nano 7.2
#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 sprintLF
call quit
```

Рис. 2.6: Редактирование файла

7

Создаю новый исполняемый файл программы и запускаю его (рис. [2.7]). Теперь вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран.

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

Рис. 2.7: Запуск исполняемого файла

8

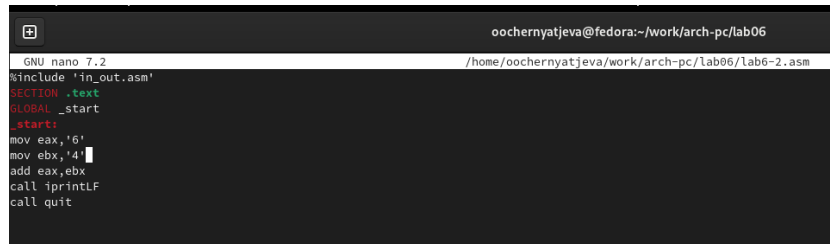
Создаю новый файл lab6-2.asm с помощью утилиты touch (рис. [2.8]).

```
[oochernyatjeva@fedora lab06]$ touch lab6-2.asm
[oochernyatjeva@fedora lab06]$ mc
```

Рис. 2.8: Создание файла

9

Ввожу в файл текст другой программы для вывода значения регистра `eax` (рис. [2.9]).

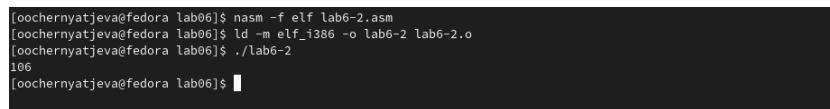


```
GNU nano 7.2 /home/oochernyatjeva/work/arch-pc/lab06/lab6-2.asm
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
add eax, ebx
call printf
call quit
```

Рис. 2.9: Редактирование файла

10

Создаю и запускаю исполняемый файл `lab6-2` (рис. [2.10]). Теперь вывод число 106, потому что программа позволяет вывести именно число, а не символ, хотя все еще происходит именно сложение кодов символов “6” и “4”.

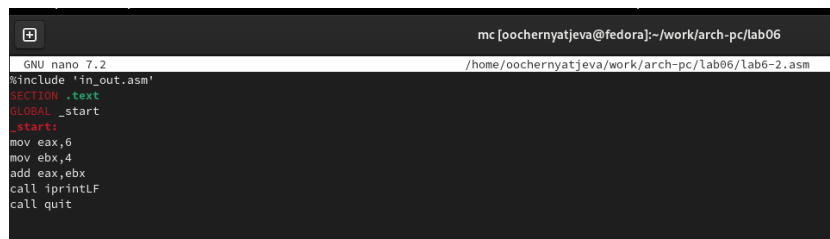


```
[oochernyatjeva@fedora lab06]$ nasm -f elf lab6-2.asm
[oochernyatjeva@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[oochernyatjeva@fedora lab06]$ ./lab6-2
106
[oochernyatjeva@fedora lab06]$
```

Рис. 2.10: Запуск исполняемого файла

11

Заменяю в тексте программы в файле `lab6-2.asm` символы “6” и “4” на числа 6 и 4 (рис. [2.11]).



```
GNU nano 7.2 /home/oochernyatjeva/work/arch-pc/lab06/lab6-2.asm
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, 6
mov ebx, 4
add eax, ebx
call printf
call quit
```

Рис. 2.11: Редактирование файла

12

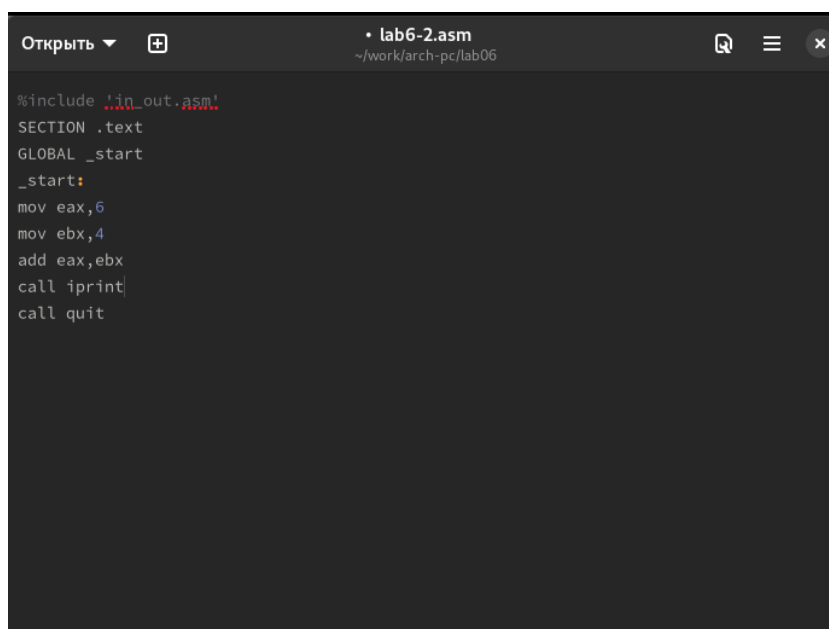
Создаю и запускаю новый исполняемый файл (рис. [2.12]).. Теперь программа складывает не соответствующие символам коды в системе ASCII, а сами числа, поэтому вывод 10.

```
[oochernyatjeva@fedora lab06]$ nasm -f elf lab6-2.asm
[oochernyatjeva@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[oochernyatjeva@fedora lab06]$ ./lab6-2
10
[oochernyatjeva@fedora lab06]$
```

Рис. 2.12: Запуск исполняемого файла

13

Заменяю в тексте программы функцию `iprintLF` на `iprint` (рис. [2.13]).



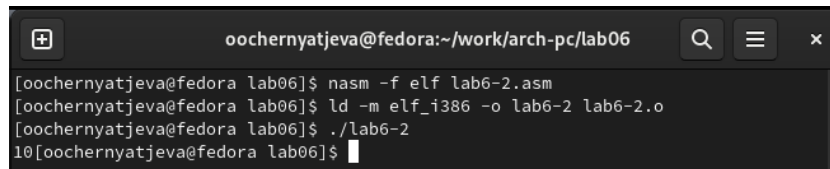
```
Открыть ▾ + lab6-2.asm
~/work/arch-pc/lab06

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

Рис. 2.13: Редактирование файла

14

Создаю и запускаю новый исполняемый файл (рис. [2.14]). Вывод не изменился, потому что символ переноса строки не отображался, когда программа исполнялась с функцией `iprintLF`, а `iprint` не добавляет к выводу символ переноса строки, в отличие от `iprintLF`.



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

Рис. 2.14: Запуск исполняемого файла

2.2 Выполнение арифметических операций в NASM

15

Создаю файл lab6-3.asm с помощью утилиты touch (рис. [2.15]).

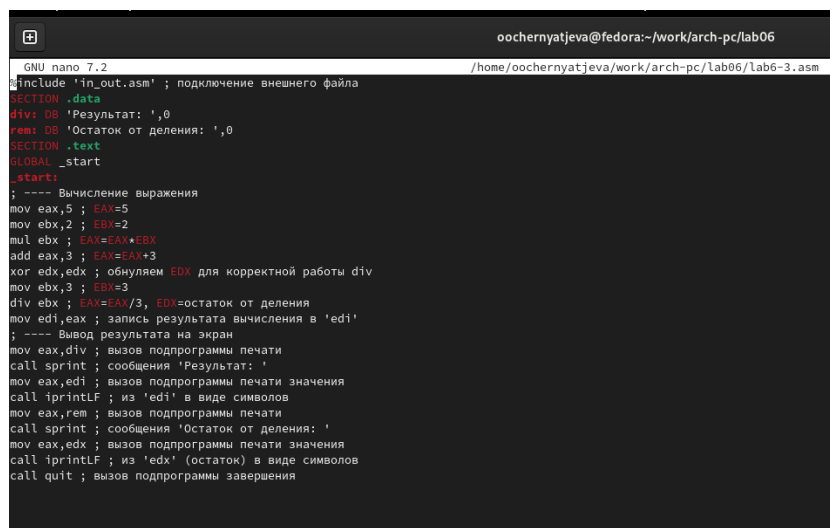


```
10
[oochernyatjeva@fedora lab06]$ touch lab6-3.asm
[oochernyatjeva@fedora lab06]$ mc
```

Рис. 2.15: Создание файла

16

Ввожу в созданный файл текст программы для вычисления значения выражения $f(x) = (5 * 2 + 3)/3$ (рис. [2.16]).



```
GNU nano 7.2 /home/oochernyatjeva/work/arch-pc/lab06/lab6-3.asm
#include "in_out.asm" ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; --- Вычисление выражения
mov eax,5 ; EAX=5
mov ebx,2 ; EBX=2
mul ebx ; EAX=EBX*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, EDI=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; --- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call print ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintf ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call print ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintf ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 2.16: Редактирование файла

17

Создаю исполняемый файл и запускаю его (рис. [2.17]).

```
[oochernyatjeva@fedora lab06]$ nasm -f elf lab6-3.asm
[oochernyatjeva@fedora lab06]$ ld -m elf_i386 -o lab6-3 lab6-3.o
[oochernyatjeva@fedora lab06]$ ./lab6-3
Результат: 4
Остаток от деления: 1
[oochernyatjeva@fedora lab06]$
```

Рис. 2.17: Запуск исполняемого файла

18

Изменяю программу так, чтобы она вычисляла значение выражения $f(x) = (4 * 6 + 2)/5$ (рис. [2.18]).

```
; ---- Вычисление выражения
mov eax,4 ; EAX=4
mov ebx,6 ; EBX=6
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+2
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=5
div ebx ; EAX=EAX/5, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в '.edi'
```

Рис. 2.18: Изменение программы

19

Создаю и запускаю новый исполняемый файл (рис. [2.19]). Программа отработала верно.

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

Рис. 2.19: Запуск исполняемого файла

20

Создаю файл variant.asm с помощью утилиты touch (рис. [2.20]).

```
[oochernyatjeva@fedora lab06]$ touch variant.asm
[oochernyatjeva@fedora lab06]$ mc
```

Рис. 2.20: Создание файла

21

Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета (рис. [2.21]).

```
GNU nano 7.2 /home/oochernyatjeva/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
xi: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
mov ecx, x
mov edx, 80
call sread
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax=x'
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprintf
mov eax, edx
call iprintf
call quit
```

Рис. 2.21: Редактирование файла

22

Создаю и запускаю исполняемый файл (рис. [2.22]). Ввожу номер своего студ. билета “1132239115” с клавиатуры, программа вывела, что мой вариант - 2.

```
[oochernyatjeva@fedora lab06]$ nasm -f elf variant.asm
[oochernyatjeva@fedora lab06]$ ld -m elf_i386 -o variant variant.o
[oochernyatjeva@fedora lab06]$ ./variant
Введите № студенческого билета:
1132239115
Ваш вариант: 16
[oochernyatjeva@fedora lab06]$
```

Рис. 2.22: Запуск исполняемого файла

2.2.1 Ответы на вопросы по программе

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

```
mov eax,rem  
call sprint
```

2. Инструкция `mov ecx, x` используется, чтобы положить адрес вводимой строки `x` в регистр `ecx` `mov edx, 80` - запись в регистр `edx` длины вводимой строки `call sread` - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры
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
```

2.3 Выполнение заданий для самостоятельной работы

1

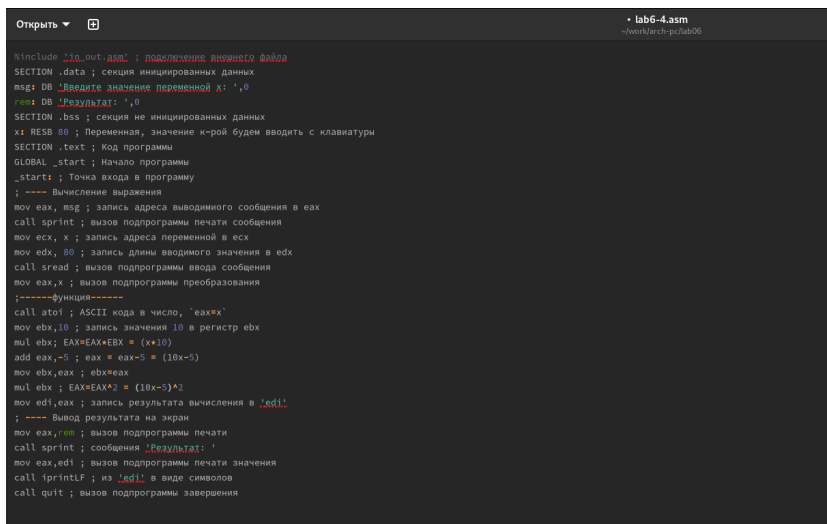
Создаю файл lab6-4.asm с помощью утилиты touch (рис. [2.23]).

```
[oochernyatjeva@fedora lab06]$ touch lab6-4.asm
[oochernyatjeva@fedora lab06]$
```

Рис. 2.23: Создание файла

2

Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения 2 варианта: $f(x) = (10x-5)^2$ (рис. [2.24]).



```
Открыть ▾
lab6-4.asm
~/work/arch-pc/lab06

#include "in_out.asm" ; подключение внешнего файла
SECTION .data ; секция инициализированных данных
msg: DB "Введите значение переменной x: ",0
res: DB "Результат: ",0
SECTION .bss ; секция не инициализированных данных
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start ; Точка входа в программу
; ---- Вычисление выражения
mov eax, msg ; запись адреса выводимого сообщения в eax
call sprint ; вызов подпрограммы печати сообщения
mov ecx, x ; запись адреса переменной в ecx
mov edx, 80 ; запись длины вводимого значения в edx
call sread ; вызов подпрограммы ввода сообщения
mov eax, x ; вызов подпрограммы преобразования
; ---- функция ----
call atoi ; ASCII код в число, 'eax'
mov ebx, 10 ; запись значения 10 в регистр ebx
mul ebx, EAX*EBX = (x*10)
add eax, -5 ; eax = eax-5 = (10x-5)
mov ebx, eax ; ebx=eax
mul ebx ; EAX=EAX*2 = (10x-5)^2
mov edi, eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax, res ; вызов подпрограммы печати
call sprint ; сообщения "Результат: "
mov eax, edi ; вызов подпрограммы печати значения
call printf ; в 'edi' в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 2.24: Написание программы

3

Создаю и запускаю исполняемый файл при вводе двух значений (рис. [2.25]). $x = 3$, $f(x) = 625$ Программа отработала верно.

```
[oochernyatjeva@fedora lab06]$ nasm -f elf lab6-4.asm
[oochernyatjeva@fedora lab06]$ ld -m elf_i386 -o lab6-4 lab6-4.o
[oochernyatjeva@fedora lab06]$ ./lab6-4
Введите значение переменной x: 3
Результат: 625
[oochernyatjeva@fedora lab06]$
```

Рис. 2.25: Запуск исполняемого файла

Запускаю исполняемый файл еще раз и ввожу второе значение икса $x=1$ для проверки на правильное вычисление результата. (рис. [2.26]). Программа отработала верно.

```
[oochernyatjeva@fedora lab06]$ ./lab6-4
Введите значение переменной x: 1
Результат: 25
[oochernyatjeva@fedora lab06]$
```

Рис. 2.26: Запуск исполняемого файла

Текст программы для вычисления значения выражения $f(x)=(10x-5)^2$

```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; секция инициированных данных
msg: DB 'Введите значение переменной x: ',0
rem: DB 'Результат: ',0
SECTION .bss ; секция не инициированных данных
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
; ---- Вычисление выражения
mov eax, msg ; запись адреса выводимого сообщения в eax
call sprint ; вызов подпрограммы печати сообщения
mov ecx, x ; запись адреса переменной в ecx
mov edx, 80 ; запись длины вводимого значения в edx
call sread ; вызов подпрограммы ввода сообщения
mov eax,x ; вызов подпрограммы преобразования
;-----функция-----
call atoi ; ASCII кода в число, `eax=x`
mov ebx,12 ; запись значения 2 в регистр ebx
```

```

mul ebx; EAX=EAX*EBX = (x*12)
add eax,3 ; eax = eax+3 = (3+12x)
mov ebx,5 ; ebx =5
mul ebx ; EAX=EAX*EBX = (3+12x)*5
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
call quit ; вызов подпрограммы завершения

```


3 Выводы

При выполнении данной лабораторной работы я освоила арифметические инструкции языка ассемблер NASM.