

Отчёт по лабораторной работе №6

Дисциплина: Архитектура компьютера

Толстых Александра Андреевна

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Символьные и численные данные в NASM	6
3.2	Выполнение арифметических операций в NASM	10
3.3	Выполнение задания для самостоятельной работы	13
4	Выводы	15

Список иллюстраций

3.1	Создание каталога и файла	6
3.2	Написание программы	6
3.3	Копирование файла	7
3.4	Выполнение программы	7
3.5	Изменение программы	7
3.6	Выполнение программы	8
3.7	Создание файла и написание программы	8
3.8	Выполнение программы	8
3.9	Изменение программы	9
3.10	Выполнение программы	9
3.11	Изменение программы	10
3.12	Выполнение программы	10
3.13	Создание файла и написание программы	11
3.14	Выполнение программы	11
3.15	Изменение программы	11
3.16	Выполнение программы	12
3.17	Создание файла и написание программы	12
3.18	Выполнение программы	12
3.19	Создание файла и написание программы	14
3.20	Выполнение программы	14

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Задание

1. Символьные и численные данные в NASM.
2. Выполнение арифметических операций в NASM.
3. Выполнение задания для самостоятельной работы (16 вариантов).

3 Выполнение лабораторной работы

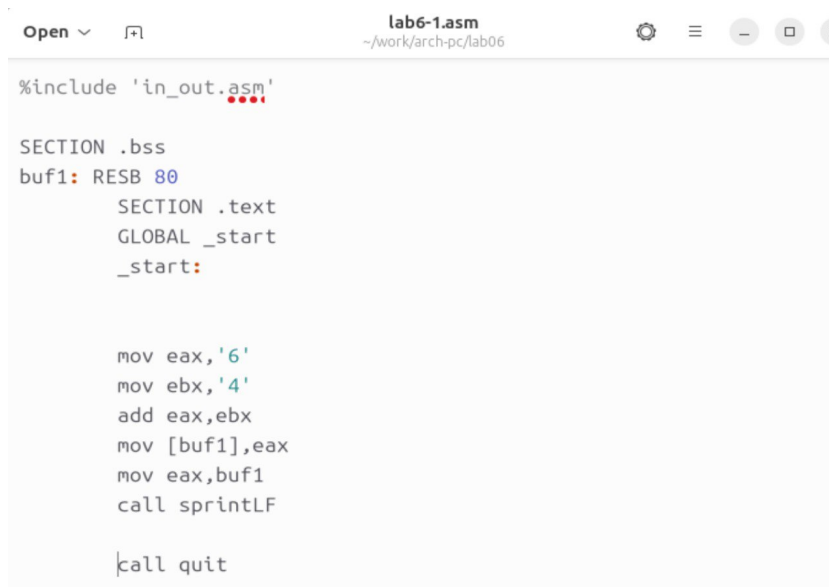
3.1 Символьные и численные данные в NASM

Создаю каталог для программ лабораторной №6. Перехожу в него и создаю файл lab6-1.asm (рис. 3.1).

```
aatolstikhkh@aatolstikhkh:~$ mkdir ~/work/arch-pc/lab06
aatolstikhkh@aatolstikhkh:~$ cd ~/work/arch-pc/lab06
aatolstikhkh@aatolstikhkh:~/work/arch-pc/lab06$ touch lab6-1.asm
aatolstikhkh@aatolstikhkh:~/work/arch-pc/lab06$
```

Рис. 3.1: Создание каталога и файла

Ввожу в созданный файл текст программы из листинга (рис. 3.2).



```
Open ▾  lab6-1.asm  ~/work/arch-pc/lab06

%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. 3.2: Написание программы

Копирую файл in_out.asm в каталог для программ лабораторной работы №6 (рис. 3.3).

```
aatolstihkh@aatolstihkh:~/work/arch-pc/lab06$ cp /home/aatolstihkh/Downloads/in_out.asm in_out.asm
aatolstihkh@aatolstihkh:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1.asm
aatolstihkh@aatolstihkh:~/work/arch-pc/lab06$
```

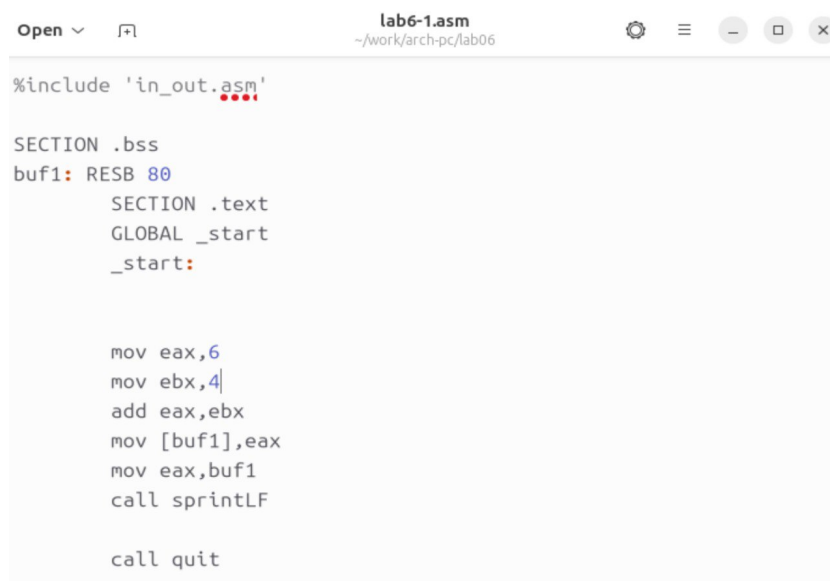
Рис. 3.3: Копирование файла

Создаю исполняемый файл и запускаю его (рис. 3.4).

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

Рис. 3.4: Выполнение программы

Изменяю текст программы, вместо символов записывая в eax, ebx числа (рис. 3.5).



```
Open  ▢  lab6-1.asm
~/work/arch-pc/lab06

#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. 3.5: Изменение программы

Создаю исполняемый файл и запускаю его (рис. 3.6).

```

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

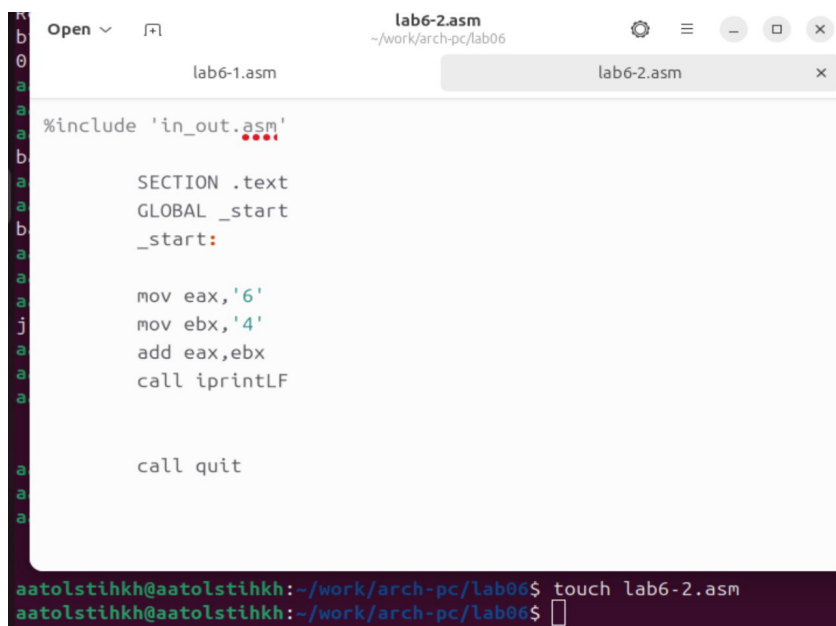
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$

```

Рис. 3.6: Выполнение программы

На экране ничего не отображается. Это связано с тем, что символ с кодом 10 - это символ перевода строки.

Создаю файл lab6-2.asm в каталоге для программ лабораторной №6. Ввожу в него текст программы из листинга 6.2 (рис. 3.7).



```

lab6-2.asm
~\work\arch-pc\lab06

lab6-1.asm
lab6-2.asm

%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

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

    call quit

aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ touch lab6-2.asm
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$

```

Рис. 3.7: Создание файла и написание программы

Создаю исполняемый файл и запускаю его (рис. 3.8).

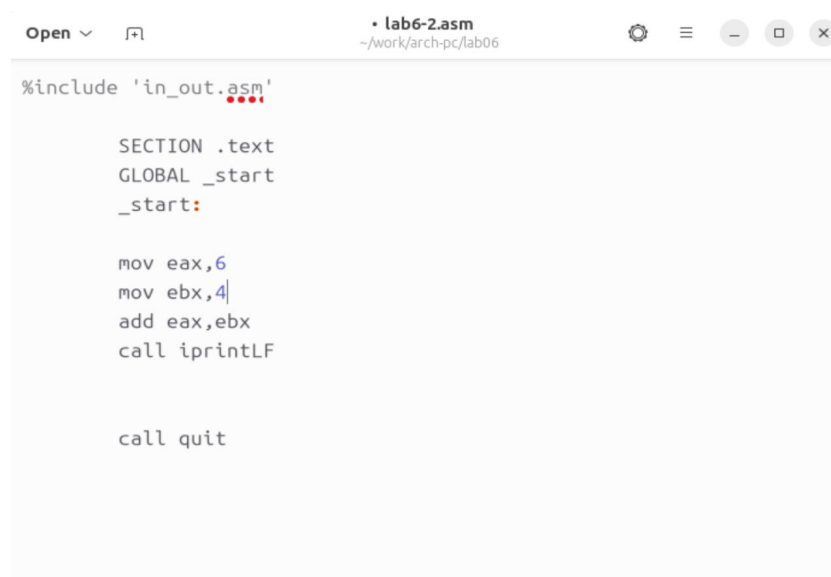
```

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

```

Рис. 3.8: Выполнение программы

Аналогично предыдущей программе заменяю символы на числа (рис. 3.9).



```
Open ▾  ↵  • lab6-2.asm  ~/work/arch-pc/lab06  ⚙  ≡  -  □  ×

#include 'in_out.asm'

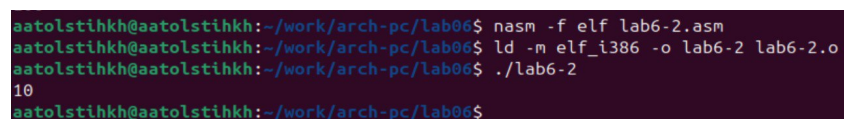
SECTION .text
GLOBAL _start
_start:

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

call quit
```

Рис. 3.9: Изменение программы

Создаю исполняемый файл и запускаю его (рис. 3.10).



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

Рис. 3.10: Выполнение программы

Теперь программа складывает не коды, соответствующие символам, а сами числа. Поэтому выводит число 10 - сумму чисел 4 и 6.

Заменяю функцию iprintLF на iprint (рис. 3.11).



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

%include 'in_out.asm'

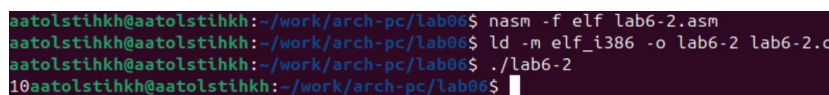
SECTION .text
GLOBAL _start
_start:

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

call quit
```

Рис. 3.11: Изменение программы

Создаю исполняемый файл и запускаю его (рис. 3.12).



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

Рис. 3.12: Выполнение программы

Вывод функции `iprintLF` от вывода функции `iprint` отличается тем, что в последнем случае после вывода не добавляется переход на новую строку.

3.2 Выполнение арифметических операций в NASM

С помощью утилиты `touch` создаю файл `lab6-3.asm`. Ввожу в него текст программы для вычисления значения указанного выражения (рис. 3.13).

```

; -----
; Программа вычисления выражения
; -----

%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=EAX*EBX

aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ touch lab6-3.asm
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$

```

Рис. 3.13: Создание файла и написание программы

Создаю исполняемый файл и запускаю его (рис. 3.14).

```

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

```

Рис. 3.14: Выполнение программы

Изменяю текст программы для вычисления нового выражения (рис. 3.15).

```

GLOBAL _start
_start:

; ---- 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'

; ---- Вывод результата на экран
mov eax,div                     ; вызов подпрограммы печати сообщения 'Результат: '
call sprint
mov eax,edi                     ; вызов подпрограммы печати значения из 'edi' в виде

```

Рис. 3.15: Изменение программы

Создаю исполняемый файл и запускаю его (рис. 3.16).

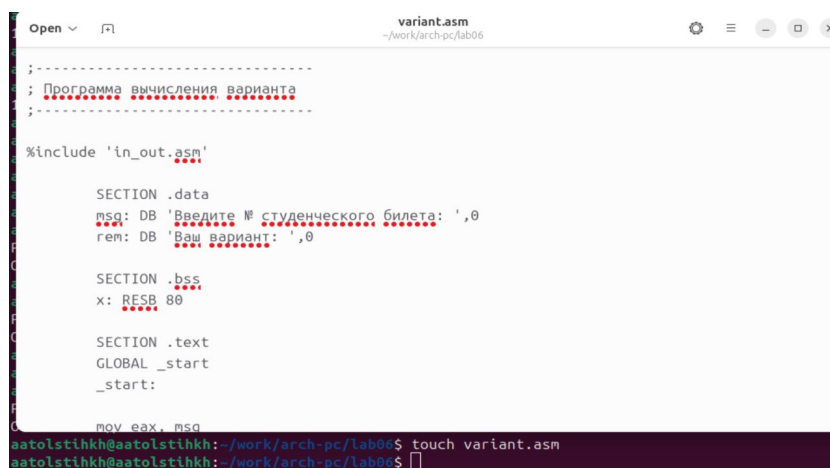
```

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

```

Рис. 3.16: Выполнение программы

С помощью утилиты `touch` создаю файл `variant.asm`. Ввожу в него текст программы для вычисления варианта (рис. 3.17).



```

;-----
; Программа вычисления варианта
;-----
%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

```

Рис. 3.17: Создание файла и написание программы

Создаю исполняемый файл и запускаю его (рис. 3.18).

```

aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ nasm -f elf variant.asm
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132246815
Ваш вариант: 16
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$

```

Рис. 3.18: Выполнение программы

Проверяю результат работы программы, вычисляя номер варианта аналитически. Также получаю число 16. Значит программа работает корректно.

Ответы на вопросы по программе: 1. За вывод сообщения “Ваш вариант” отвечают строки кода:

```

mov eax,rem
call sprint

```

2. Инструкция `mov esx, x` выполняется для того чтобы положить адрес вводимой строки `x` в регистр `esx`. Инструкция `mov edx, 80` выполняется для записи длины вводимой строки в регистр `edx`. Инструкция `call sread` выполняется для вызова подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры.
3. Инструкция `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 вычисление варианта отвечают строки:

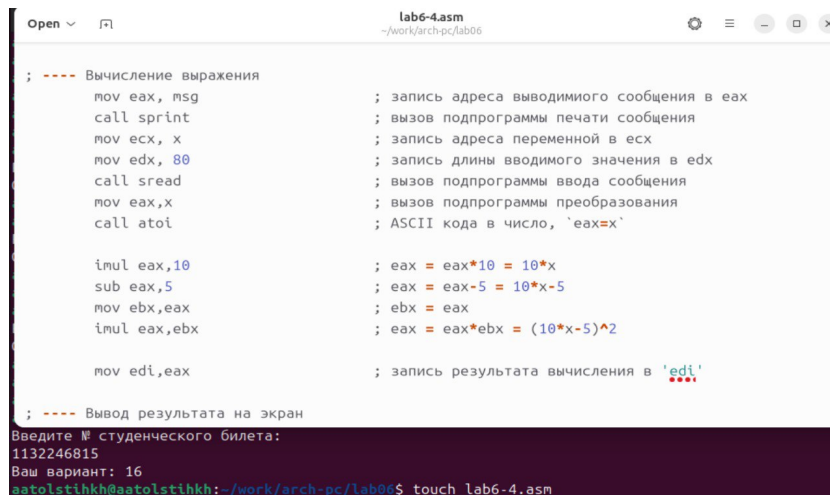
```
xor edx, edx
mov ebx, 20
div ebx
inc edx
```

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 результата вычислений отвечаю следующие строки:

```
mov eax,edx
call iprintLF
```

3.3 Выполнение задания для самостоятельной работы

Создаю файл `lab6-4.asm` и записываю в него программу для вычисления выражения 16 варианта. (рис. 3.19).



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

; ---- 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
call atoi                  ; ASCII кода в число, 'eax=x'

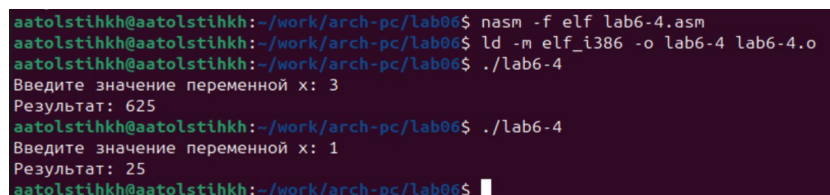
imul eax, 10                ; eax = eax*10 = 10*x
sub eax, 5                  ; eax = eax-5 = 10*x-5
mov ebx, eax                ; ebx = eax
imul eax, ebx               ; eax = eax*ebx = (10*x-5)^2

mov edi, eax                ; запись результата вычисления в 'edi'

; ---- Вывод результата на экран
Введите № студенческого билета:
1132246815
Ваш вариант: 16
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ touch lab6-4.asm
```

Рис. 3.19: Создание файла и написание программы

Создаю исполняемый файл и проверяю его работу для указанных значений (рис. 3.20).



```
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ ./lab6-4
Введите значение переменной x: 3
Результат: 625
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$ ./lab6-4
Введите значение переменной x: 1
Результат: 25
aamolstihkh@aamolstihkh:~/work/arch-pc/lab06$
```

Рис. 3.20: Выполнение программы

4 Выводы

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