

# **Лабораторная работа 6**

**Арифметические операции в NASM**

Елисейкина Надежда Михайловна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>8</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>12</b>
<b>5</b>	<b>Выводы</b>	<b>29</b>
	<b>Список литературы</b>	<b>30</b>

# Список иллюстраций

4.1	Создание каталога и файла для лабораторной работы . . . . .	12
4.2	Листинг 1 . . . . .	13
4.3	Результаты работы программы из листинга 1 . . . . .	14
4.4	Листинг 1 с изменениями . . . . .	15
4.5	Результаты работы программы после изменений в тексте программы	15
4.6	Листинг 2 . . . . .	16
4.7	Результаты работы программы из листинга 2 . . . . .	16
4.8	Листинг 2 с внесенными изменениями . . . . .	17
4.9	Результаты работы программы после внесения изменений в текст программы . . . . .	18
4.10	Результаты замены функции <code>iprintLF</code> на <code>iprint</code> . . . . .	19
4.11	Листинг 3 . . . . .	20
4.12	Результаты работы программы из листинга 3 . . . . .	21
4.13	Листинг 3 с внесенными изменениями . . . . .	22
4.14	Результаты работы программы для вычисления выражения . . . .	23
4.15	Листинг 4 . . . . .	24
4.16	Результаты работы программы вычисления варианта задания по номеру студенческого билета . . . . .	25
4.17	Листинг программы для вычислений функции . . . . .	27
4.18	Результаты работы программы . . . . .	28

## **Список таблиц**

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

## 2 Задание

1. Создать каталог для программ лабораторной работы № 6, перейти в него и создать файл lab6-1.asm.
2. Ввести в файл lab6-1.asm текст программы из листинга 1 методического указания. Создать исполняемый файл и запустить его.
3. Изменить текст программы и вместо символов, записать в регистры числа. Исправить текст программы следующим образом: заменить строки `mov eax, '6'` `mov ebx, '4'` на строки `mov eax, 6` `mov ebx, 4` Создать исполняемый файл и запустить его.
4. Создать файл lab6-2.asm. Ввести в него текст из листинга 2 методического указания. Создать исполняемый файл и запустить его.
5. Аналогично первому примеру заменить символы на числа. Создать исполняемый файл и запустить его.
6. Заменить функцию `iprintLF` на `iprint`. Создать исполняемый файл и запустить его. Посмотреть отличие функций `iprintLF` и `iprint`.
7. Создать файл lab6-3.asm. Ввести в него текст из листинга 3 методического указания для вычисления арифметического выражения  $f(x) = (5 \cdot 2 + 3) / 3$ . Создать исполняемый файл и запустить его.
8. Изменить текст программы для вычисления выражения  $f(x) = (4 \cdot 6 + 2) / 5$ . Создать исполняемый файл и запустить его.
9. Создать файл `variant.asm`. Ввести в него текст из листинга 4 методического указания. Создать исполняемый файл и запустить его.
10. Ответить на вопросы методического указания.

### Задание для самостоятельной работы

1. 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 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создать исполняемый файл и проверить его работу для значений  $x_1$  и  $x_2$ .

## 3 Теоретическое введение

### Адресация в NASM

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. Далее рассмотрены все существующие способы задания адреса хранения операндов – способы адресации.

Существует три основных способа адресации: - Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: `mov ax,bx`.

- Непосредственная адресация – значение операнда задается непосредственно в команде, Например: `mov ax,2`.
- Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

Например, определим переменную `intg DD 3` – это означает, что задается область памяти размером 4 байта, адрес которой обозначен меткой `intg`. В таком случае, команда

```
mov eax,[intg]
```

копирует из памяти по адресу `intg` данные в регистр `eax`. В свою очередь команда

```
mov [intg],eax
```



запишет в память по адресу `intg` данные из регистра `eax`.

Также рассмотрим команду

```
mov eax,intg
```

В этом случае в регистр `eax` запишется адрес `intg`. Допустим, для `intg` выделена память, начиная с ячейки с адресом `0x600144`, тогда команда `mov eax,intg` аналогична команде `mov eax,0x600144` – т.е. эта команда запишет в регистр `eax` число `0x600144`.

Целочисленное сложение `add`.

Схема команды целочисленного сложения `add` (от англ. addition - добавление) выполняет сложение двух операндов и записывает результат по адресу первого операнда. Команда `add` работает как с числами со знаком, так и без знака и выглядит следующим образом:

```
Add операнд_1, операнд_2
```

Допустимые сочетания операндов для команды `add` аналогичны сочетаниям операндов для команды `mov`.

Целочисленное вычитание `sub`.

Команда целочисленного вычитания `sub` (от англ. subtraction – вычитание) работает аналогично команде `add` и выглядит следующим образом:

```
sub операнд_1, операнд_2
```

Команды инкремента и декремента.

Довольно часто при написании программ встречается операция прибавления или вычитания единицы. Прибавление единицы называется инкрементом, а вычитание – декрементом. Для этих операций существуют специальные команды: `inc` (от англ. increment) и `dec` (от англ. decrement), которые увеличивают и уменьшают на 1 свой операнд.

```
inc операнд
```

```
dec операнд
```

Операндом может быть регистр или ячейка памяти любого размера. Команды инкремента и декремента выгодны тем, что они занимают меньше места, чем

соответствующие команды сложения и вычитания.

Команда изменения знака операнда `neg`.

Еще одна команда, которую можно отнести к арифметическим командам это команда изменения знака `neg`:

`neg` операнд

Команда `neg` рассматривает свой операнд как число со знаком и меняет знак операнда на противоположный. Операндом может быть регистр или ячейка памяти любого размера.

Команды умножения `mul` и `imul`.

Умножение и деление, в отличие от сложения и вычитания, для знаковых и беззнаковых чисел производиться по-разному, поэтому существуют различные команды. Для беззнакового умножения используется команда `mul` (от англ. `Multiply`- умножение):

`mul` операнд

Для знакового умножения используется команда `imul`:

`imul` операнд

Команды деления `div` и `idiv`.

Для деления, как и для умножения, существует 2 команды `div` (от англ. `divide` - деление) и `idiv`:

`div` делитель

`idiv` делитель

В командах указывается только один операнд – делитель, который может быть регистром или ячейкой памяти, но не может быть непосредственным операндом. Местоположение делимого и результата для команд деления зависит от размера делителя.

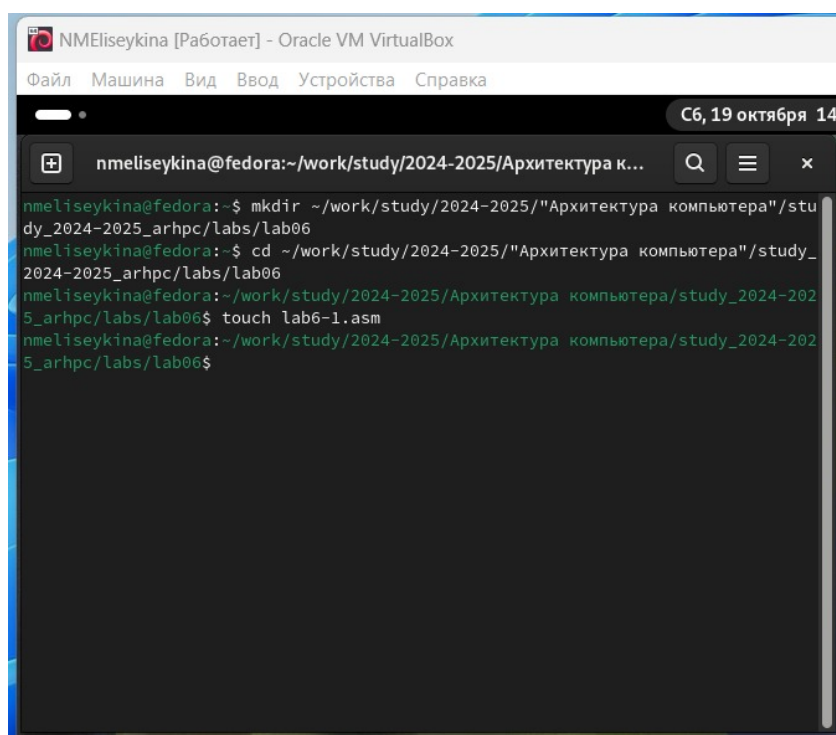
Перевод символа числа в десятичную символьную запись

Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от `American Standard Code for Information`

Interchange (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом. Расширенная таблица ASCII состоит из двух частей. Первая (символы с кодами 0-127) является универсальной (см. Приложение.), а вторая (коды 128-255) предназначена для специальных символов и букв национальных алфавитов и на компьютерах разных типов может меняться. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). По этому, например, чтобы вывести число, надо предварительно преобразовать его цифры в ASCII-коды этих цифр и выводить на экран эти коды, а не само число. Если же выводить число на экран непосредственно, то экран воспримет его не как число, а как последовательность ASCII-символов – каждый байт числа будет воспринят как один ASCII-символ – и выведет на экран эти символы. Аналогичная ситуация происходит и при вводе данных с клавиатуры. Введенные данные будут представлять собой символы, что сделает невозможным получение корректного результата при выполнении над ними арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно.

## 4 Выполнение лабораторной работы

1. Создали каталог для лабораторной работы № 6, создали файл lab6-1.asm (рис. 4.1 Создание каталога и файла для лабораторной работы).



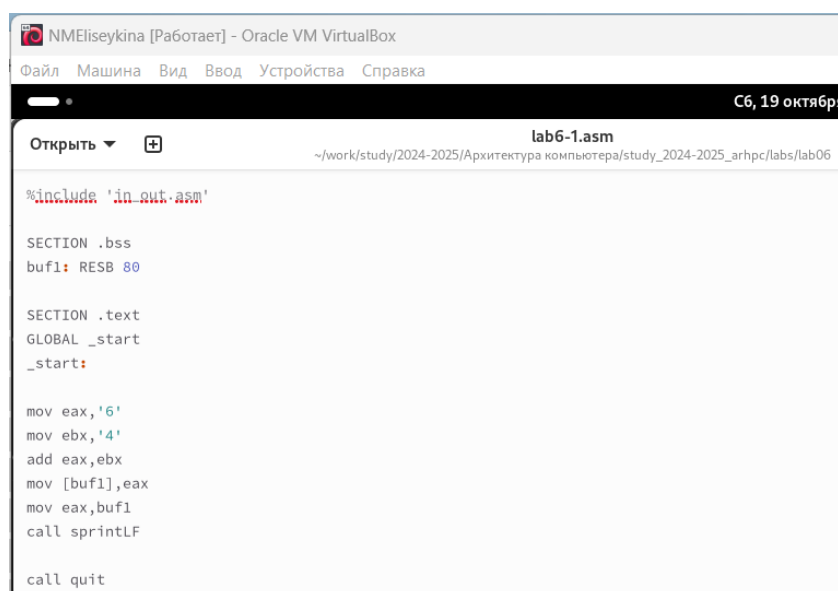
The screenshot shows a terminal window titled "nmeliseykina [Работаer] - Oracle VM VirtualBox". The terminal output is as follows:

```
nmeliseykina@fedora:~$ mkdir ~/work/study/2024-2025/"Архитектура компьютера"/study_2024-2025_arhpc/labs/lab06
nmeliseykina@fedora:~$ cd ~/work/study/2024-2025/"Архитектура компьютера"/study_2024-2025_arhpc/labs/lab06
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ touch lab6-1.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.1: Создание каталога и файла для лабораторной работы

2. Ввели в файл lab6-1.asm текст программы из листинга 1 (рис. 4.2 Листинг 1) методического указания. Создали исполняемый файл и запустили его. В данном случае при выводе значения регистра eax мы ожидаем увидеть число 10. Однако результатом был символ j. Это произошло потому, что

код символа 6 равен 00110110 в двоичном представлении (или 54 в десятичном представлении), а код символа 4 – 00110100 (52). Команда `add eax,ebx` запишет в регистр `eax` сумму кодов – 01101010 (106), что в свою очередь является кодом символа `j` (см. таблицу ASCII в приложении) (рис. 4.3 Результаты работы программы из листинга 1).



```
%include 'in_out.asm'

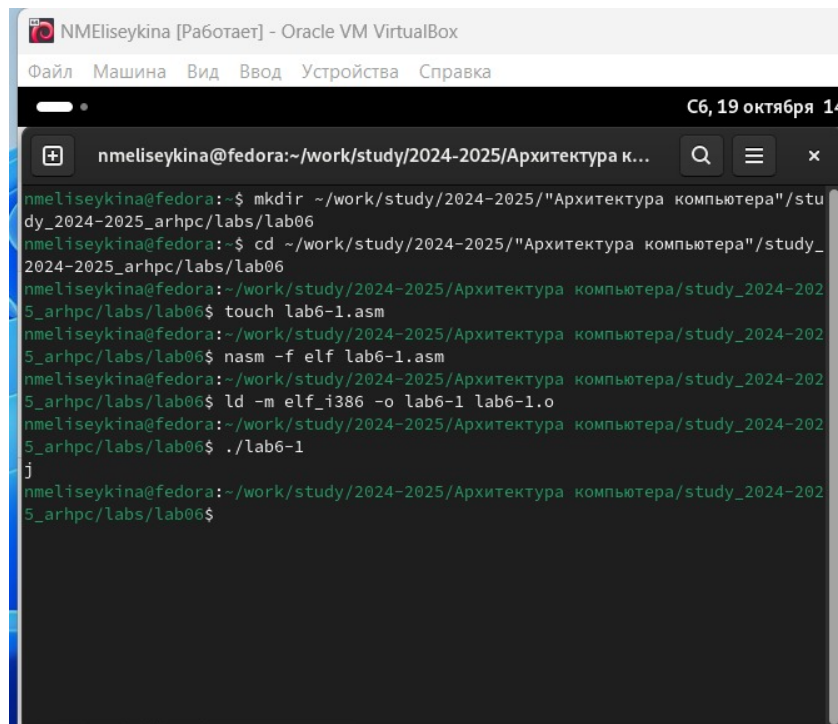
SECTION .bss
buf1: RESB 80

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. 4.2: Листинг 1



```
nmeiseykina@fedora:~$ mkdir ~/work/study/2024-2025/"Архитектура компьютера"/study_2024-2025_arhpc/labs/lab06
nmeiseykina@fedora:~$ cd ~/work/study/2024-2025/"Архитектура компьютера"/study_2024-2025_arhpc/labs/lab06
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ touch lab6-1.asm
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-1.asm
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-1
j
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.3: Результаты работы программы из листинга 1

3. Изменили текст программы (рис. 4.4 Листинг1 с внесенными изменениями).

Исправили текст программы: заменили строки

```
mov eax,'6'
```

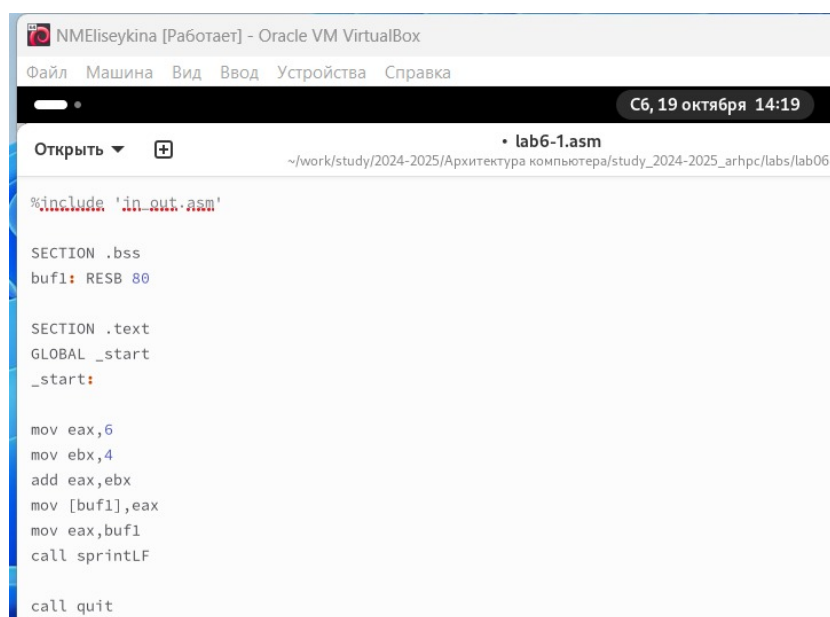
```
mov ebx,'4'
```

на строки

```
mov eax,6
```

```
mov ebx,4
```

Создали исполняемый файл и запустили его (рис. 4.5 Результаты работы программы после изменений в тексте программы).



The screenshot shows a code editor window titled "NMElseykina [Работает] - Oracle VM VirtualBox". The menu bar includes "Файл", "Машина", "Вид", "Ввод", "Устройства", and "Справка". The status bar at the top right shows "С6, 19 октября 14:19". The editor displays the file "lab6-1.asm" at the path "~/work/study/2024-2025/Архитектура компьютера/study\_2024-2025\_arhpc/labs/lab06". The code content is as follows:

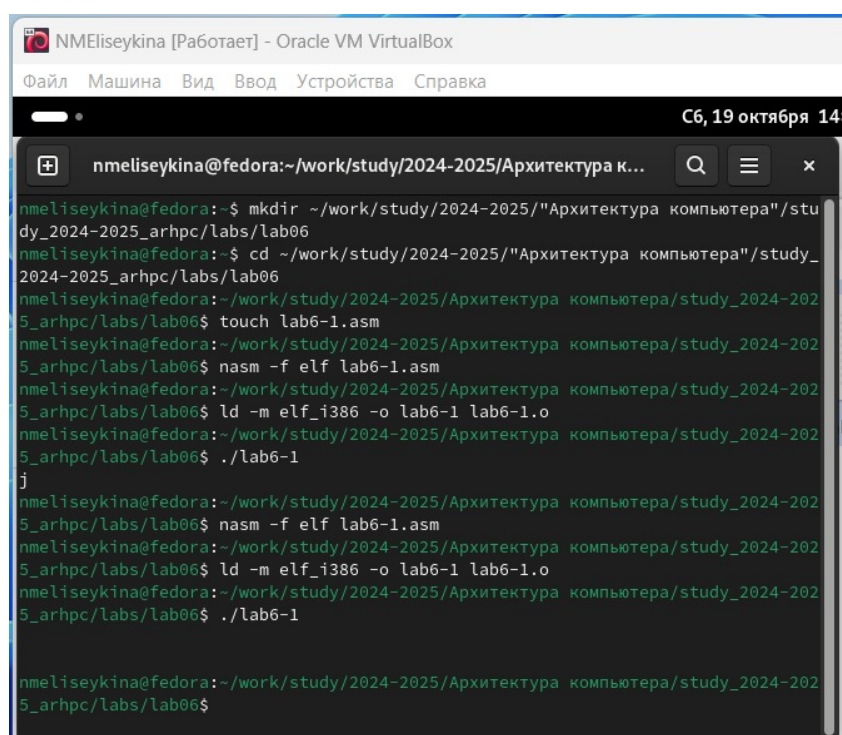
```
%include 'in_out.asm'

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. 4.4: Листинг 1 с изменениями



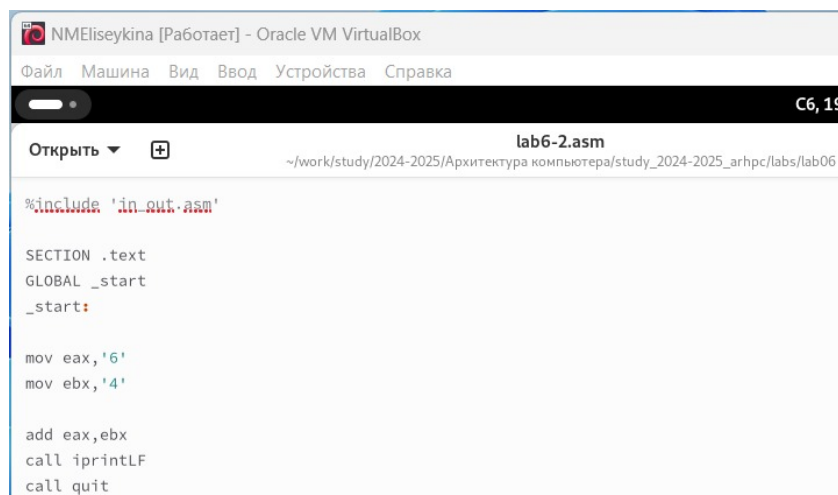
The screenshot shows a terminal window titled "NMElseykina [Работает] - Oracle VM VirtualBox". The menu bar includes "Файл", "Машина", "Вид", "Ввод", "Устройства", and "Справка". The status bar at the top right shows "С6, 19 октября 14:19". The terminal displays the following commands and output:

```
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура к...
nmeliseykina@fedora:~$ mkdir ~/work/study/2024-2025/"Архитектура компьютера"/study_2024-2025_arhpc/labs/lab06
nmeliseykina@fedora:~$ cd ~/work/study/2024-2025/"Архитектура компьютера"/study_2024-2025_arhpc/labs/lab06
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ touch lab6-1.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-1.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-1
j
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-1.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-1
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.5: Результаты работы программы после изменений в тексте программы

4. Создали файл lab6-2.asm. Ввели в него текст из листинга 2 методического

указания (рис. 4.6 Листинг 2). Создали исполняемый файл и запустили его (рис. 4.7 Результаты работы программы из листинга 2).



The screenshot shows a code editor window titled "NMEliseykina [Работает] - Oracle VM VirtualBox". The menu bar includes "Файл", "Машина", "Вид", "Ввод", "Устройства", and "Справка". The file name is "lab6-2.asm" and the path is "~/work/study/2024-2025/Архитектура компьютера/study\_2024-2025\_arhpc/labs/lab06". The code content is as follows:

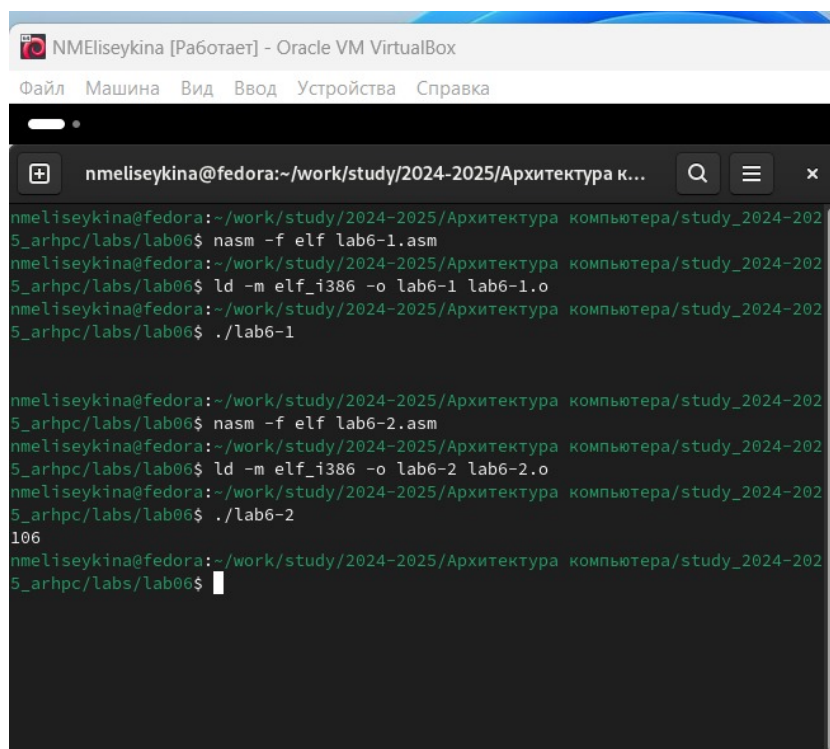
```
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax, '6'
mov ebx, '4'

add eax, ebx
call iprintLF
call quit
```

Рис. 4.6: Листинг 2



The screenshot shows a terminal window with the prompt "nmeliseykina@fedora:~/work/study/2024-2025/Архитектура к...". The terminal output is as follows:

```
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-1.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-1

nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-2.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-2
106
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.7: Результаты работы программы из листинга 2



5. Аналогично первому примеру заменили символы на числа (рис. 4.8 Листинг 2 с внесенными изменениями). Создали исполняемый файл и запустили его (рис. 4.9 Результаты работы программы после внесения изменений в текст программы (вместо символов, записали в регистры числа)).



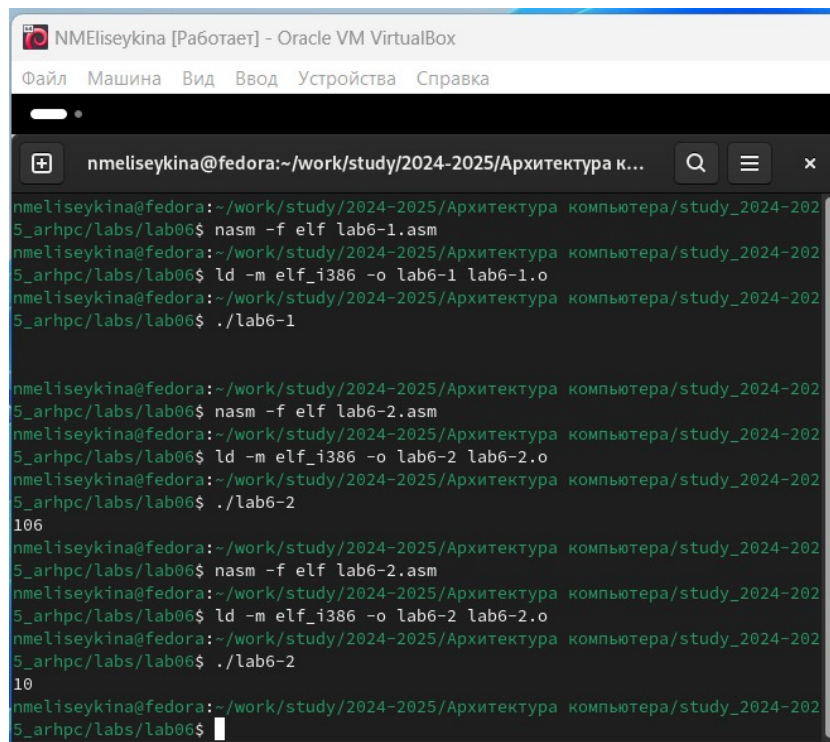
```
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax, 6
mov ebx, 4

add eax, ebx
call iprintLF
call quit
```

Рис. 4.8: Листинг 2 с внесенными изменениями



```
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-1.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-1

nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-2.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-2
106
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-2.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-2
10
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.9: Результаты работы программы после внесения изменений в текст программы

6. Заменяли функцию `iprintLF` на `iprint`. Создали исполняемый файл и запустили его. Посмотрели отличие функций `iprintLF` и `iprint` (рис. 4.10 Результаты замены функции `iprintLF` на `iprint`).

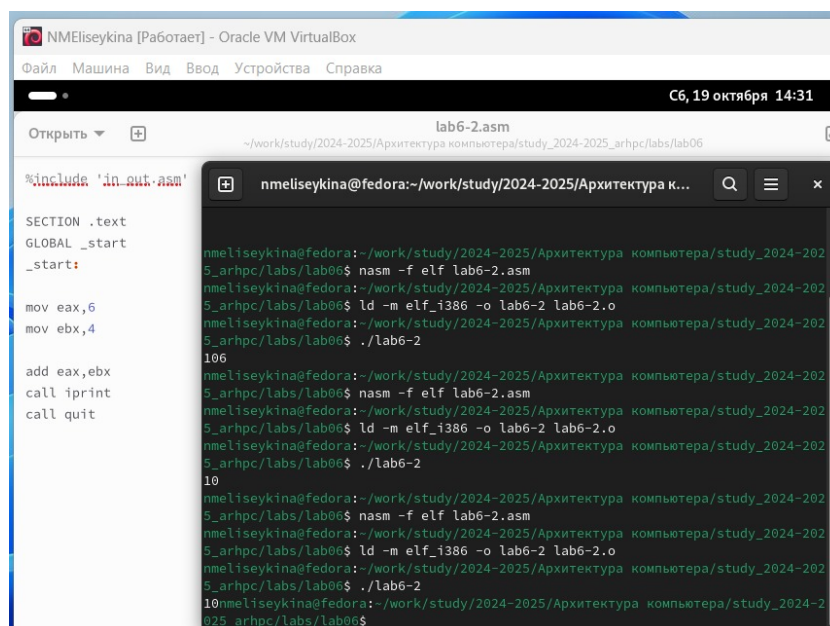


Рис. 4.10: Результаты замены функции `iprintLF` на `iprint`

7. Создали файл `lab6-3.asm`. Ввели в него текст из листинга 3 методического указания (рис. 4.11 Листинг 3) Создали исполняемый файл и запустили его (рис. 4.12 Результаты работы программы из листинга 3 методического указания).

```

%include "in-div.asm" ; подключение внешнего файла

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

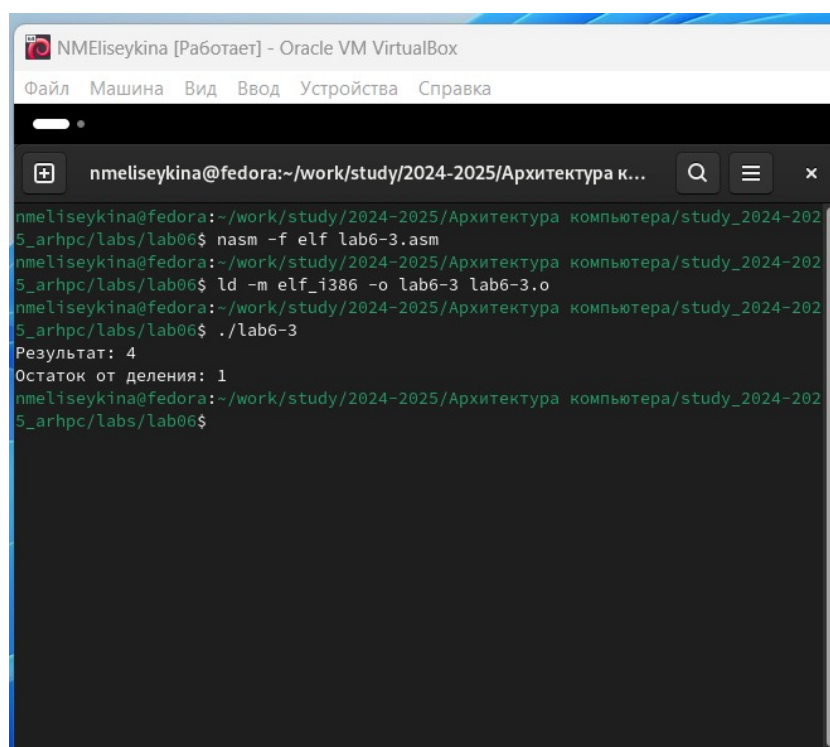
mov eax,div
call sprint
mov eax,edi
call iprintLF

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

call quit

```

Рис. 4.11: Листинг 3



```
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-3.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.12: Результаты работы программы из листинга 3

8. Изменили текст программы для вычисления выражения  $f(x)=(4*6+2)/5$  (рис. 4.13 Листинг 3 с внесенными изменениями). Создали исполняемый файл и запустили его (рис. 4.14 Результаты работы программы для 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:

mov eax,4
mov ebx,6
mul ebx

add eax,2
xor edx,edx
mov ebx,5
div ebx

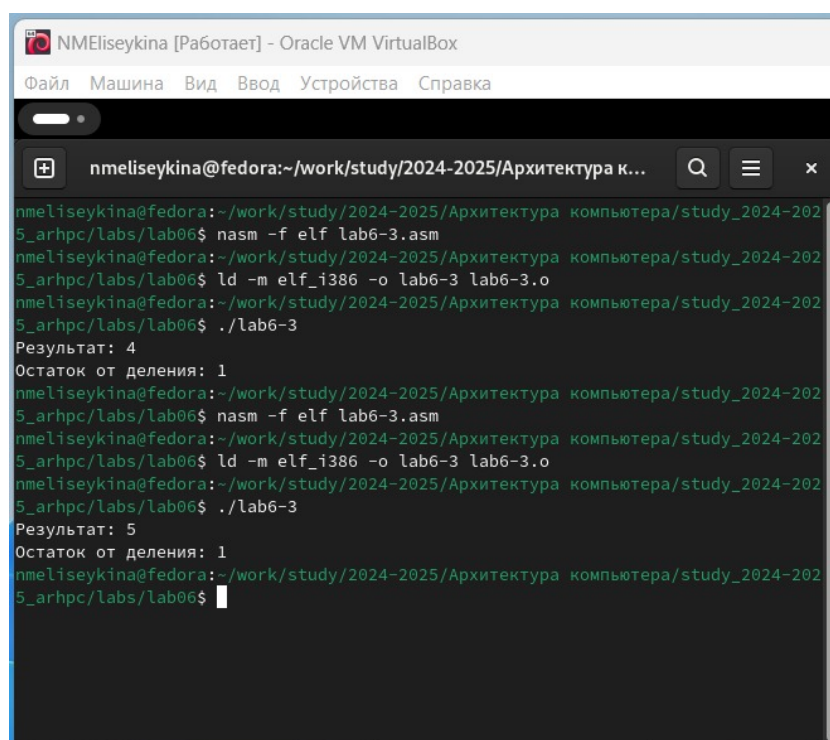
mov edi,eax

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

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

call quit
```

Рис. 4.13: Листинг 3 с внесенными изменениями



```
nmeliseykina [Работаer] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка

nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-3.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-3.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.14: Результаты работы программы для вычисления выражения

9.Создали файл variant.asm. Ввели в него текст из листинга 4 методического указания (рис. 4.15 Листинг 4). Создали исполняемый файл и запустили его (рис. 4.16 Результаты работы программы вычисления варианта задания по номеру студенческого билета).

```
#include 'in-out.asm'

SECTION .data
msg: DB 'Введите № студента: ',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

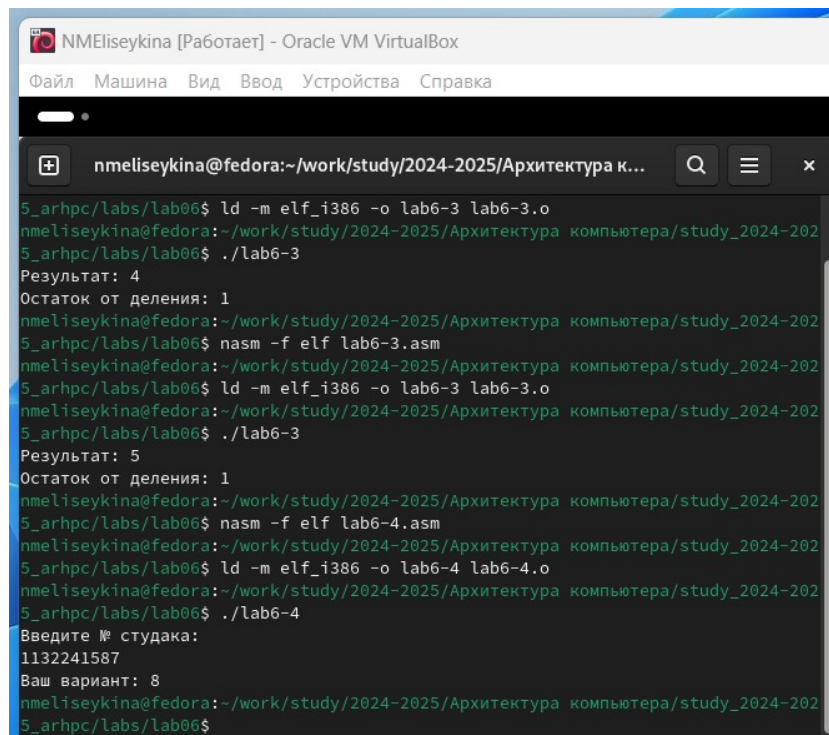
xor edx, edx
mov ebx, 20
div ebx
inc edx

mov eax, rem
call sprintf
mov eax, edx
call iprintLF

call quit
```

Рис. 4.15: Листинг 4





```
NMEliseykina [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка

nmeliseykina@fedora:~/work/study/2024-2025/Архитектура к...
5_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-3.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-4.asm
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-4
Введите № студака:
1132241587
Ваш вариант: 8
nmeliseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.16: Результаты работы программы вычисления варианта задания по номеру студенческого билета

Ответы на вопросы:

1. Какие строки листинга 4 отвечают за вывод на экран сообщения 'Ваш вариант:'?

Ответ: `mov eax,msg` `call sprintf`

2. Для чего используются следующие инструкции?

```
mov ecx, x
mov edx, 80
call sread
```

Ответ: Эти инструкции используются для ввода переменной X с клавиатуры и сохранения введенных данных.

3. Для чего используется инструкция "call atoi"?

Ответ: Эта инструкция используется для преобразования Кода переменной ASCII в число.

4. Какие строки листинга 4 отвечают за вычисления варианта?

Ответ: `mov ebx,20 div ebx inc edx`

5. В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”?

Ответ: В регистре ebx.

6. Для чего используется инструкция “inc edx”?

Ответ: Для увеличения значения edx на 1.

7. Какие строки листинга 4 отвечают за вывод на экран результата вычислений?

Ответ: `mov eax,edx call iprintLF`

Задание для самостоятельной работы

Написать программу вычисления выражения  $y = f(x)$ . Программа должна вывести 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 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создать исполняемый файл и проверить его работу для значений  $x_1$  и  $x_2$ . В результате выполнения программы вычисления варианта задания по номеру студенческого билета мне выдало вариант 11. Выбрала из таблицы вариантов заданий функцию  $f(x) = (11+x)*2-6$ ;  $x_1 = 1$ ,  $x_2 = 9$ . Листинг программы для вычислений записала в файл `sr.asm` (рис. 4.17 Листинг программы для вычислений функции)

```
SECTION .data
prim: DB '(11+x)*2-6',0
x1: DB 'Введите значение x: ',0
otv1: DB 'Ответ при x= ',0

SECTION .bss
p: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax,prim
call sprintf

mov eax,x1
call sprintf

mov ecx,p
mov edx,80
call sread

mov eax,p
call atoi

xor edx,edx

add eax,11
mov ebx,2
mul ebx
sub eax,6

mov edi,eax

mov eax,otv1
call sprintf
mov eax,p
call sprintf
mov eax,edi
call iprintfLF

call quit
```

Рис. 4.17: Листинг программы для вычислений функции

```
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ nasm -f elf lab6-5.asm
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ld -m elf_i386 -o lab6-5 lab6-5.o
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-5
(11+x)*2-6
Введите значение x: 1
Ответ при x= 1
18
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$ ./lab6-5
(11+x)*2-6
Введите значение x: 9
Ответ при x= 9
34
nmeiseykina@fedora:~/work/study/2024-2025/Архитектура компьютера/study_2024-2025_arhpc/labs/lab06$
```

Рис. 4.18: Результаты работы программы

## **5 Выводы**

Освоили арифметических инструкций языка ассемблера NASM.

## Список литературы

1. GDB: The GNU Project Debugger. — URL:<https://www.gnu.org/software/gdb/>.
2. GNU Bash Manual. — 2016. — URL: <https://www.gnu.org/software/bash/manual/>.
3. Midnight Commander Development Center. — 2021. — URL: <https://midnight-commander.org/>.
4. NASM Assembly Language Tutorials. — 2021. — URL: <https://asmtutor.com/>.
5. Newham C. Learning the bash Shell: Unix Shell Programming. — O'Reilly Media, 2005. — 354 с. — (In a Nutshell). — ISBN 0596009658. — URL: <http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658>.
6. Robbins A. Bash Pocket Reference. — O'Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
7. The NASM documentation. — 2021. — URL: <https://www.nasm.us/docs.php>.
8. Zarrelli G. Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
9. Колдаев В. Д., Лупин С. А. Архитектура ЭВМ. — М. : Форум, 2018.
10. Куляс О. Л., Никитин К. А. Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
11. Новожилов О. П. Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
12. Расширенный ассемблер: NASM. — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>.
13. Робачевский А., Немнюгин С., Стесик О. Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
14. Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: [http://www.stolyarov.info/books/asm\\_unix](http://www.stolyarov.info/books/asm_unix).

15. Таненбаум Э. Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
16. Таненбаум Э., Бос Х. Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science).