

Отчёт по лабораторной работе №6

НКАбд-03-25

Кулаженкова Яна Сергеевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Символьные и численные данные в NASM	9
4.2	Выполнение арифметических операций в NASM	13
4.3	Вопросы лабораторной работы	15
5	Задание для самостоятельной работы	17
6	Выводы	18

Список иллюстраций

4.1	Начало работы с файлом	9
4.2	Подготовка файла	10
4.3	Исполняемый файл	10
4.4	Изменим код	10
4.5	Запуск файла	11
4.6	Второй файл	11
4.7	Запуск второго файла	11
4.8	Новый код программы	12
4.9	Запуск файла	12
4.10	Новый код	12
4.11	Работа файла	12
4.12	Код программы	13
4.13	Выполнение программы	13
4.14	Исполнение файла	14
4.15	Редактирование файла	14
4.16	Выполнение файла	14
5.1	Самостоятельная работа	17

Список таблиц

3.1	Основные арифметические команды NASM {#tbl}	7
-----	---	---

1 Цель работы

Цель работа - освоение арифметических инструкций языка ассемблера NASM.

2 Задание

1. Изучить информацию об арифметических операциях в NASM
2. Выполнить лабораторную работу с использованием изученной информации
3. Выполнить задания для самостоятельной работы.

3 Теоретическое введение

В языке ассемблера NASM существуют три основных способа адресации операндов. 1. Регистровая адресация - операнды находятся в регистрах процессора (пример: `mov ax, bx`) 2. Непосредственная адресация - значение операнда задается непосредственно в команде (пример: `mov ax, 2`) 3. Адресация памяти - операнд указывает на адрес в памяти (пример: `mov eax, [intg]`)

Команда `mov eax, [intg]` загружает значение из памяти, а `mov eax, intg` загружает адрес переменной.

В [tbl] приведены основные арифметические команды в NASM.

Таблица 3.1: Основные арифметические команды NASM {#tbl}

Арифметическая операция	Команда
Сложение	<code>add <операнд1>, <операнд2></code>
Вычитание	<code>sub <операнд1>, <операнд2></code>
Инкремент/декремент	<code>inc <операнд>, dec <операнд></code>
Изменение знака	<code>neg <операнд></code>
Умножение	<code>mul (беззнаковое), imul (знаковое)</code>
Деление	<code>div (беззнаковое), idiv (знаковое)</code>

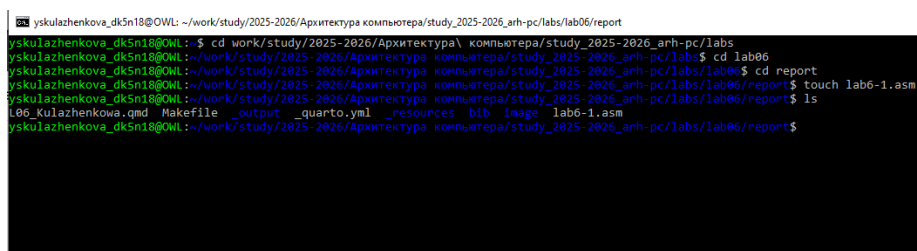
Операции умножения и деления имеют особенности: один операнд задается явно, второй находится в регистрах AL/AX/EAX, а результат помещается в определенные пары регистров в зависимости от размера операндов.

Ввод и вывод данных осуществляются в символьном виде с использованием кодировки ASCII. Для корректной работы с числами необходимо преобразование между символьным и числовым представлением с помощью специализированных функций.

4 Выполнение лабораторной работы

4.1 Символьные и численные данные в NASM

Создадим каталог для программ лабораторной работы № 6. Внутри него создадим файл lab6-1.asm (рис. 4.1).



```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report
yskulazhenkova_dk5n18@OWL:~$ cd work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs$ cd lab06
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06$ cd report
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ touch lab6-1.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ls
lab6_kulazhenkova.qmd  Makefile  _output  _quarto.yml  _resources  bib  image  lab6-1.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$
```

Рисунок 4.1: Начало работы с файлом

Рассмотрим примеры программ вывода символьных и численных значений. Для этого введем в наш файл следующий текст программы (рис. 4.2).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report
GNU nano 7.2 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
```

Рисунок 4.2: Подготовка файла

После создадим исполняемый файл и запустим его (рис. 4.3).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf lab6-1.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-1 lab6-1.o
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./lab6-1
6
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$
```

Рисунок 4.3: Исполняемый файл

Далее изменим текст программы и вместо символов, запишем в регистры числа (рис. 4.4).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report
GNU nano 7.2 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
```

Рисунок 4.4: Изменим код

Снова создадим исполняемый файл. Проверим работу файла (рис. 4.5).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ nano lab6-1.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ nasm -f elf lab6-1.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-1 lab6-1.o
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ ./lab6-1
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$
```

Рисунок 4.5: Запуск файла

Теперь преобразуем текст программы так, чтобы были использованы другие функции. Для этого создадим файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и введем в него новый текст программы (рис. 4.6).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report
GNU nano 7.2 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_
```

Рисунок 4.6: Второй файл

Продemonстрируем работу файла. Создадим исполняемый файл и запустим его (рис. 4.7).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ touch lab6-2.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ nasm -f elf lab6-2.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-2 lab6-2.o
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$ ./lab6-2
106
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arch-pc/labs/lab06/report$
```

Рисунок 4.7: Запуск второго файла

Аналогично предыдущему примеру заменим символы на числа (рис. 4.8).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report
GNU nano 7.2 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

```

Рисунок 4.8: Новый код программы

Создадим исполняемый файл и запустим его (рис. 4.9).

```
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ touch lab6-2.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano lab6-2.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf lab6-2.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-2 lab6-2.o
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./lab6-2
106
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano lab6-2.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf lab6-2.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-2 lab6-2.o
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./lab6-2
10
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ _

```

Рисунок 4.9: Запуск файла

Теперь заменим функцию `iprintLF` на `print` (рис. 4.10).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report
GNU nano 7.2 lab6-2.asm *
#include "in_out.asm"
SECTION .text
GLOBAL _start
_start:
mov eax, 6
mov ebx, 4
add eax, ebx
call iprint_
call quit

```

Рисунок 4.10: Новый код

Создадим исполняемый файл и запустим его (рис. 4.11).

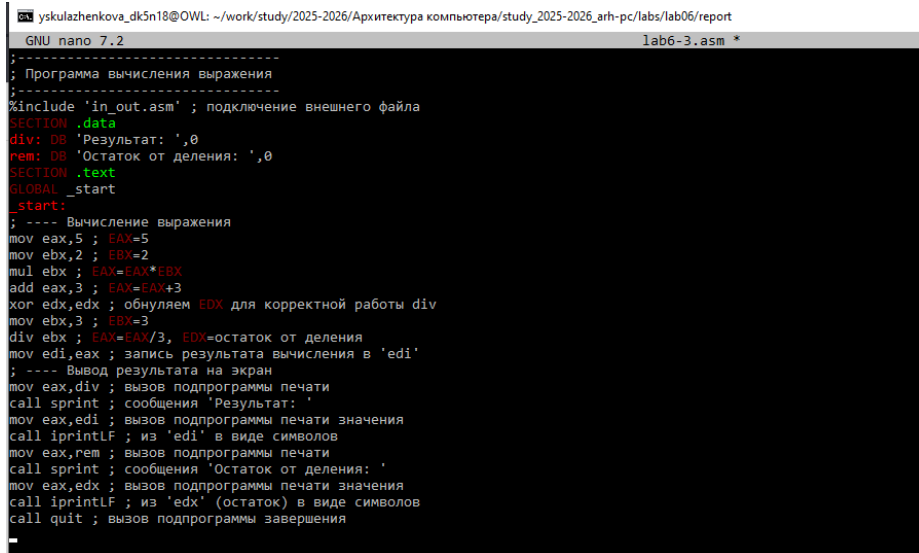
```
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano lab6-2.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf lab6-2.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-2 lab6-2.o
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./lab6-2
10
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ _

```

Рисунок 4.11: Работа файла

4.2 Выполнение арифметических операций в NASM

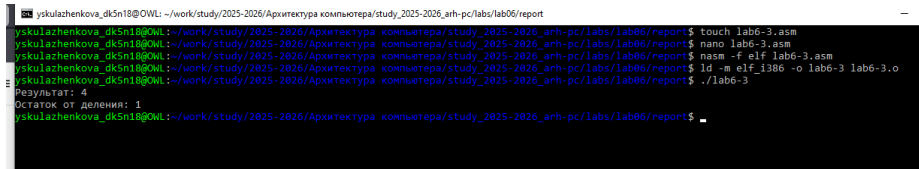
Напишем программу для вычисления значения арифметического выражения. Для этого создадим файл lab6-3.asm. Запишем нужный код программы (рис. 4.12).



```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report
GNU nano 7.2 lab6-3.asm *
;-----
; 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
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, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; --- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintf ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintf ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рисунок 4.12: Код программы

Создадим исполняемый файл и проверим результат работы программы (рис. 4.13).



```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ touch lab6-3.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano lab6-3.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf lab6-3.asm
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-3 lab6-3.o
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./lab6-3
Результат: 4
Остаток от деления: 1
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ _
```

Рисунок 4.13: Выполнение программы

Далее найдём значение другого выражения. Для этого изменим текст программы. Проверим работу программы (рис. 4.14).

```
yskulazhenkova_dk5n18@OWL: ~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ touch lab6-3.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano lab6-3.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf lab6-3.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-3 lab6-3.o
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./lab6-3
Результат: 4
Остаток от деления: 1
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano lab6-3.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf lab6-3.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o lab6-3 lab6-3.o
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./lab6-3
Результат: 5
Остаток от деления: 1
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$
```

Рисунок 4.14: Исполнение файла

В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета. Для написания кода создадим файл `variant.asm`. Введем в файл `variant.asm` следующую программу (рис. 4.16).

```
GNU nano 7.2 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 ; 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 sprint
mov eax, edx
call iprintf
call quit
```

Рисунок 4.15: Редактирование файла

После этого проверим работу программы (рис. 4.16).

```
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ touch variant.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano variant.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf variant.asm
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o variant variant.o
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./variant
Введите № студенческого билета:
183253409
Ваш вариант: 18
yskulazhenkova_dk5n18@OWL:~/work/study/2025-2026/Архитектура компьютера/study_2025-2026_arh-pc/labs/lab06/report$
```

Рисунок 4.16: Выполнение файла

4.3 Вопросы лабораторной работы

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения „Ваш вариант:“?

Ответ: Строки, отвечающие за вывод сообщения „Ваш вариант:“

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

2. Для чего используются следующие инструкции?

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

Ответ: данные инструкции используются для помещения адреса переменной x в регистр ecx, указания максимальной длины ввода (80 символов) и вызова подпрограммы чтения строки с клавиатуры

3. Для чего используется инструкция „call atoi“?

Ответ: функция atoi преобразует ascii-код символа в целое число и записывает результат в регистр eax.

4. Какие строки листинга 6.4 отвечают за вычисления варианта?

Ответ: строки, отвечающие за вычисления варианта.

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

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

Ответ: остаток от деления записывается в регистр EDX.

6. Для чего используется инструкция „inc edx“?

Ответ: инструкция «inc edx» используется для увеличения на 1 остатка от деления (номера варианта), поскольку варианты нумеруются с 1, а не с 0.

7. Какие строки листинга 6.4 отвечают за вывод на экран результата вычислений?

Ответ: строки, отвечающие за вывод результата вычислений.

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


5 Задание для самостоятельной работы

Напишем программу для вычисления значения функции. Затем проверим работу программы для различных функций и входных данных (рис. 5.1).

```
yskulazhenkova_dk5n18@Oml:~/work/study/2025-2026/Архитектура_компьютера/study_2025-2026_arh-pc/labs/lab06/report$ touch 18-var.asm
yskulazhenkova_dk5n18@Oml:~/work/study/2025-2026/Архитектура_компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nano 18-var.asm
yskulazhenkova_dk5n18@Oml:~/work/study/2025-2026/Архитектура_компьютера/study_2025-2026_arh-pc/labs/lab06/report$ nasm -f elf 18-var.asm
yskulazhenkova_dk5n18@Oml:~/work/study/2025-2026/Архитектура_компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ld -m elf_i386 -o 18-var.o 18-var.o
Программа вычисления функции  $f(x) = 3(x + 10) - 20$ 
F(x) = 3(x + 10) - 20
Введите значение x: 100
Результат вычисления: 310
yskulazhenkova_dk5n18@Oml:~/work/study/2025-2026/Архитектура_компьютера/study_2025-2026_arh-pc/labs/lab06/report$ ./18-var
Программа вычисления функции  $f(x) = 3(x + 10) - 20$ 
F(x) = 3(x + 10) - 20
Введите значение x: 1.5
Результат вычисления: 13
yskulazhenkova_dk5n18@Oml:~/work/study/2025-2026/Архитектура_компьютера/study_2025-2026_arh-pc/labs/lab06/report$ .
```

Рисунок 5.1: Самостоятельная работа

6 Выводы

В результате выполнения лабораторной работы была достигнута основная цель - освоение арифметических инструкций языка ассемблера NASM. На практике были успешно применены команды сложения (add), вычитания (sub), умножения (mul/imul) и деления (div/ldiv), а также изучены особенности работы с операндами в регистрах и памяти. Особое внимание было уделено правилам использования регистров при операциях умножения и деления, где результат занимает фиксированные пары регистров в зависимости от размера операндов.

Важным результатом работы стало освоение методики преобразования данных между символьным и числовым представлением с использованием подпрограмм `iprint`, `iprintLF` и `atoi`. Практическое применение этих функций позволило организовать корректный ввод-вывод данных и выполнение арифметических операций над числами, введенными пользователем. Полученные навыки программирования на языке ассемблера NASM являются фундаментальными для дальнейшего изучения архитектуры ЭВМ и низкоуровневого программирования.