Лабораторная работа №6

Арифметические операции в NASM

Солдатов Алексей

Содержание

# 1 Цель работы

Освоить арифметические инструкции языка ассемблера NASM.

# 2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Ответы на вопросы
4. Выполнение заданий для самостоятельной работы

# 3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. [1](#tbl:std-dir) приведено краткое описание стандартных каталогов Unix.

Table 1: Описание некоторых каталогов файловой системы GNU Linux

| Имя каталога | Описание каталога |
| --- | --- |
| / | Корневая директория, содержащая всю файловую |
| /bin | Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям |
| /etc | Общесистемные конфигурационные файлы и файлы конфигурации установленных программ |
| /home | Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя |
| /media | Точки монтирования для сменных носителей |
| /root | Домашняя директория пользователя root |
| /tmp | Временные файлы |
| /usr | Вторичная иерархия для данных пользователя |

Более подробно об Unix см. в [1–6].

# 4 Выполнение лабораторной работы

## 4.1 Символьные и численные данные в NASM

Создал каталог для программ лабораторной работы №6, перешел в него и создал файл “lab6-1.asm” (рис. [**fig?**];001).

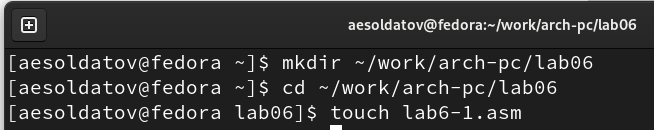


Figure 1: Подготовка к работе

Ввел в файл “lab6-1.asm” текст программы из листинга 6.1. со страницы в ТУИС (рис. [2](#fig:002)).



Figure 2: Ввод команд

%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

Перенес файл “in\_out.asm” из прошлой папки с лабораторной работой в нынешнюю (рис. [3](#fig:003)).

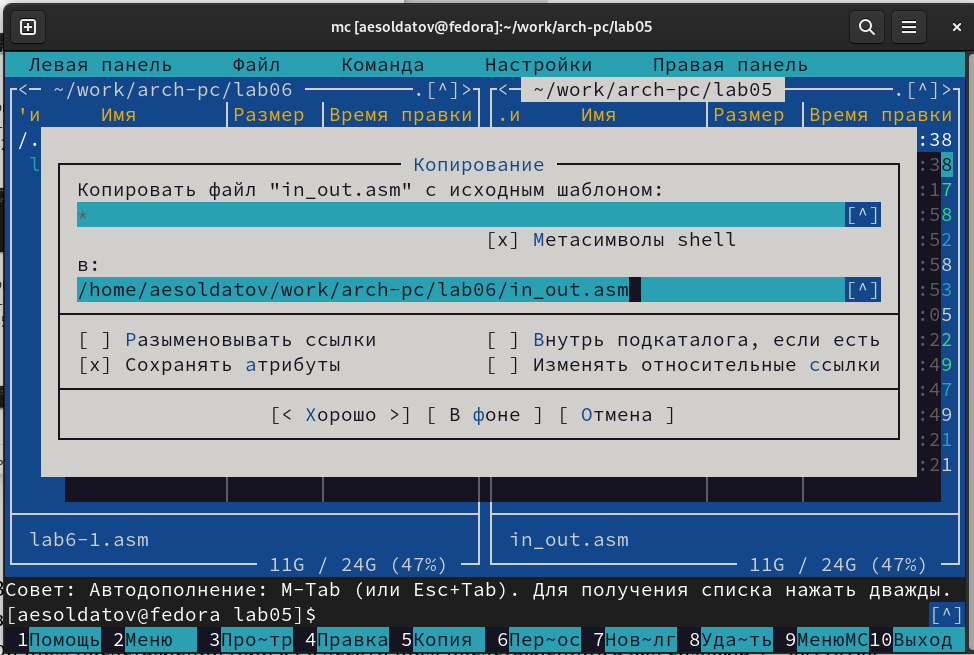


Figure 3: Перенос файла

Создал исполняемый файл и запустил его (рис. [4](#fig:004)).

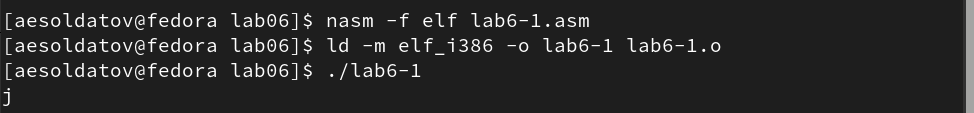


Figure 4: Создание и запуск файла

Далее изменил текст программы и вместо символов, записал в регистры числа (рис. [5](#fig:005)).



Figure 5: Изменение текста программы

%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

Создал исполняемый файл и запустил его (рис. [6](#fig:006)).

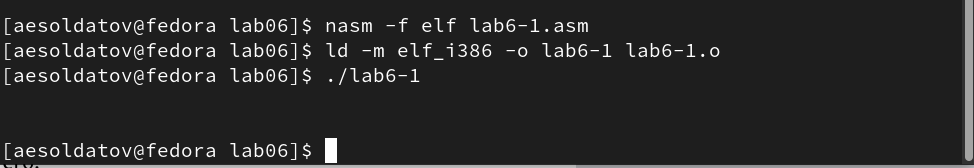


Figure 6: Создание файла

В данном случае выводится символ с кодом 10, этот символ не отображается при выводе на экран.

Создал файл “lab6-2.asm” в каталоге “~/work/arch-pc/lab06” (рис. [7](#fig:007)).

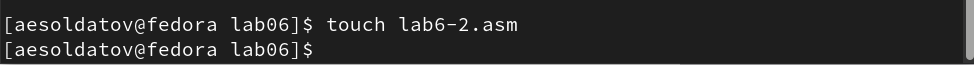


Figure 7: Создание файла

Ввел в него текст программы из листинга 6.2 со страницы в ТУИС (рис. [8](#fig:008)).



Figure 8: Ввод команд

%include 'in\_out.asm'  
SECTION .text  
GLOBAL \_start  
\_start:  
mov eax,'6'  
mov ebx,'4'  
add eax,ebx  
call iprintLF  
call quit

Создал исполняемый файл и запустил его (рис. [9](#fig:009)).

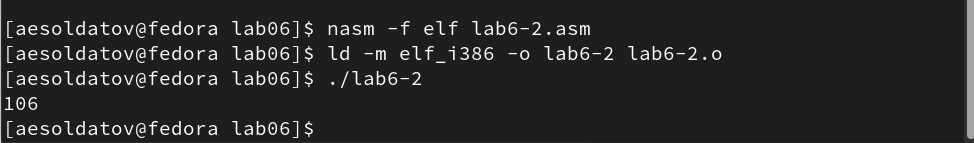


Figure 9: Создание и запуск файла

Исправил текст программы и вместо символов, записал в регистры числа (рис. [10](#fig:010)).



Figure 10: Редактирование

Создал исполняемый файл и проверил его работу (рис. [11](#fig:011)).

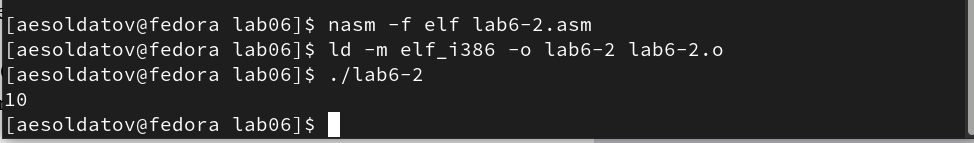


Figure 11: Создание файла

Заменил функцию “iprintLF” на “iprint” (рис. [12](#fig:011-1)).

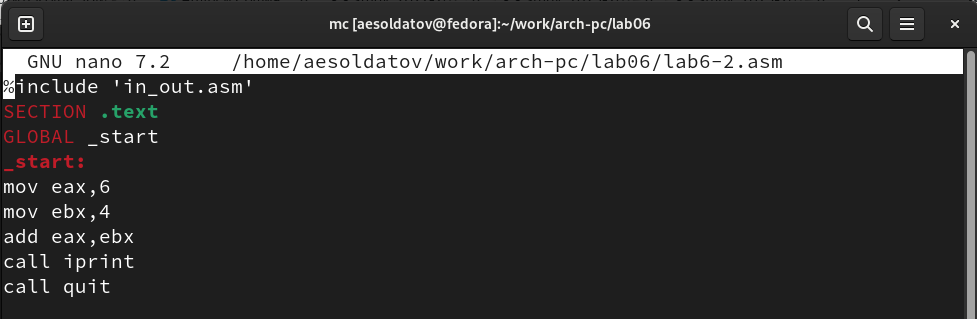


Figure 12: Редактирование файла

Создал исполняемый файл и проверил его работу (рис. [13](#fig:011-2)).

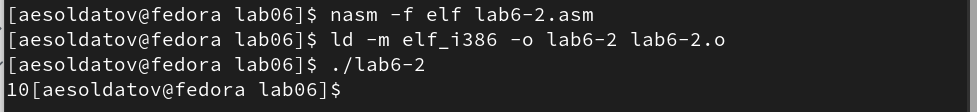


Figure 13: Сборка файла

Подпрограмма “iprintLF” делает перевод на новую строку, а “iprint” остается на той же.

## 4.2 Выполнение арифметических операций в NASM

Создал файл “lab6-3.asm” в каталоге “~/work/arch-pc/lab06” (рис. [14](#fig:012)).

Figure 14: Создание нового файла

Figure 14: Создание нового файла

Внимательно изучил текст программы из листинга 6.3 со страницы в ТУИС и ввел его в “lab6-3.asm” (рис. [15](#fig:013)).

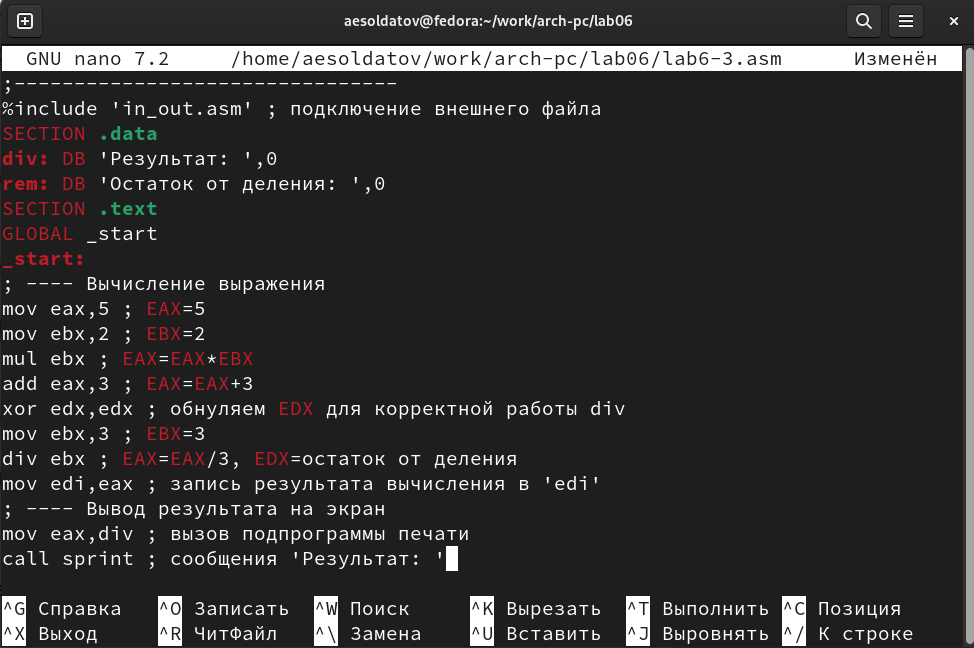


Figure 15: Ввод команд

;--------------------------------  
; 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 iprintLF ; 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 iprintLF ; из 'edx' (остаток) в виде символов  
call quit ; вызов подпрограммы завершения

Создал исполняемый файл и проверил его работу, результат совпал с примером из лабораторной (рис. [16](#fig:014)).

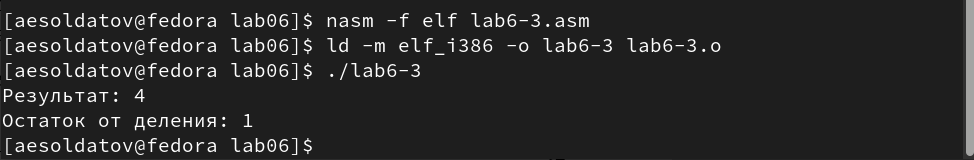


Figure 16: Создание файла

Изменил текст программы для вычисления выражения f(x) = (4 \* 6 + 2)/5 (рис. [17](#fig:015)).

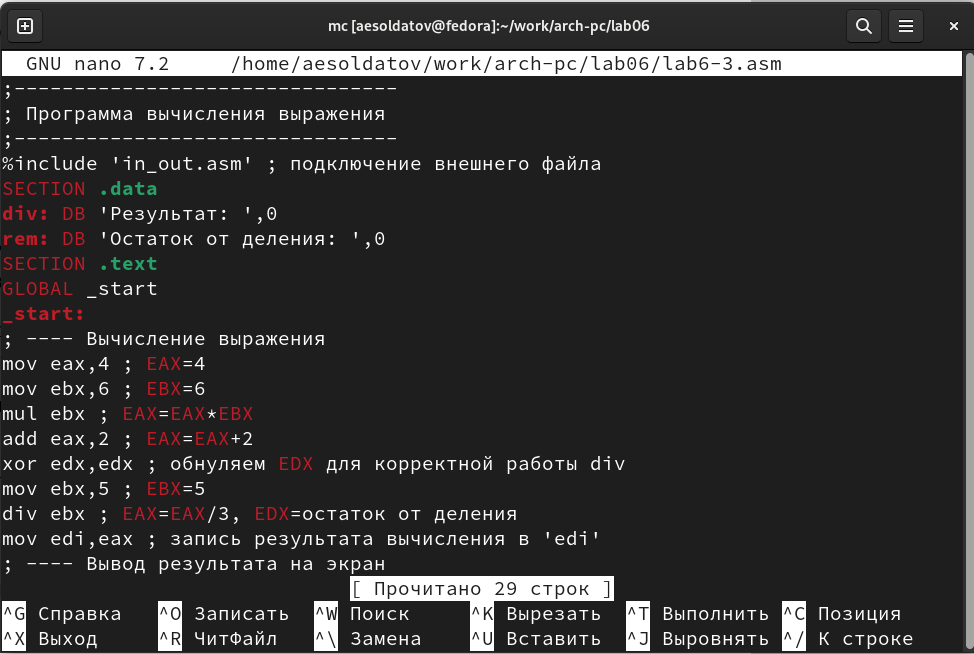


Figure 17: Внесение изменений в код программы

;--------------------------------  
; 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,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/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 iprintLF ; 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 iprintLF ; из 'edx' (остаток) в виде символов  
call quit ; вызов подпрограммы завершения

Создал исполняемый файл и проверил его работу (рис. [18](#fig:016)).

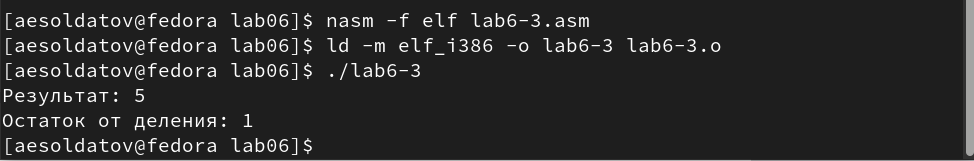


Figure 18: Создание и проверка работы файла

Создал файл “variant.asm” в каталоге “~/work/arch-pc/lab06” (рис. [19](#fig:017)).

Figure 19: Создание файла

Figure 19: Создание файла

Внимательно изучил текст программы из листинга 6.4 на странице в ТУИС и ввел его в файл variant.asm (рис. [20](#fig:018)).

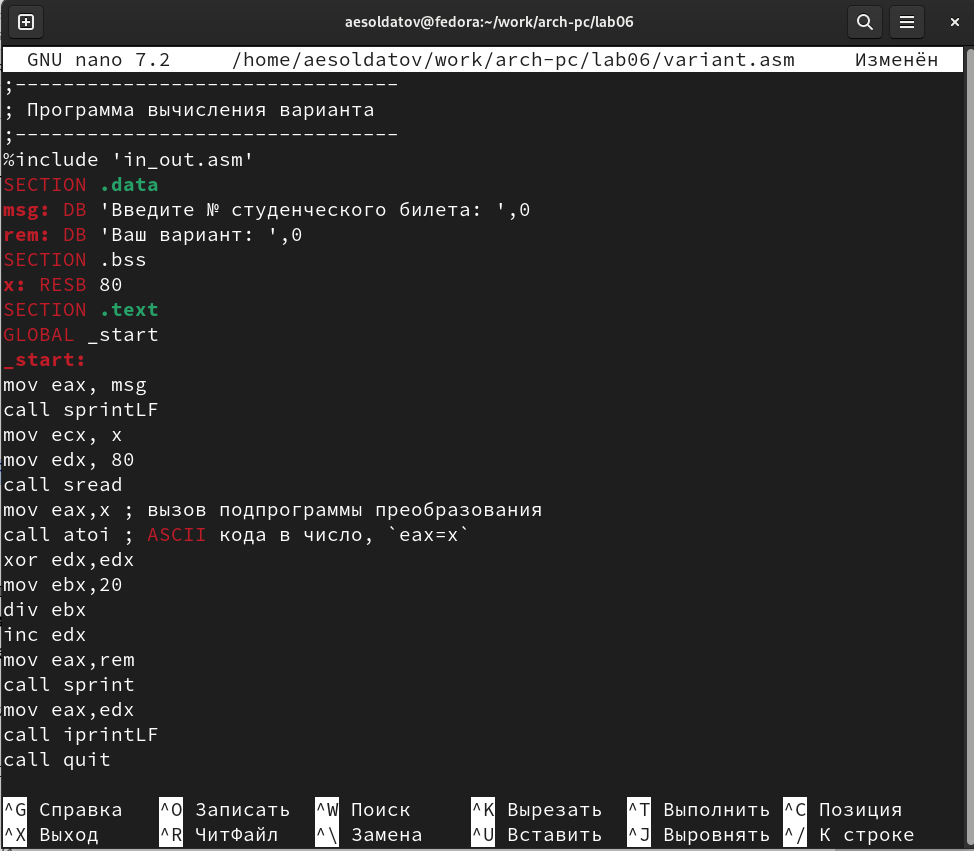


Figure 20: Ввод текста в файл

;--------------------------------  
; 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  
call sprintLF  
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 iprintLF  
call quit

Создал исполняемый файл и проверил его работу, программа выдала, что мой номер 10 (Ответ совпадает с аналитическим решением) (рис. [21](#fig:019)).

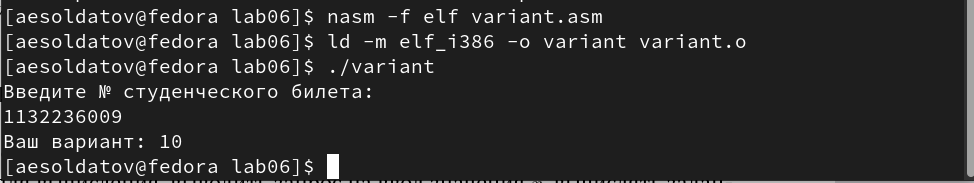


Figure 21: Создание и проверка работы файла

## 4.3 Ответы на вопросы

1. За вывод сообщения “Ваш вариант” отвечают строчки кода

mov eax,rem  
call sprint

1. Инструкция “mov, есx, х” используется, чтобы положить адрес вводимой строки “х” в регистр “есх”. “mov edx, 80” - запись в регистр “edx” длины вводимой строки. “call sread” - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры.
2. “call atoi” используется для вызова подпрограммы из внешнего файла, которая преобразует ascii-код символа в целое число и записывает результат в регистр “eax”.
3. За вычисления варианта отвечают строки:

хот edx, edx ; обнуление еах для корректной работы div  
mov ebx, 20 ; ebx = 20  
div ebx ; eax = eax/20, edx - остаток от деления  
inc edx ; edx = edx + 1

1. При выполнении инструкции “div ebx” остаток от деления записывается в регистр “edx”.
2. Инструкция “inc edx” увеличивает значение регистра “edx” на 1.
3. За вывод на экран результатов вычислений отвечают строки:

mov eax, edx  
call iprintLF

## 4.4 Выполнение заданий для самостоятельной работы

Создаю файл “samrab.asm” (рис. [22](#fig:019-1)).

Figure 22: Создание файла в каталоге

Figure 22: Создание файла в каталоге

Пишу в нем программу по варианту, вычисленному из предыдущей программы (рис. [23](#fig:020)).

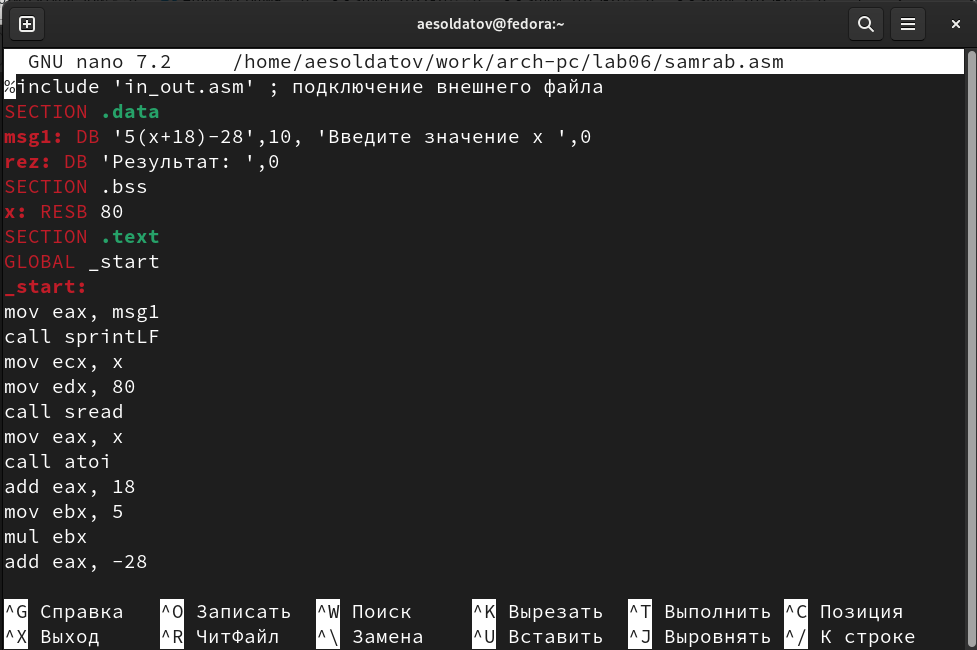


Figure 23: Написание программы

%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data  
msg1: DB '5(x+18)-28',10, 'Введите значение x ',0  
rez: DB 'Результат: ',0  
SECTION .bss  
x: RESB 80  
SECTION .text  
GLOBAL \_start  
\_start:  
mov eax, msg1  
call sprintLF  
mov ecx, x  
mov edx, 80  
call sread  
mov eax, x  
call atoi  
add eax, 18  
mov ebx, 5  
mul ebx  
add eax, -28  
mov edi, eax  
mov eax, rez  
call sprint  
mov eax, edi  
call iprintLF  
call quit

Создаю исполняемый файл и запускаю программу (рис. [24](#fig:021)).

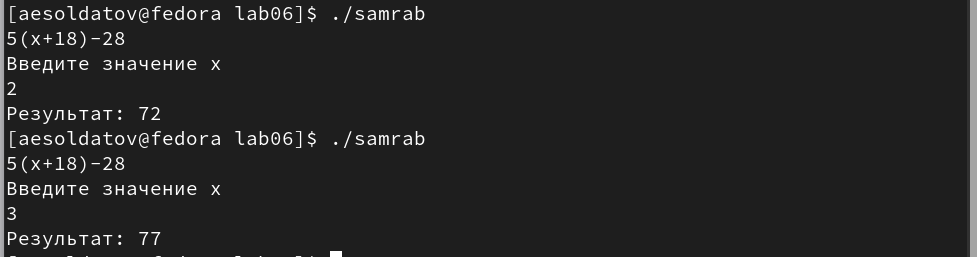


Figure 24: Запуск программы

Программа выдает правильный ответ.

# 5 Выводы

Освоил арифметические инструкции языка ассемблера NASM.

# Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016. URL: <https://www.gnu.org/software/bash/manual/>.

2. Newham C. [Learning the bash Shell: Unix Shell Programming](http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658). O’Reilly Media, 2005. 354 с.

3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.

4. Robbins A. [Bash Pocket Reference](https://www.ncbi.nlm.nih.gov/pubmed/25246403). O’Reilly Media, 2016. 156 с.

5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.

6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.