Лабораторная работа №7

Команды безусловного и условного переходов в Nasm. Программирование ветвлений.

Солдатов Алексей

Содержание

# 1 Цель работы

Изучить команды условного и безусловного переходов. Приобрести навыки написания программ с использованием переходов. Познакомиться с назначением и структурой файла листинга.

# 2 Задание

1. Реализация переходов в NASM
2. Изучение структуры листинга
3. 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

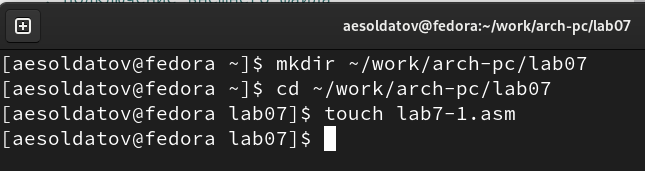
| Имя каталога | Описание каталога |
| --- | --- |
| / | Корневая директория, содержащая всю файловую |
| /bin | Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям |
| /etc | Общесистемные конфигурационные файлы и файлы конфигурации 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

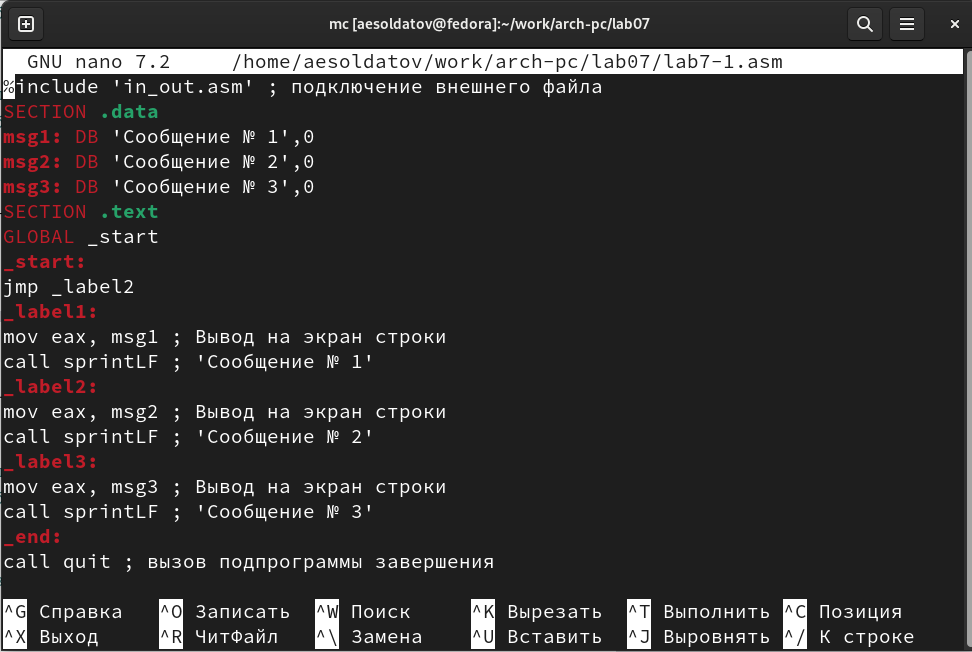
## 4.1 Реализация переходов в NASM

Создал каталог для программ лабораторной работы №7, перешел в него и создал файл “lab7-1.asm” (рис. ??).



Подготовка к работе

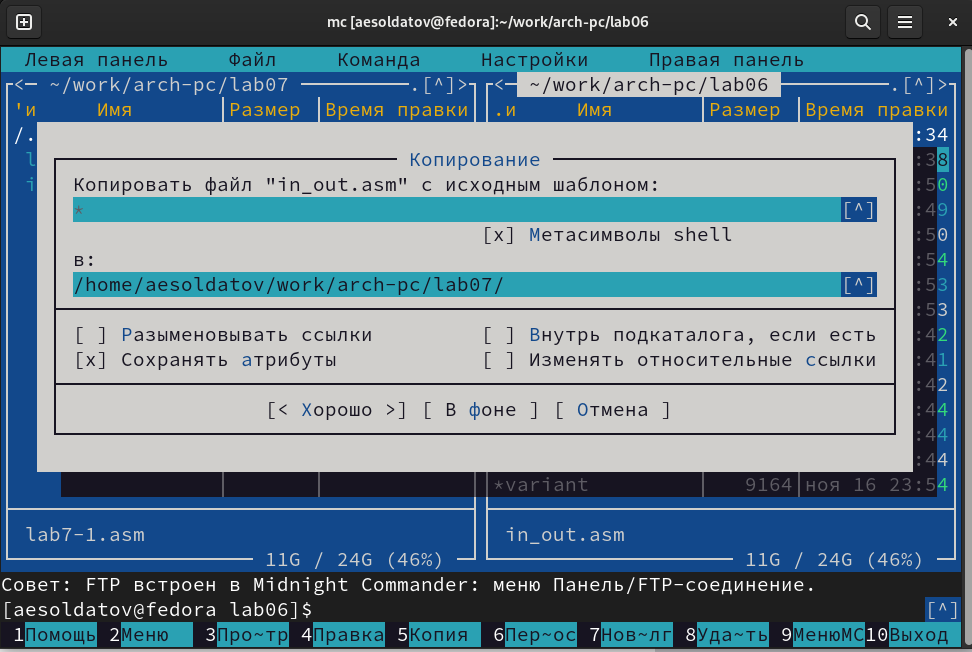
Ввел в файл “lab7-1.asm” текст программы из листинга 7.1. со страницы в ТУИС (рис. ??).



Ввод команд

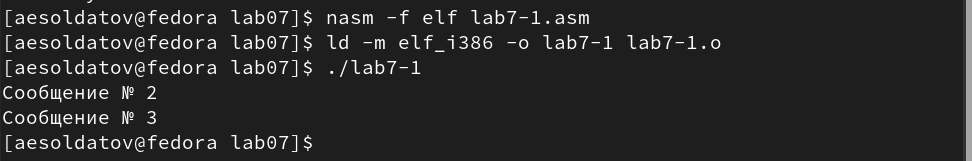
%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data  
msg1: DB 'Сообщение № 1',0  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL \_start  
\_start:  
jmp \_label2  
\_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 1'  
\_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 2'  
\_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 3'  
\_end:  
call quit ; вызов подпрограммы завершения

Перенес файл “in\_out.asm” из прошлой папки с лабораторной работой в нынешнюю (рис. ??).



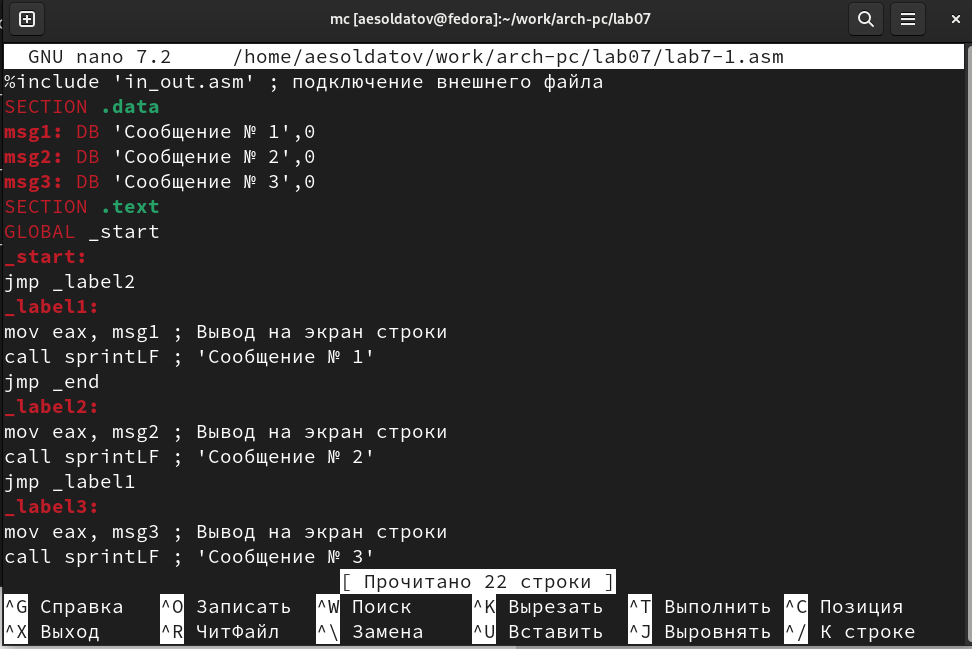
Перенос файла

Создал исполняемый файл и запустил его. Он совпадает с примером (рис. ??).



Создание и запуск файла

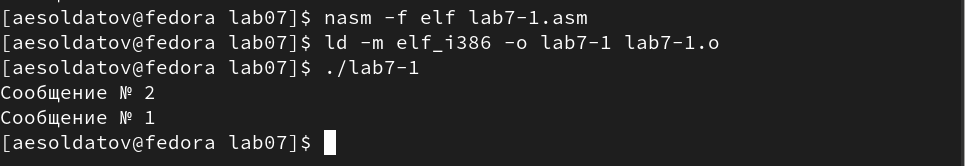
Далее изменил текст программы в соответствии с листингом 7.2 (рис. ??).



Изменение текста программы

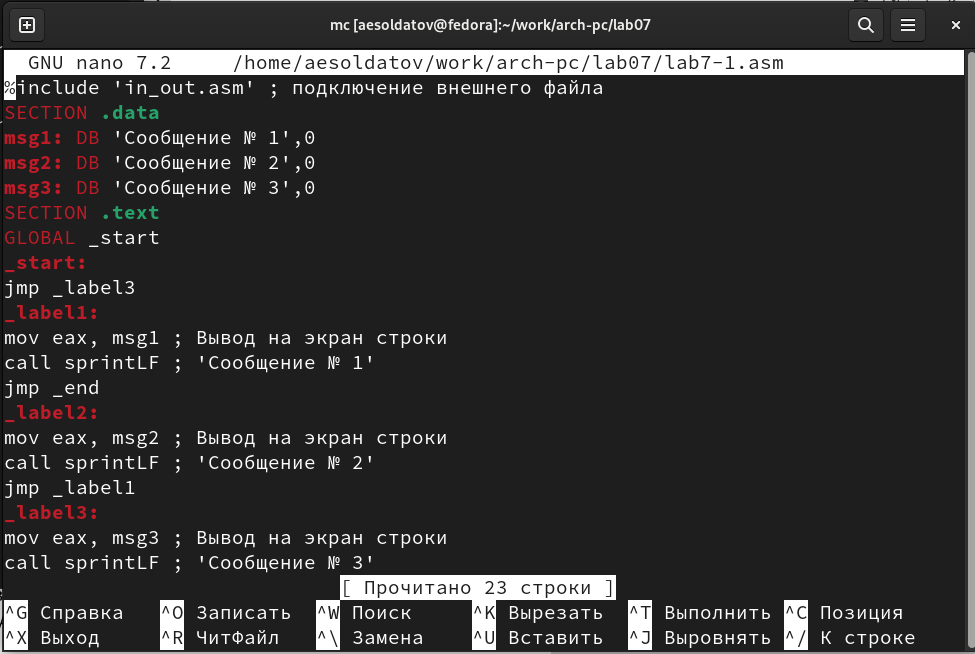
%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data  
msg1: DB 'Сообщение № 1',0  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL \_start  
\_start:  
jmp \_label2  
\_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 1'  
jmp \_end  
\_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 2'  
jmp \_label1  
\_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 3'  
\_end:  
call quit ; вызов подпрограммы завершения

Создал исполняемый файл и запустил его, программа работает правильно (рис. ??).



Создание файла

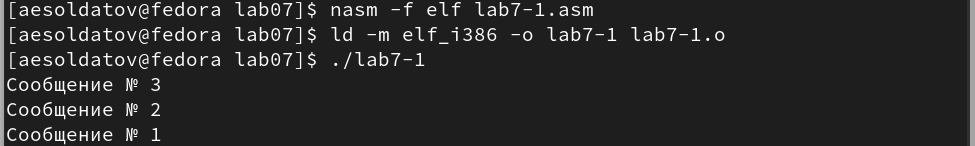
Изменил текст программы, чтобы вывод программы соответствовал примеру (рис. ??).



Изменение текста программы

%include 'in\_out.asm' ; подключение внешнего файла  
SECTION .data  
msg1: DB 'Сообщение № 1',0  
msg2: DB 'Сообщение № 2',0  
msg3: DB 'Сообщение № 3',0  
SECTION .text  
GLOBAL \_start  
\_start:  
jmp \_label3  
\_label1:  
mov eax, msg1 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 1'  
jmp \_end  
\_label2:  
mov eax, msg2 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 2'  
jmp \_label1  
\_label3:  
mov eax, msg3 ; Вывод на экран строки  
call sprintLF ; 'Сообщение № 3'  
jmp \_label2  
\_end:  
call quit ; вызов подпрограммы завершения

Создал исполняемый файл и запустил его, вывод соответствует примеру (рис. ??).



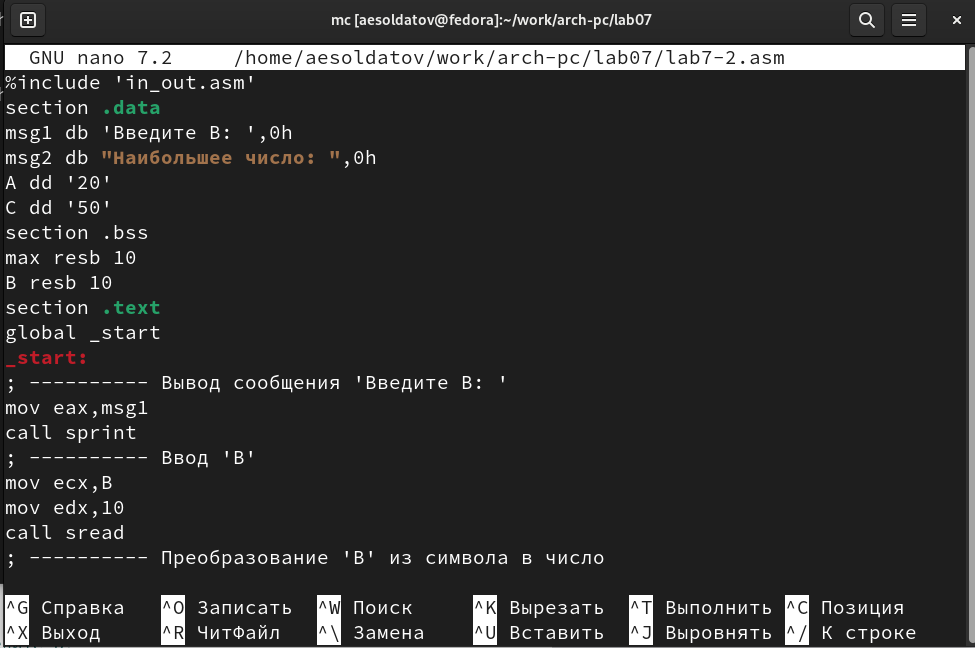
Создание и проверка работы файла

Создал файл “lab7-2.asm” в каталоге “~/work/arch-pc/lab07” (рис. ??).

Создание файла

Создание файла

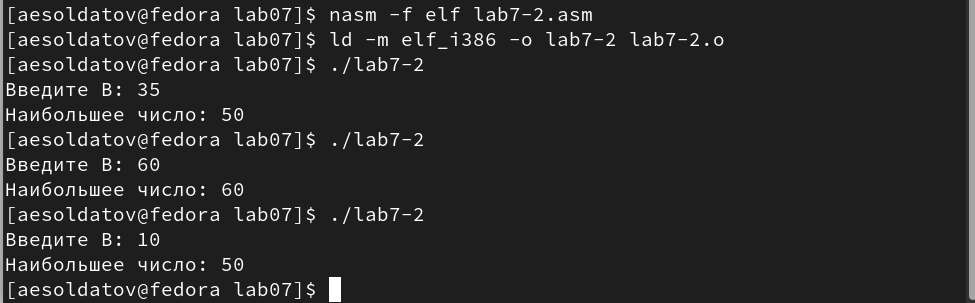
Внимательно изучил текст программы из листинга 7.3 и ввел его в файл (рис. ??).



Ввод команд

%include 'in\_out.asm'  
section .data  
msg1 db 'Введите B: ',0h  
msg2 db "Наибольшее число: ",0h  
A dd '20'  
C dd '50'  
section .bss  
max resb 10  
B resb 10  
section .text  
global \_start  
\_start:  
; ---------- Вывод сообщения 'Введите B: '  
mov eax,msg1  
call sprint  
; ---------- Ввод 'B'  
mov ecx,B  
mov edx,10  
call sread  
; ---------- Преобразование 'B' из символа в число  
mov eax,B  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [B],eax ; запись преобразованного числа в 'B'  
; ---------- Записываем 'A' в переменную 'max'  
mov ecx,[A] ; 'ecx = A'  
mov [max],ecx ; 'max = A'  
; ---------- Сравниваем 'A' и 'С' (как символы)  
cmp ecx,[C] ; Сравниваем 'A' и 'С'  
jg check\_B ; если 'A>C', то переход на метку 'check\_B',  
mov ecx,[C] ; иначе 'ecx = C'  
mov [max],ecx ; 'max = C'  
; ---------- Преобразование 'max(A,C)' из символа в число  
check\_B:  
mov eax,max  
call atoi ; Вызов подпрограммы перевода символа в число  
mov [max],eax ; запись преобразованного числа в `max`  
; ---------- Сравниваем 'max(A,C)' и 'B' (как числа)  
mov ecx,[max]  
cmp ecx,[B] ; Сравниваем 'max(A,C)' и 'B'  
jg fin ; если 'max(A,C)>B', то переход на 'fin',  
mov ecx,[B] ; иначе 'ecx = B'  
mov [max],ecx  
; ---------- Вывод результата  
fin:  
mov eax, msg2  
call sprint ; Вывод сообщения 'Наибольшее число: '  
mov eax,[max]  
call iprintLF ; Вывод 'max(A,B,C)'  
call quit ; Выход

Создал исполняемый файл и проверил его работу для разных значений (рис. ??).



Создание файла

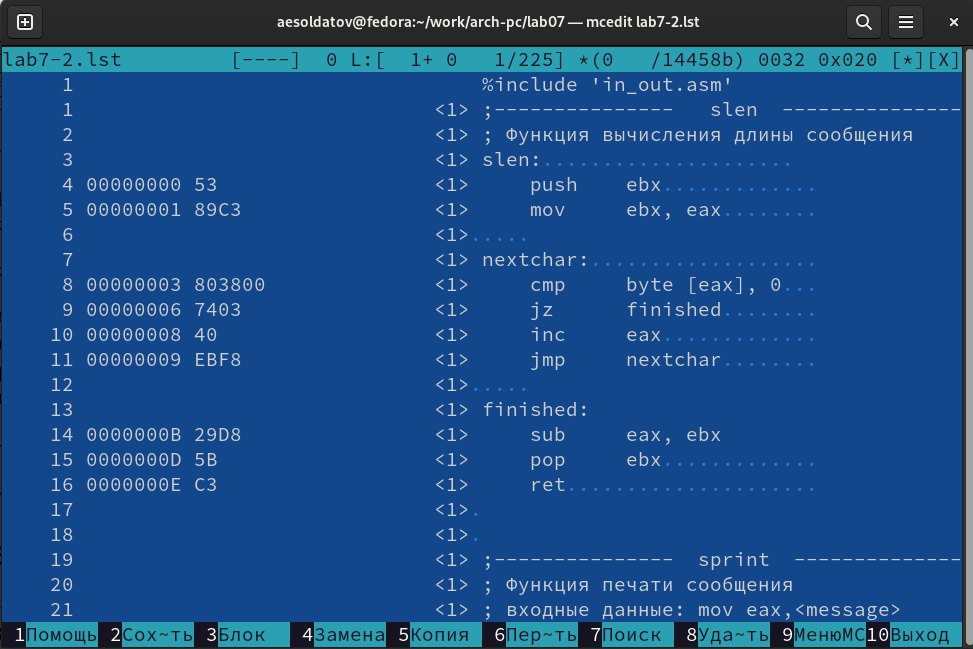
## 4.2 Изучение структуры листинга

Создал файл листинга для файла “lab7-2.asm” с помощью ключа “-l” и открыл его (рис. ??).

Создание листинга файла

Создание листинга файла

Внимательно изучил его формат и содержимое (рис. ??).



Ознакомление

Строки в файле листинга имеют следующую структуру. Первое число обозначает номер строки файла листинга (может не соответствовать номеру строки в файле с исходным кодом). Далее идет адрес (указывает на смещение машинного кода от начала текущего сегмента). Потом идет сам машинный код, он представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности. А в конце пишется исходный текст программы, которая является исходной строкой программы вместе с комментариями.

Первая строка, которую я буду описывать имеет номер 38. Далее идет ее адрес (0000013А), который дает нам информацию о смещении машинного кода. Следом идет сам машинный код который является ассемблированием инструкции “mov ecx,max” в шестнадцатеричный код. В конце мы видим как выглядит строка в самой программе (рис. ??).

Анализ первой строки

Анализ первой строки

Вторая строка, которую я буду описывать имеет номер 29. Далее идет ее адрес (0000011D), который дает нам информацию о смещении машинного кода. Следом идет сам машинный код который является ассемблированием инструкции “jg check\_B” в шестнадцатеричный код. В конце мы видим как выглядит строка в самой программе (рис. ??).

Анализ второй строки

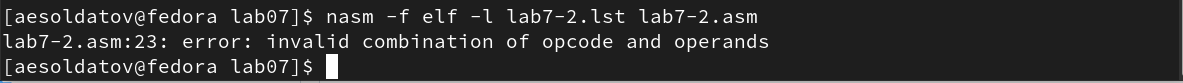
Анализ второй строки

Третья строка, которую я буду описывать имеет номер 35. Далее идет ее адрес (00000130), который дает нам информацию о смещении машинного кода. Следом идет сам машинный код который является ассемблированием инструкции “call atoi” в шестнадцатеричный код. В конце мы видим как выглядит строка в самой программе (рис. ??).

Анализ третьей строки

Анализ третьей строки

Открыл файл с программой “lab7-2.asm” и в инструкции с двумя операндами удалил один операнд, выполнил трансляцию листинга и получил ошибку с указанием номера строки (рис. ??).



Попытка транслирования файла листинга

После этого на месте ошибки в файле листинга появился комментарий с типом ошибки (рис. ??).

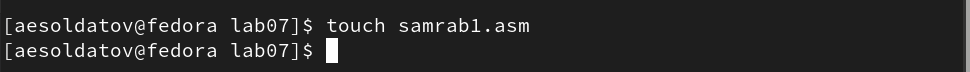
Ошибка в листинге

Ошибка в листинге

## 4.3 Выполнение заданий для самостоятельной работы

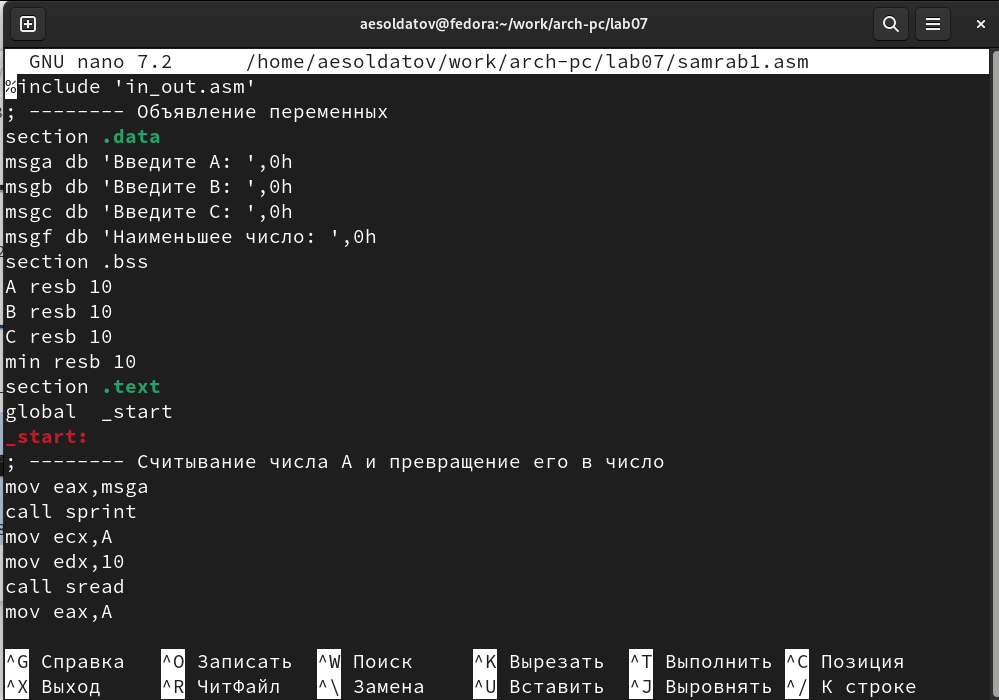
### 4.3.1 Задание 1

Создал файл “samrab1.asm” (рис. ??).



Создание файла

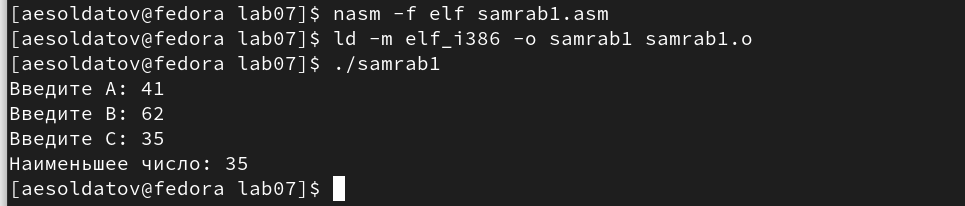
Написал программу нахождения наименьшей из трех целочисленных переменных (рис. ??).



Ввод текста программы

%include 'in\_out.asm'  
; -------- Объявление переменных  
section .data  
msga db 'Введите A: ',0h  
msgb db 'Введите B: ',0h  
msgc db 'Введите C: ',0h  
msgf db 'Наименьшее число: ',0h  
section .bss  
A resb 10  
B resb 10  
C resb 10  
min resb 10  
section .text  
global \_start  
\_start:  
; -------- Считывание числа А и превращение его в число  
mov eax,msga  
call sprint  
mov ecx,A  
mov edx,10  
call sread  
mov eax,A  
call atoi  
mov [A],eax  
; -------- Считывание числа В и превращение его в число  
mov eax,msgb  
call sprint  
mov ecx,B  
mov edx,10  
call sread  
mov eax,B  
call atoi  
mov [B],eax  
; -------- Считывание числа C и превращение его в число  
mov eax,msgc  
call sprint  
mov ecx,C  
mov edx,10  
call sread  
mov eax,C  
call atoi  
mov [C],eax  
; -------- Нахождение максимума  
mov ecx,[A]  
mov [min],ecx  
cmp ecx,[C]  
jl check\_B  
mov ecx,[C]  
mov [min],ecx  
  
check\_B:  
cmp ecx,[B]  
jl fin  
mov ecx,[B]  
mov [min],ecx  
  
fin:  
mov eax, msgf  
call sprint  
mov eax,[min]  
call iprintLF  
call quit

Создал исполняемый файл и ввел значения переменных из таблицы 7.5 в соответствии с вариантом из прошлой работы (рис. ??).



Проверка работы программы

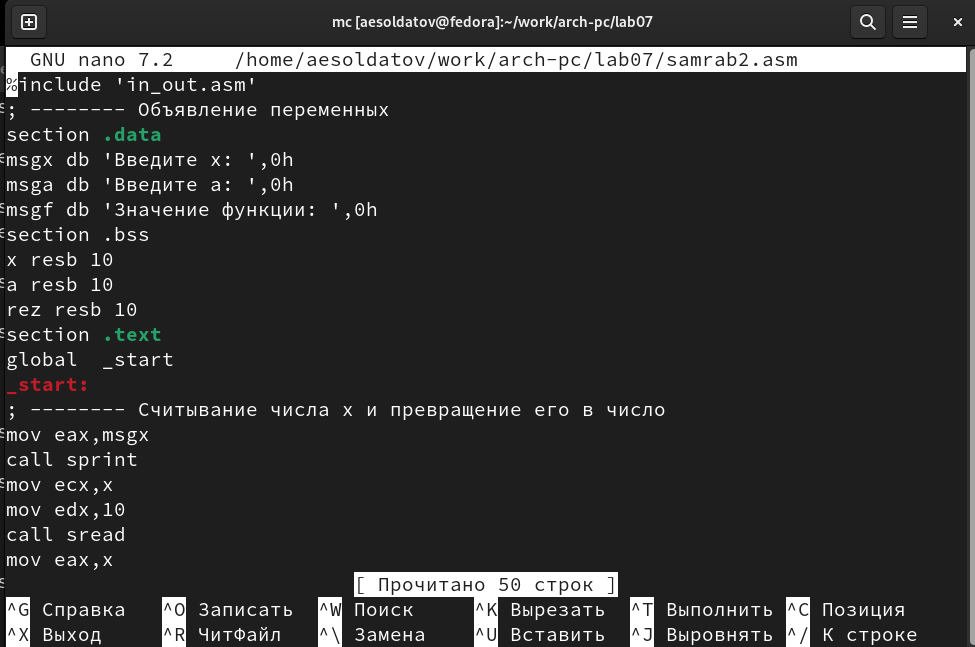
### 4.3.2 Задание 2

Создал файл “samrab2.asm” (рис. ??).

Создание файла

Создание файла

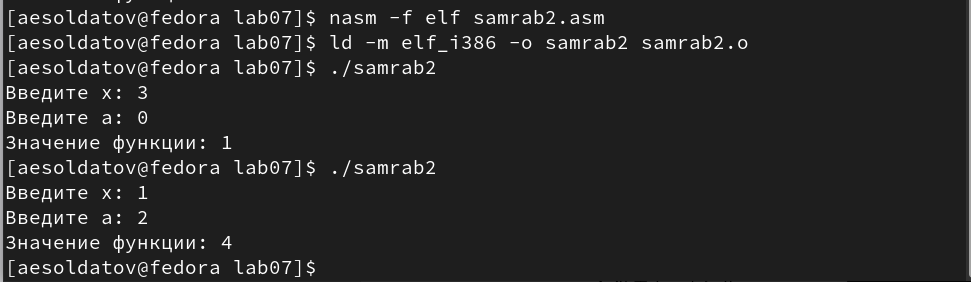
Написал программу для вычисления значения функции (вид функции взял из таблицы 7.6 в соответствии с вариантом из прошлой работы) (рис. ??).



Ввод текста программы

%include 'in\_out.asm'  
; -------- Объявление переменных  
section .data  
msgx db 'Введите x: ',0h  
msga db 'Введите a: ',0h  
msgf db 'Значение функции: ',0h  
section .bss  
x resb 10  
a resb 10  
rez resb 10  
section .text  
global \_start  
\_start:  
; -------- Считывание числа x и превращение его в число  
mov eax,msgx  
call sprint  
mov ecx,x  
mov edx,10  
call sread  
mov eax,x  
call atoi  
mov [x],eax  
; -------- Считывание числа a и превращение его в число  
mov eax,msga  
call sprint  
mov ecx,a  
mov edx,10  
call sread  
mov eax,a  
call atoi  
mov [a],eax  
; -------- Расчет значения функции  
mov eax,[x]  
cmp eax,2  
ja \_else  
mov eax,[a]  
mov ebx,3  
mul ebx  
mov [rez],eax  
  
\_else:  
add eax,-2  
mov [rez],eax  
  
fin:  
mov eax, msgf  
call sprint  
mov eax,[rez]  
call iprintLF  
call quit

Создал исполняемый файл и ввел значения переменных из таблицы 7.6 в соответствии с вариантом из прошлой работы (рис. ??).



Проверка работы программы

# 5 Выводы

Изучил команды условного и безусловного переходов. Приобрел навыки написания программ с использованием переходов. Познакомился с назначением и структурой файла листинга.

# 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.