Отчёт по лабораторной работе №7

Дисциплина: архитектура компьютера

Арфонос Дмитрий

Содержание

# 1 Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга

# 2 Выполнение лабораторной работы

**1**

С помощью утилиты mkdir создаю директорию lab07, перехожу в нее и создаю файл для работы. (рис. [[1](#fig:001)])



Figure 1: Создание директории

**2**

Копирую в текущий каталог файл in\_out.asm из загрузок, т.к. он будет использоваться в других программах (рис. [[2](#fig:002)]).

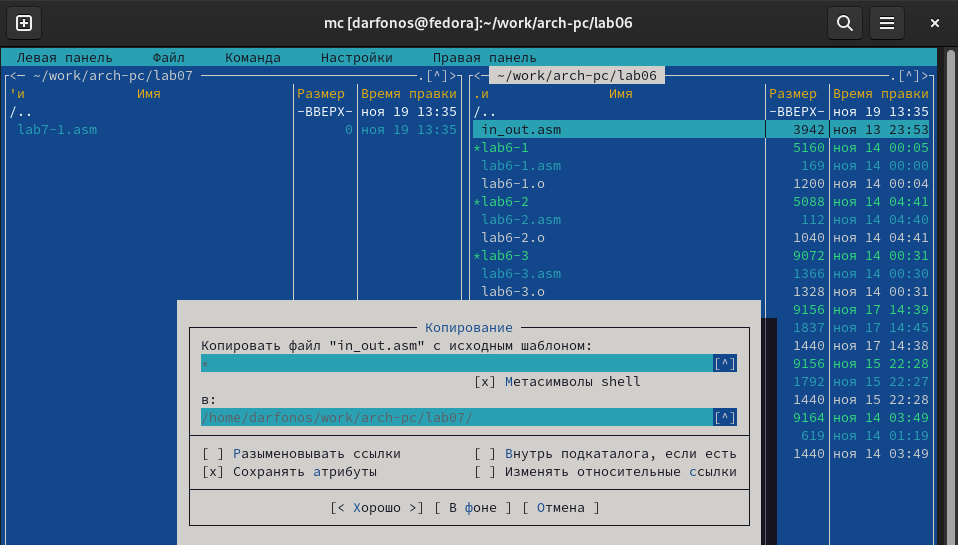


Figure 2: Создание копии файла для дальнейшей работы

**3**

Открываю созданный файл lab7-1.asm, вставляю в него программу реализации безусловных переходов(рис. [[3](#fig:003)]).

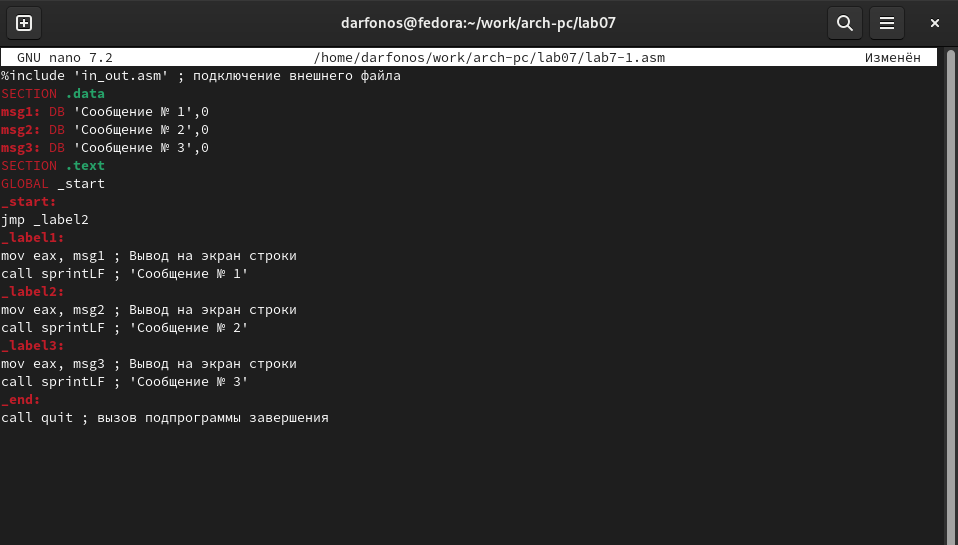


Figure 3: Редактирование файла

**4**

Создаю исполняемый файл программы и запускаю его (рис. [[4](#fig:004)]). Инструкции jmp \_label2 меняет порядок исполнения инструкций и позволяет выполнить инструкции начиная с метки \_label2.

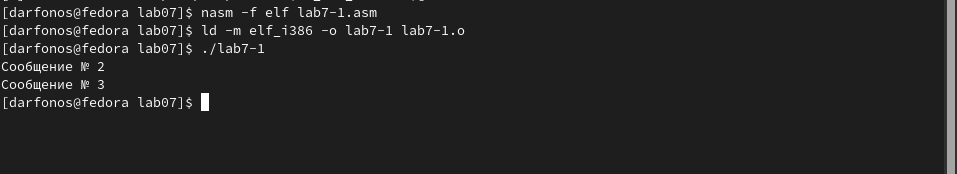


Figure 4: Запуск исполняемого файла

**5**

Изменяю текст программы так, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу (рис. [[5](#fig:005)]).

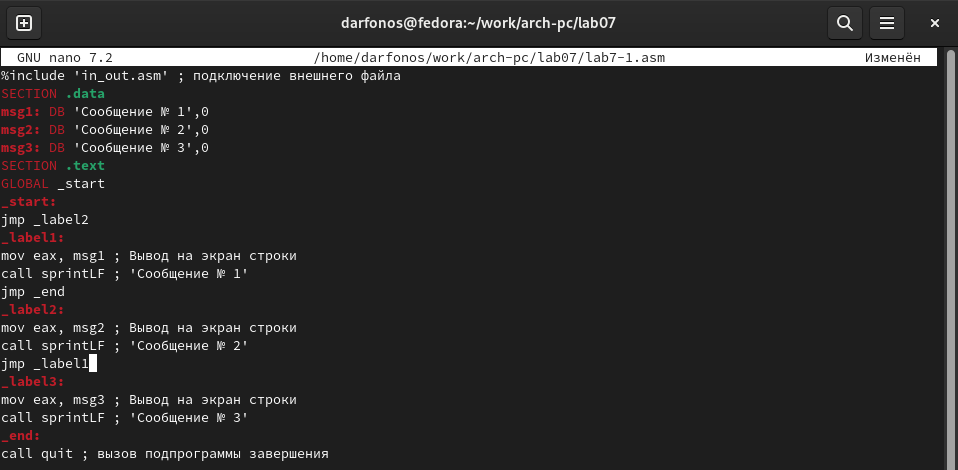


Figure 5: Редактирование файла

**6**

Создаю новый исполняемый файл программы и запускаю его (рис. [[6](#fig:006)]). Убеждаюсь в том, программа раотает верно.

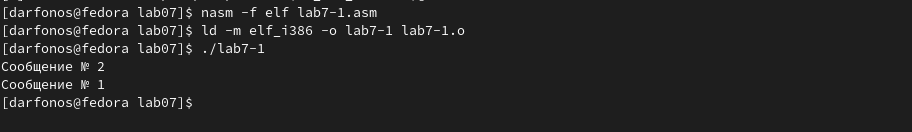


Figure 6: Запуск исполняемого файла

**7**

Изменяю текст программы, так чтобы вывод происходил в обратном порядке (рис. [[7](#fig:007)]).

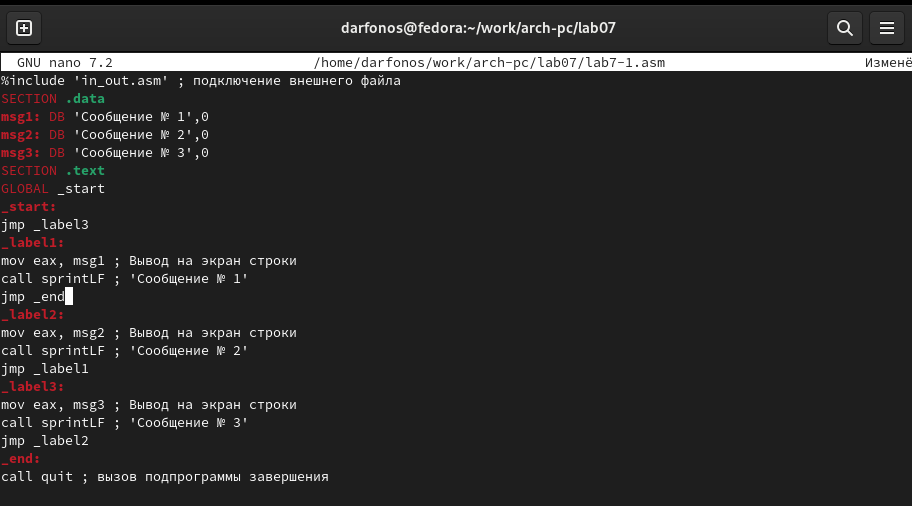


Figure 7: Редактирование программы

**8**

Создаю исполняемый файл и проверяю работу программы (рис. [[8](#fig:008)]). Программа отработало верно.

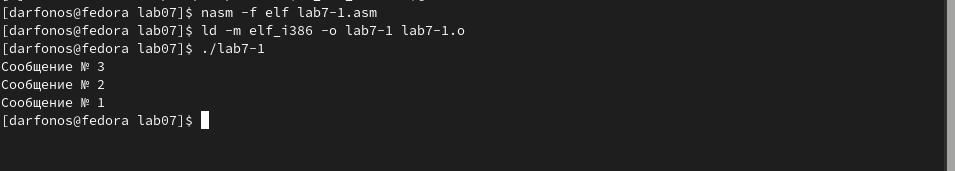


Figure 8: Создание исполняемого файла

**9**

Создаю новый файл lab7-2.asm для программы с условным оператором. (рис. [[9](#fig:009)]).

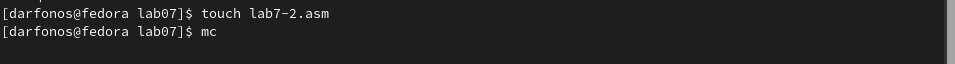


Figure 9: Создание файла

**10**

Вставляю программу, которая определяет и выводит на экран наибольшее число (рис.[[10](#fig:010)]).

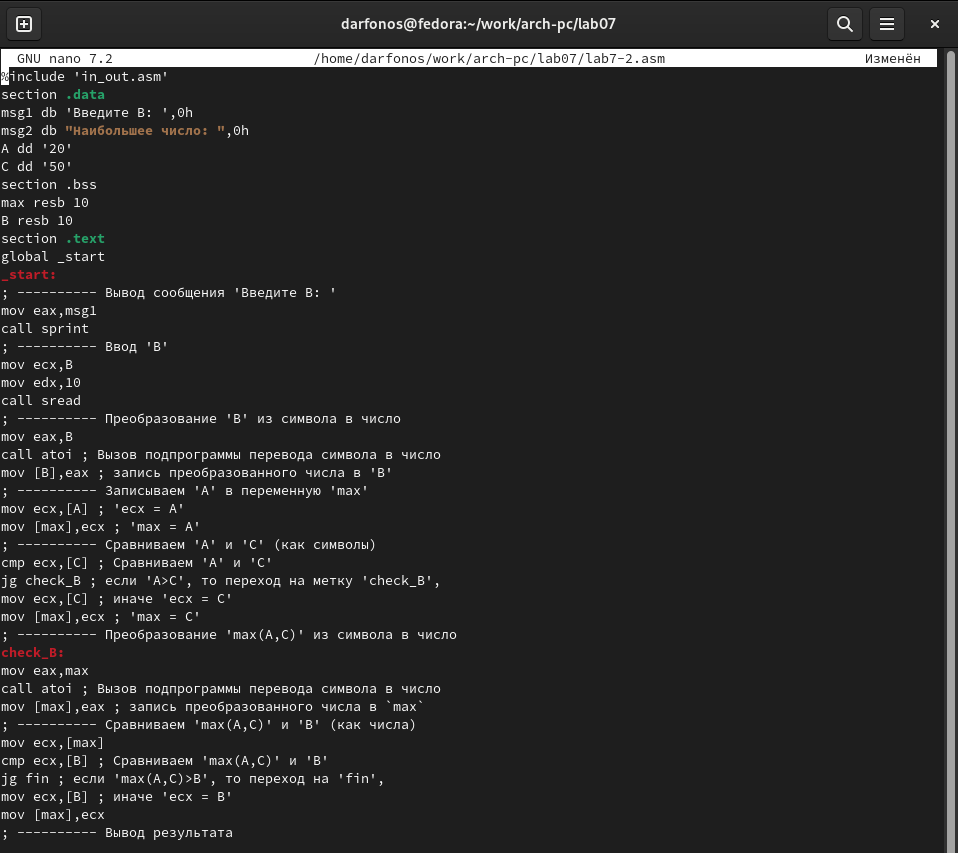


Figure 10: Вставляю текст в файл

**11**

Создаю и запускаю новый исполняемый файл, проверяю работу программы для разных B=40 и B=60, при А=20 и С=50 (рис. [[11](#fig:011)]).

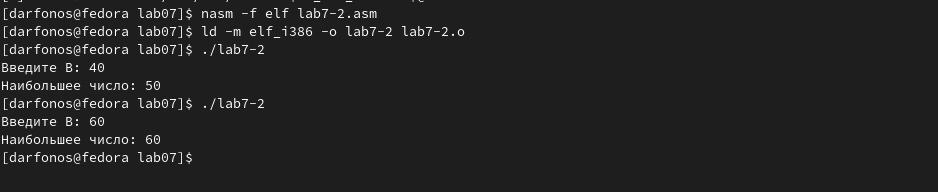


Figure 11: Запуск исполняемого файла

**12**

Создаю файл листинга для программы в файле lab7-2.asm (рис. [[12](#fig:012)]).

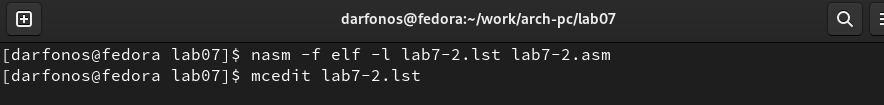


Figure 12: Редактирование файла

**13**

Открываю файл листинга с помощью редактора mcedit. Расмотрим 9-11 строки: (рис. [[13](#fig:013)]).

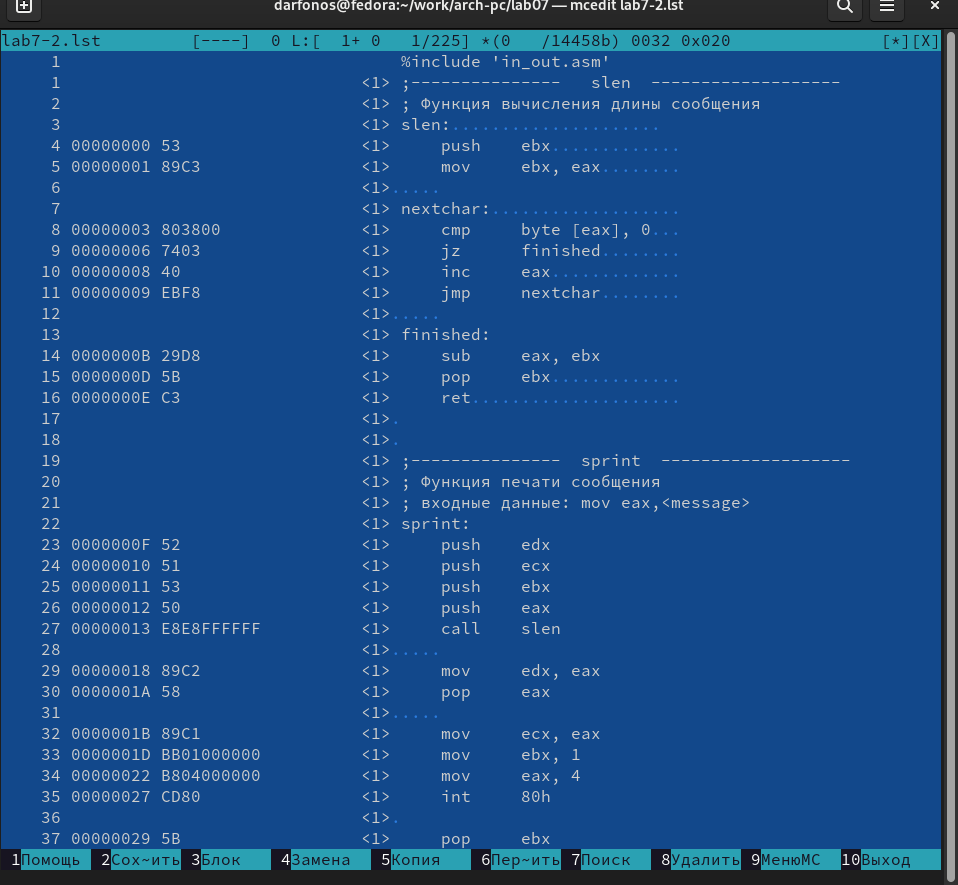


Figure 13: Файл листинга

9 строка:

* Перые цифры [9] - это номер строки файла листинга.
* Cледующие цифры [00000006] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел.
* следующие числа [7403] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтоу и появляются буквы латынского алфавита.
* следющее [jz finished] - исходный текст программы, которая просто состоит из строкк исходной программы вместе с комментариями.

10 строка:

* Перые цифры [10] - это номер строки файла листинга.
* Cледующие цифры [00000008] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел.
* следующие числа [40] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтоу и появляются буквы латынского алфавита.
* следющее [inc eax] - исходный текст программы, которая просто состоит из строкк исходной программы вместе с комментариями

11 строка:

* Перые цифры [11] - это номер строки файла листинга.
* Cледующие цифры [00000009] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел.
* следующие числа [EBF8] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтоу и появляются буквы латынского алфавита.
* следющее [jmp nextchar] - исходный текст программы, которая просто состоит из строкк исходной программы вместе с комментариями

**14**

Открываю файл lab7-2.asm с помощью редактора и Удаляю один операнд в инструкции cmp. (рис. [[14](#fig:014)]).

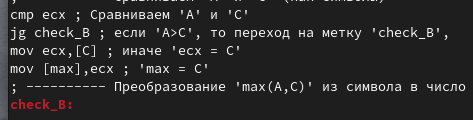


Figure 14: Файл листинга

**15**

Открываю файл листинга с помощью редактора mcedit и замечаю, что в файле листинга появляется ошибка. (рис. [[15](#fig:015)]).

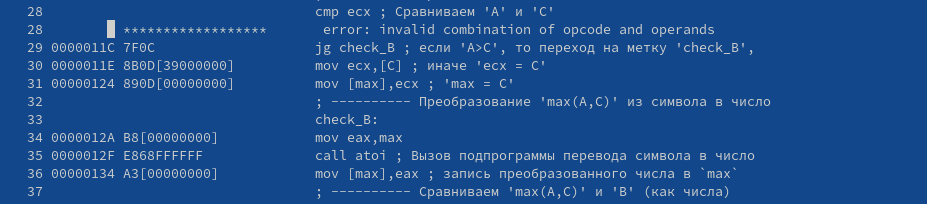


Figure 15: Файл листинга

Отсюда можно сделать вывод, что, если в коде появляется ошибка, то ее описание появится в файле листинга

# 3 Сомтоятельная работа

**1**

Создаю файл lab7-3.asm с помощью утилиты touch (рис. [[16](#fig:016)]).

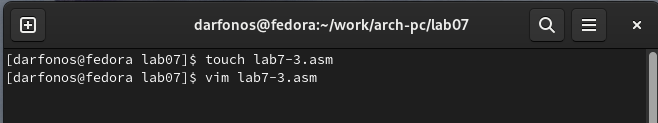


Figure 16: Создание файла

**2**

Ввожу в созданный файл текст программы для вычисления наибольшего из 3 чисел. Числа беру, учитывая свой вариант из прошлой лабораторной работы. 2 вариант (рис. [[17](#fig:017)]).

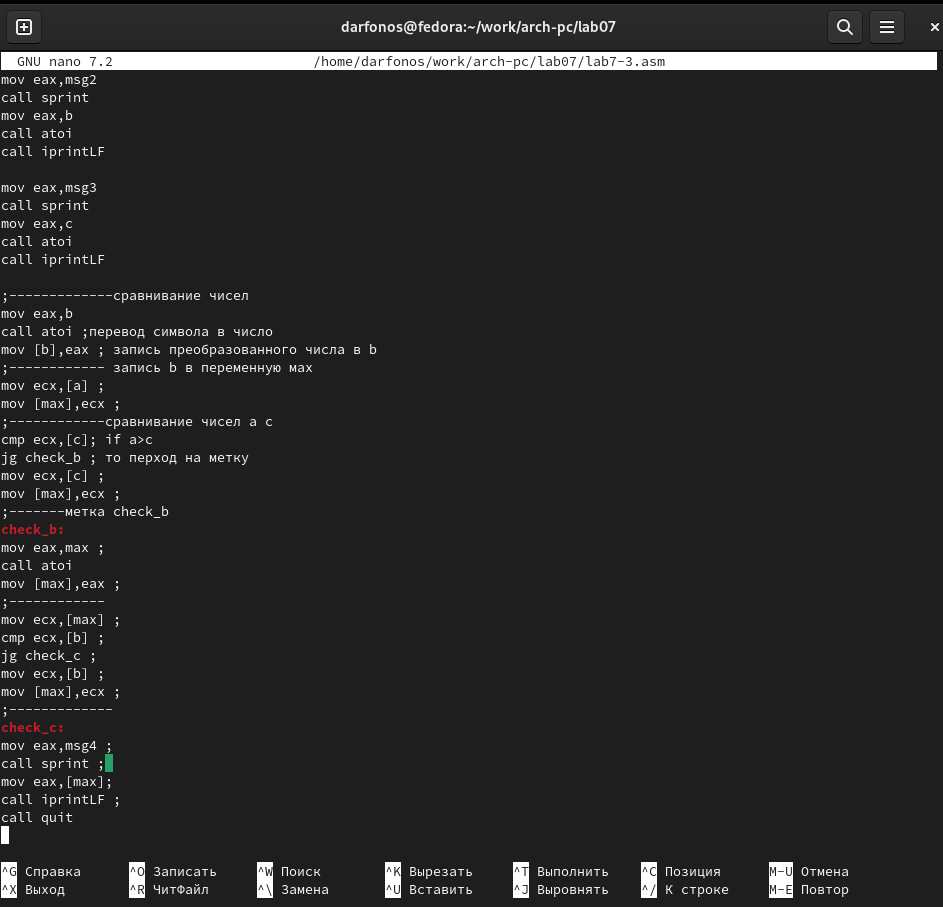


Figure 17: Редактирование файла

**3**

Создаю исполняемый файл и запускаю его (рис. [[18](#fig:018)]).

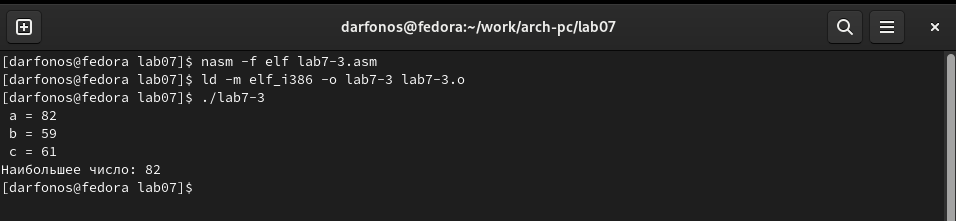


Figure 18: Запуск исполняемого файла

**Текст программы**

%include 'in\_out.asm'  
section .data  
msg1 db ' а = ',0h  
msg2 db ' b = ',0h  
msg3 db ' c = ',0h  
msg4 db "Наибольшее число: ",0h  
a dd '82'  
b dd '59'  
c dd '61'  
  
section .bss  
max resb 10  
  
section .text  
global \_start  
\_start:  
; ---------- Вывод всех чисел:   
mov eax,msg1  
call sprint  
mov eax,a  
call atoi  
call iprintLF  
  
mov eax,msg2  
call sprint  
mov eax,b  
call atoi  
call iprintLF  
  
mov eax,msg3  
call sprint  
mov eax,c  
call atoi  
call iprintLF  
  
;-------------сравнивание чисел  
mov eax,b  
call atoi ;перевод символа в число  
mov [b],eax ; запись преобразованного числа в b  
;------------ запись b в переменную мах  
mov ecx,[a] ;  
mov [max],ecx ;  
;------------сравнивание чисел a c  
cmp ecx,[c]; if a>c  
jg check\_b ; то перход на метку  
mov ecx,[c] ;  
mov [max],ecx ;  
;-------метка check\_b  
check\_b:  
mov eax,max ;  
call atoi  
mov [max],eax ;  
;------------  
mov ecx,[max] ;  
cmp ecx,[b] ;  
jg check\_c ;  
mov ecx,[b] ;  
mov [max],ecx ;  
;-------------  
check\_c:  
mov eax,msg4 ;  
call sprint ;   
mov eax,[max];  
call iprintLF ;  
call quit

**4**

Создаю новый файл lab7-4 для написания программы второго задания. (рис. [[19](#fig:019)]).

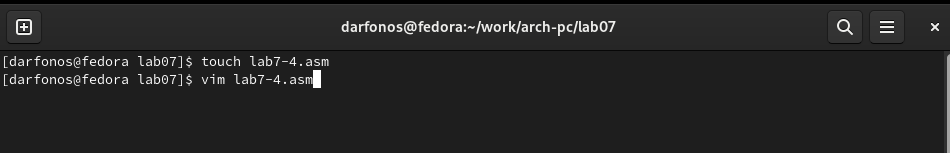


Figure 19: создание файла

**5**

Ввожу в него программу, (рис. [[21](#fig:020)]). в которую ввожу 2 значения x и a, и которая выводит значения функции. Функцию беру из таблицы в соответствии со своим вариантом (Вариант рис. [[20](#fig:023)]).



Figure 20: функция f(x)



Figure 21: ввод программы в файл

**6**

Создаю испольняемый файл и проверяю её выполнение при x=5, a=7 (рис. [[22](#fig:021)]). Программа отработала верно!

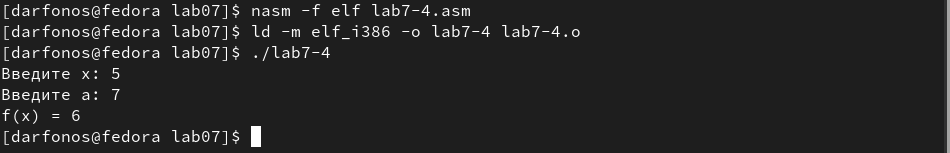


Figure 22: Создание исполняемого файла

**7**

Повторный раз запускаю программу и проверяю ее выполнение при x=6 и a=4 (рис. [[23](#fig:022)]). Программа отработала верно!



Figure 23: запуск исполняемого файла

**Текст программы**

%include 'in\_out.asm'  
section .data  
msg1 db 'Введите x: ',0h  
msg2 db 'Введите a: ',0h  
msg3 db 'f(x) = ',0h  
  
section .bss  
x resb 10  
a resb 10  
  
section .text  
global \_start  
\_start:  
mov eax,msg1  
call sprint  
mov ecx,x  
mov edx,10  
call sread  
mov eax,x  
;----------  
call atoi  
mov [x],eax  
;-----------  
  
mov eax,msg2  
call sprint  
mov ecx,a  
mov edx,10  
call sread  
mov eax,a ;  
call atoi  
mov [a],eax ;  
;----------  
mov ecx,[a]  
cmp ecx,[x] ;x<a  
jg check\_a ;  
mov ecx,[x]  
check\_a:  
add ecx,-1;  
mov eax,msg3 ;  
call sprint ;  
mov eax,ecx ;  
call iprintLF;  
call quit ;

# 4 Выводы

При выполнении данной лабораторной работы я освоил инструкции условного и безусловного вывода и ознакомился с структурой файла листинга.ы