

Отчет по лабораторной работе №5

Дисциплина: Архитектура компьютера

Чипурной Михаил Евгеньевич

Содержание

1 Цель работы	5
2 Выполнение лабораторной работы	6
3 Задание для самостоятельной работы	13
4 Выводы	16

Список иллюстраций

2.1	Открываем с помощью команды mc	6
2.2	Переходим в каталог	7
2.3	Создаем каталог функциональной клавишей	7
2.4	Создаем файл с помощью команды touch	8
2.5	Открывем файл функциональной клавишей, заполняем и сохраняем	8
2.6	Открываем файл и убеждаемся, что файл содержит текст программы	9
2.7	Проверяем, как работает данная программа	9
2.8	Скачиваем файл	9
2.9	Копируем скаченный файл	10
2.10	Создаем копию файла клавишей с помощью функциональной клавиши и проверяем	10
2.11	Открывем файл функциональной клавишей, заполняем и сохраняем	11
2.12	Смотрим, как сработала программа	11
2.13	Редактируем файл и сохраняем	12
2.14	Смотрим, как сработал программа и сравниваем с прошлой	12
3.1	Создаем копию файла с помощью функциональной клавиши	13
3.2	Редактируем файл	14
3.3	Смотрим, как сработал программа	14
3.4	Создаем копию файла с помощью функциональной клавиши	14
3.5	Редактируем файл	15
3.6	Смотрим, как сработал программа	15

Список таблиц

1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера mov и int.

2 Выполнение лабораторной работы

Открываем Midnight Commander (Рисунок 2.1).

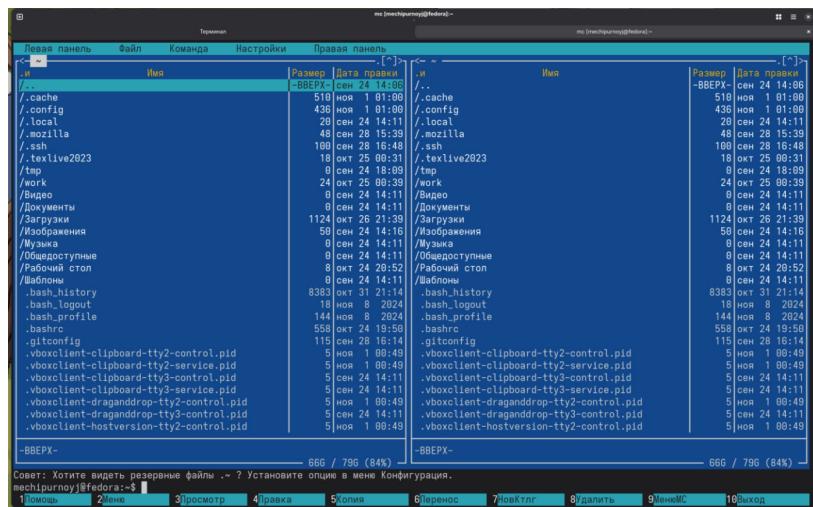


Рисунок 2.1: Открываем с помощью команды mc

Переходим в каталог, созданный при выполнении 4 ЛБ (Рисунок 2.2).

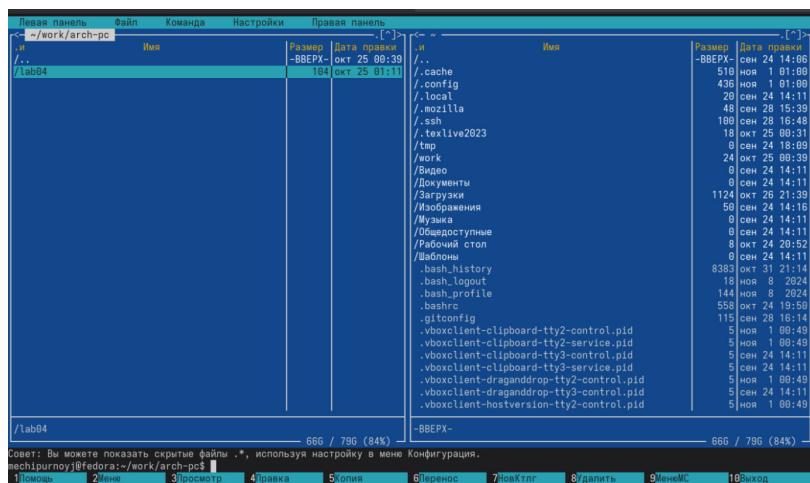


Рисунок 2.2: Переходим в каталог

Создаем каталог lab05 (Рисунок 2.3).

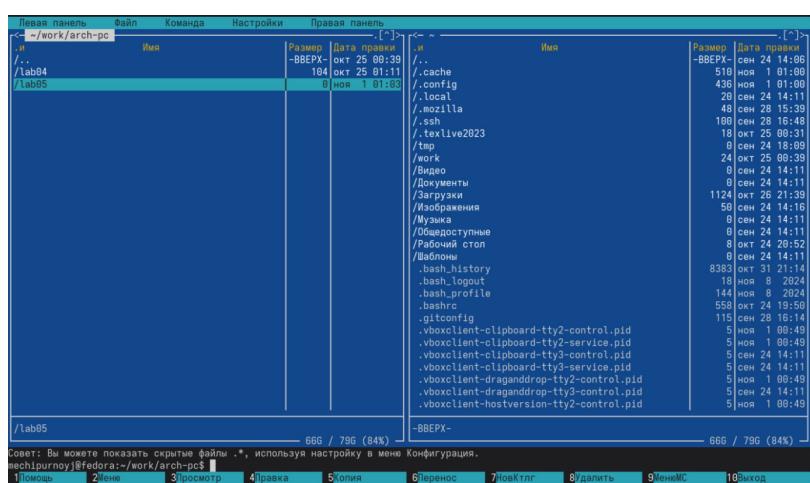


Рисунок 2.3: Создаем каталог функциональной клавишей

Создаем файл lab5-1.asm (Рисунок 2.4).

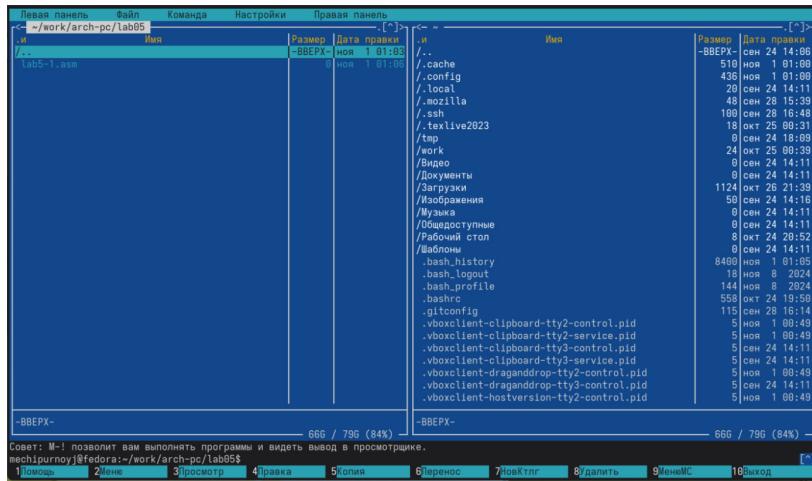


Рисунок 2.4: Создаем файл с помощью команды touch

Открываем файл для редактирования и заполняем его по листингу (Рисунок 2.5).

```
lab05-1.asm [M-] 40 L: 1=19 20/ 36 */1359/2432b) 0910 Вт094
; Программа вывода сообщения на экран и ввода строки с клавиатуры
; ----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плеск
msglen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
; ----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
    ;----- Вывод сообщения -----
    ; После вызова инструкции 'Int 80h' на экран будет
    ; выведено сообщение из переменной 'msg' длиной 'msglen'
    mov eax,4 ; Исполнитель файла 1 - стандартный вывод
    mov ebx,1 ; Адрес строки 'msg' в 'eax'
    mov edx,msglen ; Размер строки 'msg' в 'edx'
    int 80h ; Вызов ядра
    ;----- Чтение системной язва 'read' -----
    ; После вызова инструкции 'Int 80h' программа будет ожидать ввода
    ; строки, которая будет записана в переменную 'buf1' размером 80 байт
    mov eax,3 ; Системный вызов для чтения (sys.read)
    mov ebx,0 ; Дескриптор файла 0 - стандартный ввод
    mov ecx,buf1 ; Адрес буфера под вводимую строку
    mov edx,80 ; Длина вводимой строки
    int 80h ; Вызов ядра
    ;----- Выполнение системной язва 'exit' -----
    ; После вызова инструкции 'Int 80h' программа завершит работу
    mov eax,1 ; Системный вызов для выхода (sys.exit)
    mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
    int 80h ; Вызов ядра
```

Рисунок 2.5: Открываем файл функциональной клавишей, заполняем и сохраняем

Открываем файл для просмотра (Рисунок 2.6).

```

/home/mechipurnoyj/work/arch-pc/lab05/lab5-1.asm
2432/2432 100%
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку!',10 ; сообщение плюс
; символ перевода строки
msgLen EQU $-msg ; Длина переменной 'msg'
SECTION .text ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов 'write' -----
; После вызова инструкция 'int 80h' на экран будет
; выведено содержимое переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys.write)
mov ebx,1 ; Адрес строки 'msg' в 'eax'
mov edx,msg ; Размер строки 'msg' в 'edx'
int 80h ; Вызов ядра
;----- системный вызов 'read' -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт
mov eax,3 ; Системный вызов для чтения (sys.read)
mov ebx,0 ; Дескриптор файла 0 - стандартный ввод
mov edx,buf1 ; Адрес буфера под вводимую строку
mov edx,80 ; Длина вводимой строки
int 80h ; Вызов ядра
;----- системный вызов 'exit' -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys.exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра

```

Рисунок 2.6: Открываем файл и убеждаемся, что файл содержит текст программы

Транслируем текст программы и запускаем исполняемый файл (Рисунок 2.7).

```

mechipurnoyj@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Чипурной Михаил Евгеньевич
mechipurnoyj@fedora:~/work/arch-pc/lab05$

```

Рисунок 2.7: Проверяем, как работает данная программа

Скачиваем файл со страницы курса (Рисунок 2.8).



Рисунок 2.8: Скачиваем файл

Копируем файл в нужную директорию (Рисунок 2.9).

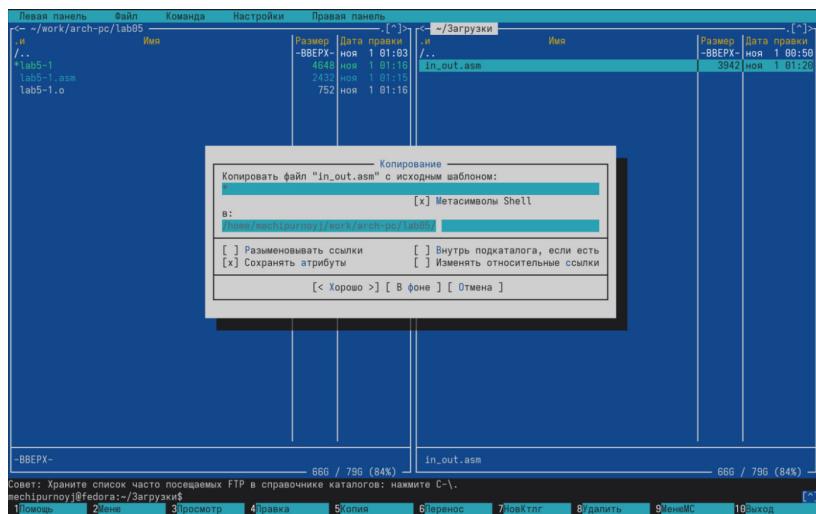


Рисунок 2.9: Копируем скачанный файл

Создаем копию файла lab5-1.asm и называем lab5-2.asm (Рисунок 2.10).

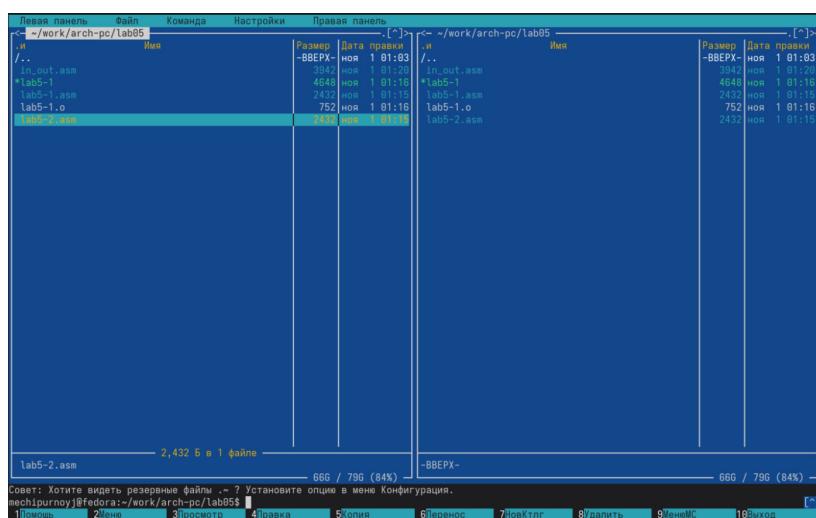


Рисунок 2.10: Создаем копию файла клавишей с помощью функциональной клавиши и проверяем

Открываем новый файл и заполняем его в соответствии с листингом (Рисунок 2.11).

```
lab5-2.asm [-M--] 0 L:[ 1+18 19/ 19] *(1228/1228b) <EOF>
;
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .rsrc ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Точка входа в программу
_start: ; Точка входа в программу
    mov eax, msg ; запись адреса выодимого сообщения в 'EAX'
    call sprintf ; вызов подпрограммы печати сообщения
    mov ecx, buf1 ; запись адреса переменной в 'ECX'
    mov edx, 80 ; запись длины вводимого сообщения в 'EBX$'
    call read ; вызов подпрограммы ввода сообщения
    call quit ; вызов подпрограммы завершения
```

Рисунок 2.11: Открываем файл функциональной клавишей, заполняем и сохраняем

Транслируем и запускаем новый файл (Рисунок 2.12).

```
mechipurnoyj@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Чипурной Михаил Евгеньевич
mechipurnoyj@fedora:~/work/arch-pc/lab05$
```

Рисунок 2.12: Смотрим, как сработала программа

Снова открываем файл для редактирования и меняем sprintLF на sprint (Рисунок 2.13).

```
lab5-2.asm      [-M--] 11 L:[ 1+12 13/ 19 ] *(847 /1226b) 8832 0x020
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----[include 'in_out.asm'; подключение внешнего файла-----]
SECTION .data ; Секция инициализированных данных
msg:  db 'Введите строку: ',0h ; сообщение
buff: RESB 80 ; буфер размером 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
    mov eax, msg ; запись адреса выводимого сообщения в 'EAX'
    call sprint ; вызов подпрограммы печати сообщения
    mov ecx, buff ; запись адреса переменной в 'EAX'
    mov edx, 80 ; запись длины выводимого сообщения в 'EBX56'
    call read ; вызов подпрограммы ввода сообщения
    call quit ; вызов подпрограммы завершения
```

Рисунок 2.13: Редактируем файл и сохраняем

Транслируем и запускаем файл (Рисунок 2.14).

```
mechipurnoyj@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Чипурной Михаил Евгеньевич
mechipurnoyj@fedora:~/work/arch-pc/lab05$ █
```

Рисунок 2.14: Смотрим, как сработал программа и сравниваем с прошлой

Таким образом, команда `sprint` выводит текст в той же строке, а `sprintLF` выполняет переход на новую строку.

3 Задание для самостоятельной работы

Создаем копию файла lab5-1.asm (Рисунок 3.1).

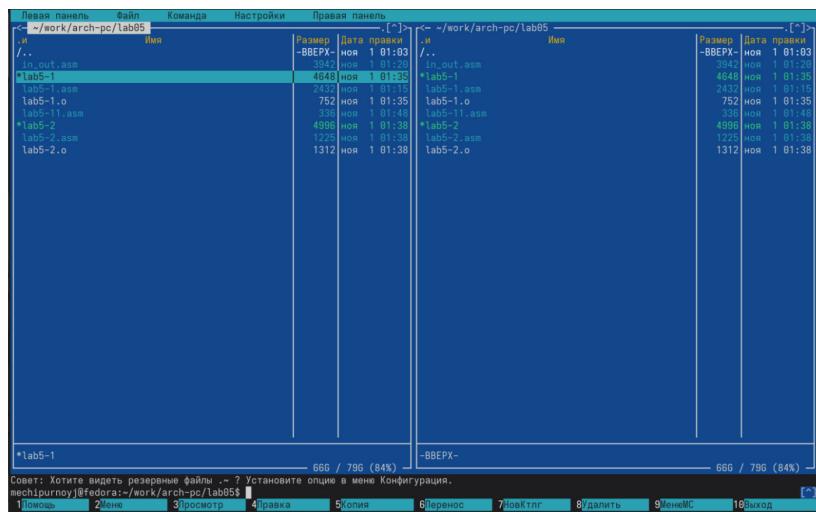


Рисунок 3.1: Создаем копию файла с помощью функциональной клавиши

Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль (Рисунок 3.2).

```

lab5-11.asm [-M--] 8 L:[ 1+ 2 3/ 31 ] *(56 / 3365) 0109 0x0000
SECTION .data
msg: DB 'Введите строку:',10
msglen: EQU $-msg
SECTION .text
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
    mov eax,4
    mov ebx,1
    mov ecx,msg
    mov edx,msgLen
    int 80h
    mov eax,3
    mov ebx,0
    mov ecx,buf1
    mov edx,80
    int 80h
    mov eax,4
    mov ebx,1
    mov ecx,buf1
    mov edx,80
    int 80h
    mov eax,1
    mov ebx,0
    int 80h

```

1 Помощь 2 Сохранить 3 Блок 4 Замена 5 Копия 6 Переносить 7 Поиск 8 Удалить 9 МенюМС 10 Выход

Рисунок 3.2: Редактируем файл

Транслируем файл и запускаем программу (Рисунок 3.3).

```

mechipurnoyj@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-11.asm
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-11 lab5-11.o
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ./lab5-11
Введите строку:
Чипурной Михаил Евгеньевич
Чипурной Михаил Евгеньевич
mechipurnoyj@fedora:~/work/arch-pc/lab05$ 

```

Рисунок 3.3: Смотрим, как сработал программа

Создаем копию файла lab5-2.asm (Рисунок 3.4).

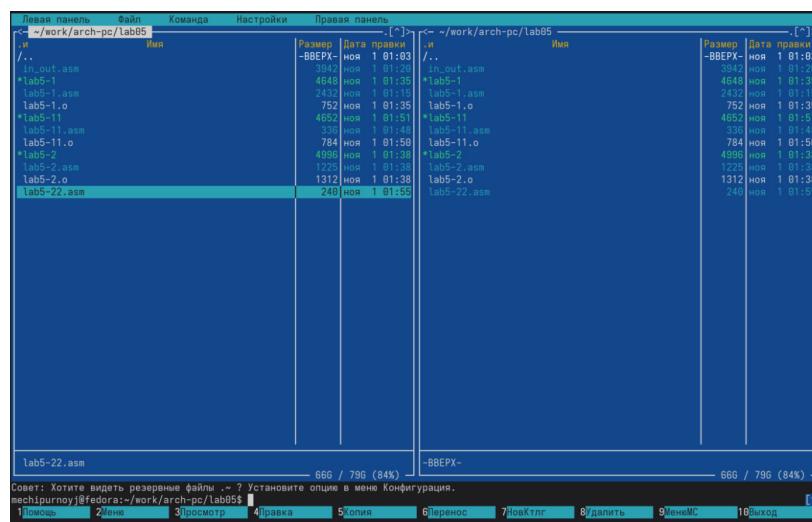
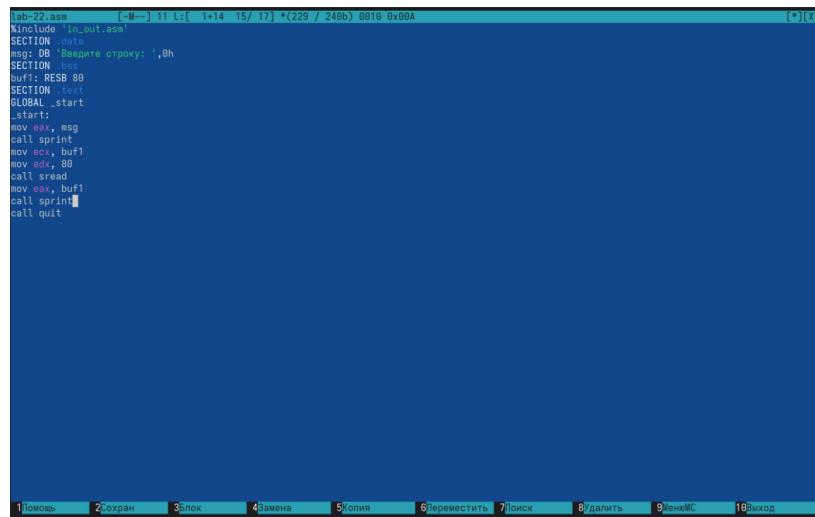


Рисунок 3.4: Создаем копию файла с помощью функциональной клавиши

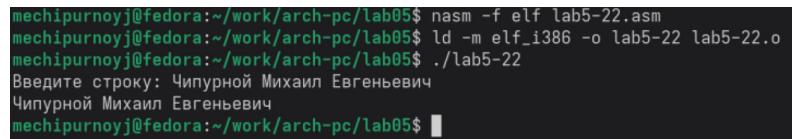
Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль
(Рисунок 3.5).



```
lab5-22.asm [M-] 11 L:[ 1x14 15/ 17 ] *(229 / 2400) 0010 0x00A
#include "in_out.asm"
SECTION .data
msg: DB 'Введите строку: ',0h
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start
_start:
    mov ax, msg
    mov es, msg
    call sprint
    mov ecx, buf1
    mov edx, 80
    call sread
    mov eax, buf1
    call sprint
    call quit
```

Рисунок 3.5: Редактируем файл

Транслируем файл и запускаем программу (Рисунок 3.6).



```
mechipurnoyj@fedora:~/work/arch-pc/lab05$ nasm -f elf lab5-22.asm
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-22 lab5-22.o
mechipurnoyj@fedora:~/work/arch-pc/lab05$ ./lab5-22
Введите строку: Чипурной Михаил Евгеньевич
Чипурной Михаил Евгеньевич
mechipurnoyj@fedora:~/work/arch-pc/lab05$
```

Рисунок 3.6: Смотрим, как сработал программа

4 Выводы

Мы приобрели практические навыки работы в файловом менеджере Midnight Commander, а также освоили инструкции языка ассемблера mov и int.