# Лабораторная работа 5

## Отчёт

Соловьев Богдан Михайлович

Содержание

## 1 Цель работы

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

## 2 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной. Для активации оболочки Midnight Commander достаточно ввести в командной строке mc и нажать клавишу Enter (рис. 5.1). В Midnight Commander используются функциональные клавиши F1 — F10 , к которым привязаны часто выполняемые операции.

Архитектура ЭВМ Функцио- нальные клавиши Выполняемое действие F5 копирование файла или группы отмеченных файлов из каталога, отображаемого в активной панели, в каталог, отображаемый на второй панели F6 перенос файла или группы отмеченных файлов из каталога, отображаемого в активной панели, в каталог, отображаемый на второй панели F7 создание подкаталога в каталоге, отображаемом в активной панели F8 удаление файла (подкаталога) или группы отмеченных файлов F9 вызов основного меню программы F10 выход из программы

Следующие комбинации клавиш облегчают работу с Midnight Commander: • Tab используется для переключениями между панелями; • ↑ и ↓ используется для навигации, Enter для входа в каталог или открытия файла (если в файле расширений mc.ext заданы правила связи определённых расширений файлов с инструментами их запуска или обработки); • Ctrl + u (или через меню Команда > Переставить панели ) меняет местами содержимое правой и левой панелей; • Ctrl + o (или через меню Команда > Отключить панели ) скрывает или возвращает панели Midnight Commander, за которыми доступен

для работы командный интерпретатор оболочки и выводимая туда информация. • Ctrl + x + d (или через меню Команда > Сравнить каталоги ) позволяет сравнить содер- жимое каталогов, отображаемых на левой и правой панелях.

Программа на языке ассемблера NASM, как правило, состоит из трёх секций: секция кода программы (SECTION .text), секция инициированных (известных во время компиляции) данных (SECTION .data) и секция неинициализированных данных (тех, под которые во время компиляции только отводится память, а значение присваивается в ходе выполнения программы) (SECTION .bss). Таким образом, общая структура программы имеет следующий вид:

```
SECTION .data ; Секция содержит переменные, для
... ; которых задано начальное значение
```

# 3 Выполнение лабораторной работы

## 3.1 Лабораторная работа

Открываю Midnight Commabnder (предварительно скачав библиотеку) написав в командной строке mc (рис. 1).

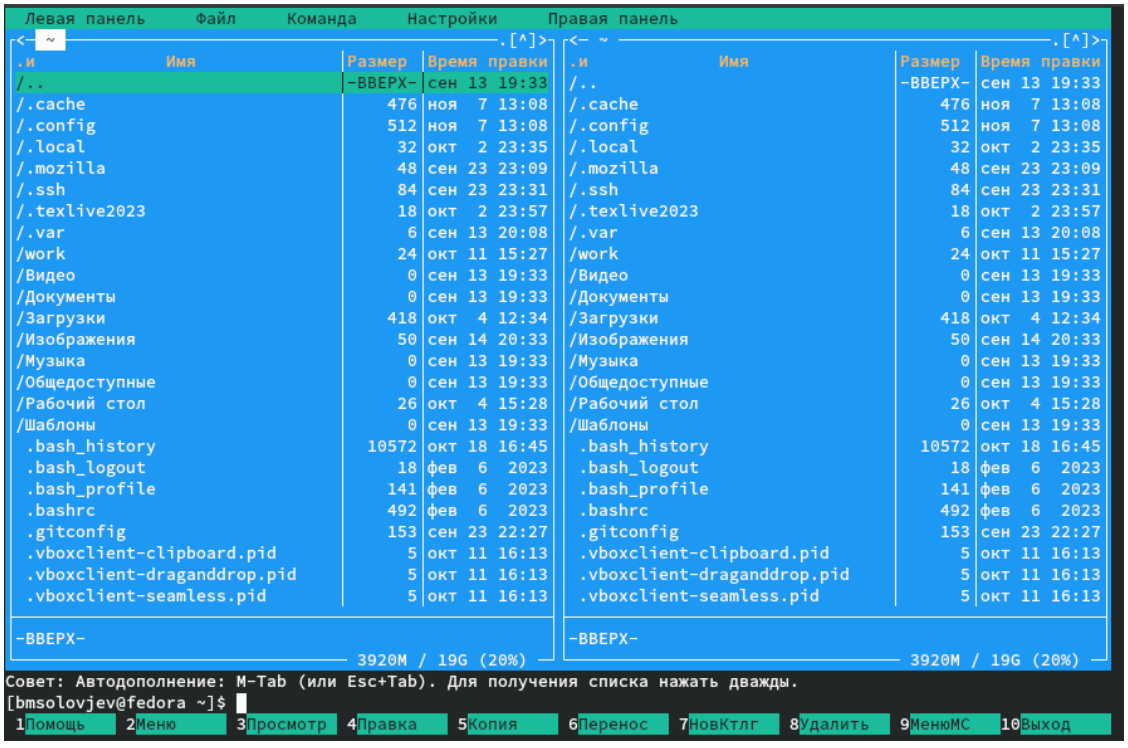


*Figure 1: Внешний вид mc*

Создаю каталог lab05 с помощью встроенных функций mc(рис. 2).

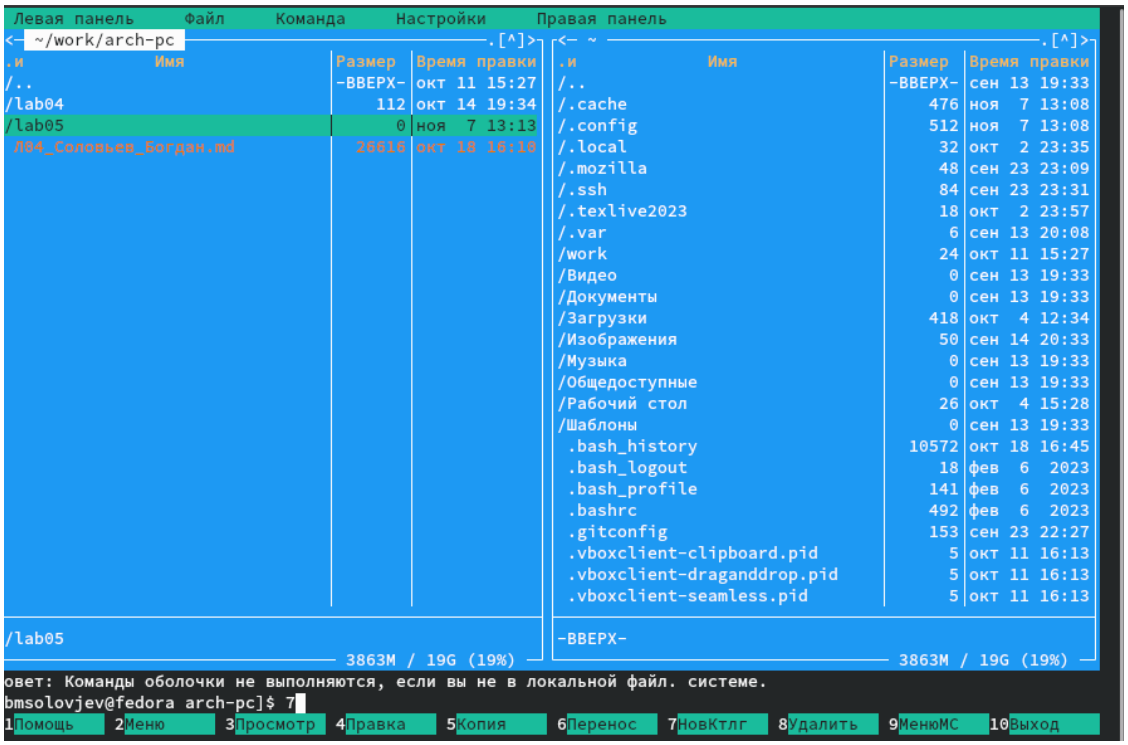


*Figure 2: Папка lab05*

Ввожу в созданных файл код программы, которая будет запрашивать текст (рис. 3).

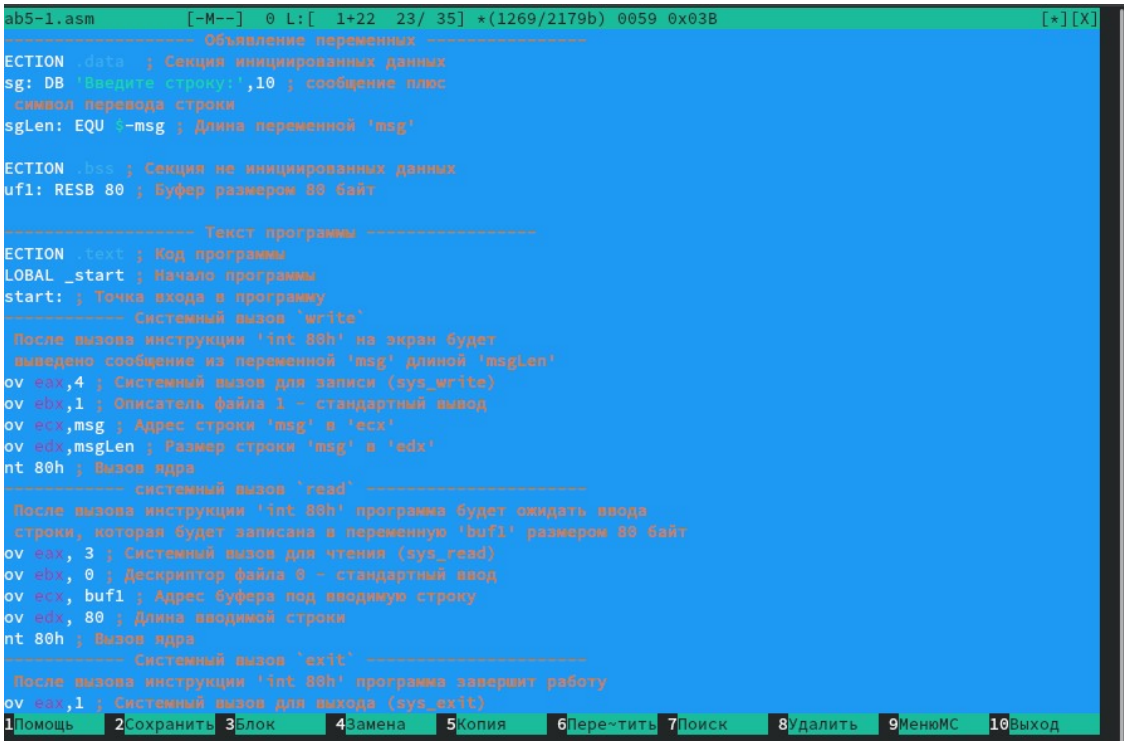


*Figure 3: Код программы*

Создаю объектный и сполняемый файл с помощью -f elf lab5-1.asm и lab5-1.o (рис. 4).

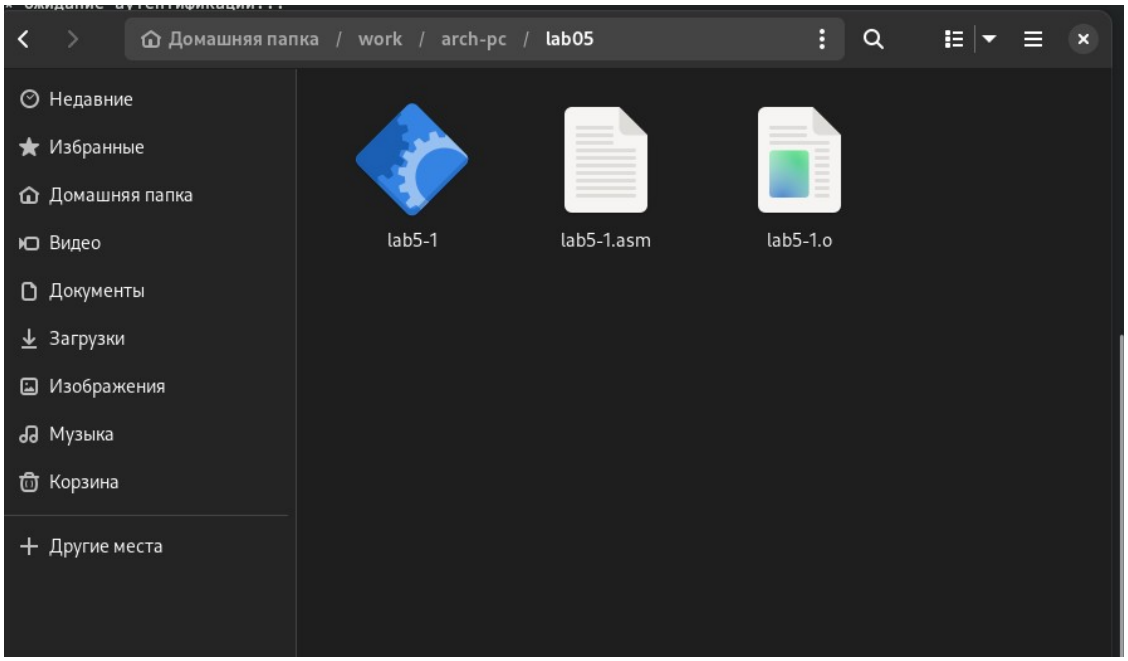


*Figure 4: Объектный и исполняемый файл*

Запускаю исполняемый файл (рис. 5).


*Figure 5: Программа запрашивает строку*

Скачиваю файл in_out.asm и переношу его в каталог lab05 (рис. 6).

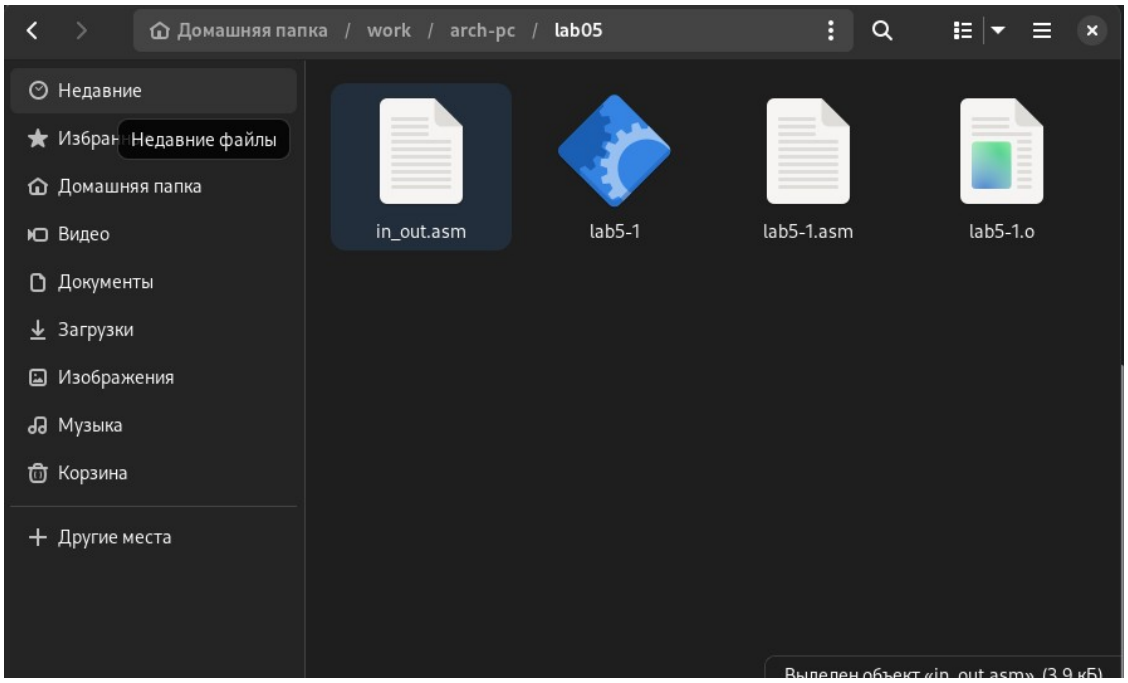


*Figure 6: Файл in_out.asm*

Копирую файл lab5-1.asm и изменяю его название (рис. 7).

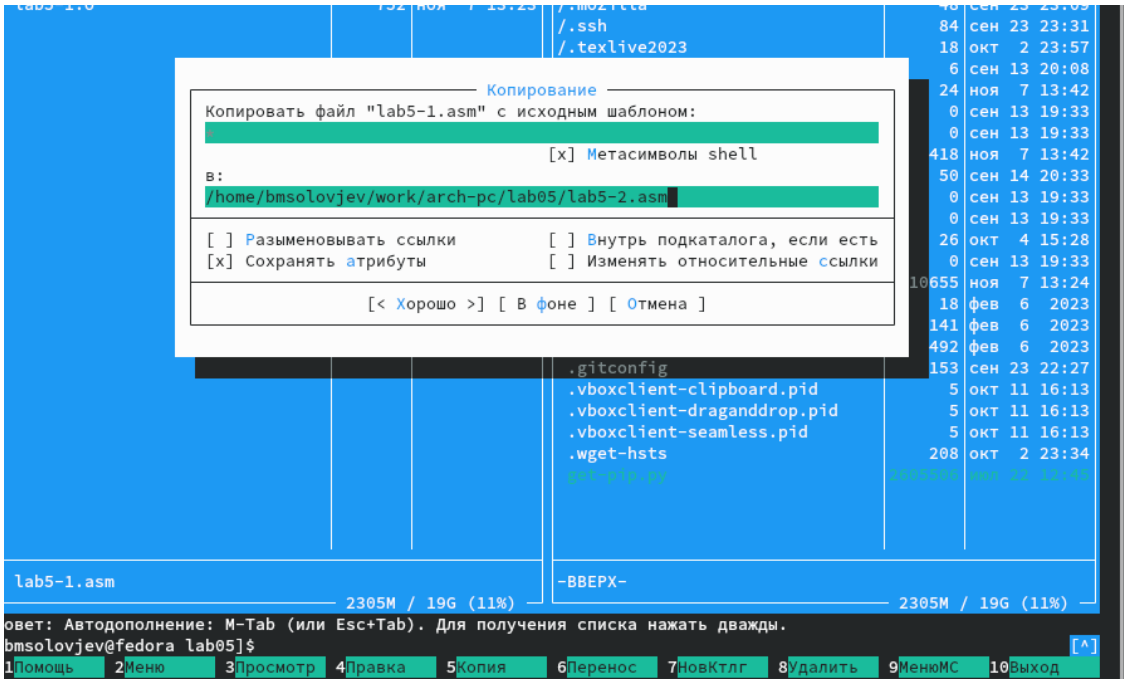


*Figure 7: Копирование и переименовывание файла*

Изменяю текст программы lab5-2.asm (рис. 8).

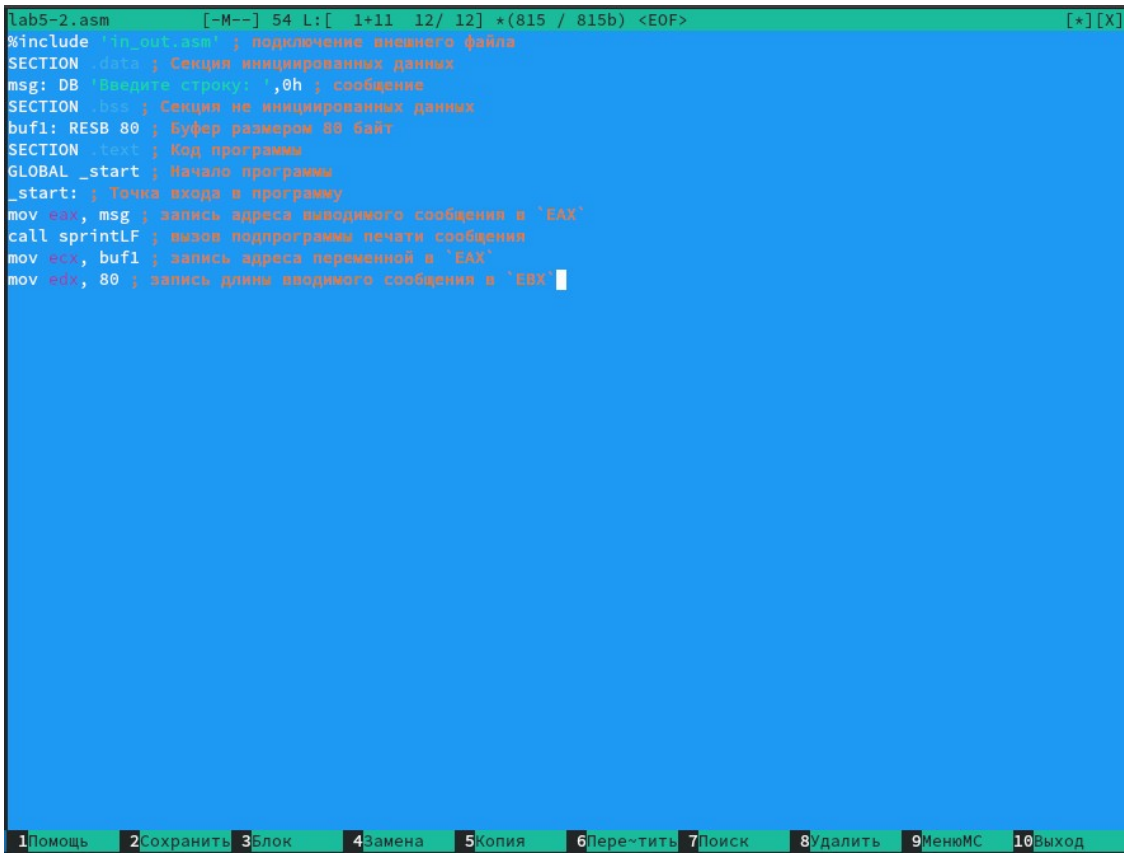


*Figure 8: Новый код программы*

Создание объектного и исполняемого файла и проверка работы команды (рис. 9).


*Figure 9: Создание и проверка*

Изменяю текст программы, прописывая sprint вместо sprintLF(рис. 10).

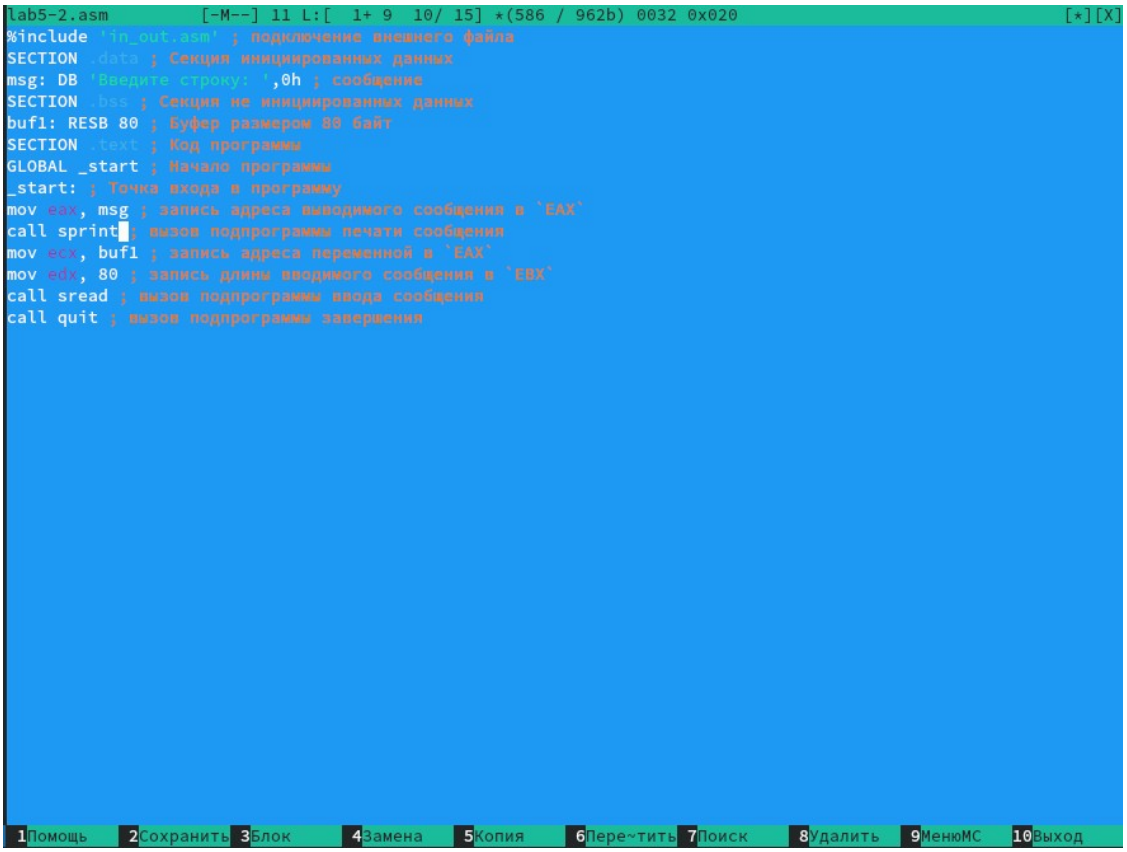


*Figure 10: Замена sprintLF на sprint*

Повторяю свои действия в третий раз (рис. 11).

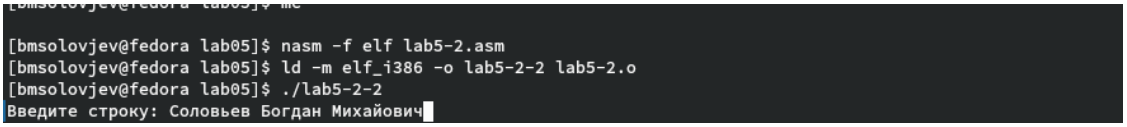


*Figure 11: Программа больше не переносит строку*

## 3.2 Самостоятельная работа

Создаю файл lab5-1-1.asm и вставляю в него новый код (рис. 12).

```
 GNU nano 7.2                  /home/bmsolovjev/work/arch-pc/lab05/lab5-1-1.asm                  Изменён

SECTION .bss ; Секция не инициированных 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 инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла 1 - стандартный вывод
mov ecx,msg ; Адрес строки 'msg' в 'ecx'
mov edx,msgLen ; 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 ecx, 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,4 ; Системный вызов для выхода (sys_exit)
mov ebx,1 ;
mov ecx,buf1 ;
mov edx,buf1 ;
int 80h ;
mov eax,1 ;
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра


^G Справка     ^O Записать    ^W Поиск       ^K Вырезать    ^T Выполнить   ^C Позиция     M-U Отмена
^X Выход       ^R ЧитФайл     ^\ Замена      ^U Вставить    ^J Выровнять   ^/ К строке    M-E Повтор
```

*Figure 12: Новый код файла lab5-1-1.asm*

Повторяю свои действия в четвёртый раз (рис. 13).

```
[bmsolovjev@fedora lab05]$ nasm -f elf lab5-1-1.asm
[bmsolovjev@fedora lab05]$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
[bmsolovjev@fedora lab05]$ ./lab5-1-1
Введите строку:
Соловьев Богдан
Соловьев Богдан
[bmsolovjev@fedora lab05]$
```

*Figure 13: Теперь программа пишет введённый текст*

Создаю файл lab5-2-1.asm и вставляю в него новый код (рис. 14).

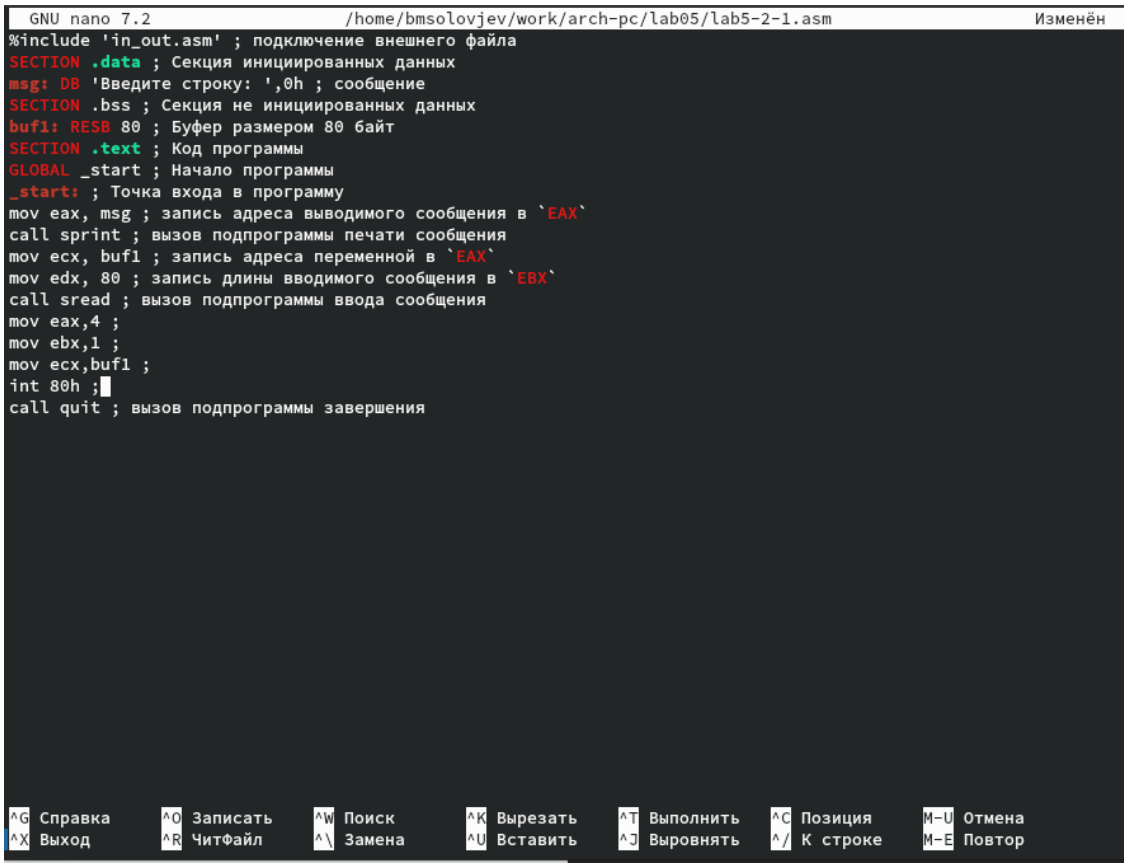


*Figure 14: Новый код файла lab5-2-1.asm*

Повторяю свои действия в пятый раз (рис. 15).

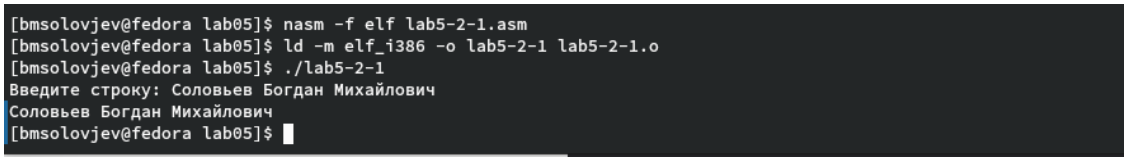


*Figure 15: Теперь программа пишет введённый текст и не переносит строку*

## 4 Выводы

При выполнении лабораторной работы, я приобрёл навыки работы в Midnight Commander, а также узнал принцип работы mov, int, print, printLF на языке программирования NASM.

## Список литературы