Лабораторная работа №5  
  
Midnight Commander

Архитектура компьютера и Операционные системы

Ван Сихэм Франклин О Нил Джон

Содержание

# 1 Цель работы

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

# 2 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. В Midnight Commander используются функциональные клавиши F1 — F10, к которым привязаны часто выполняемые операции.



Midnight Commander

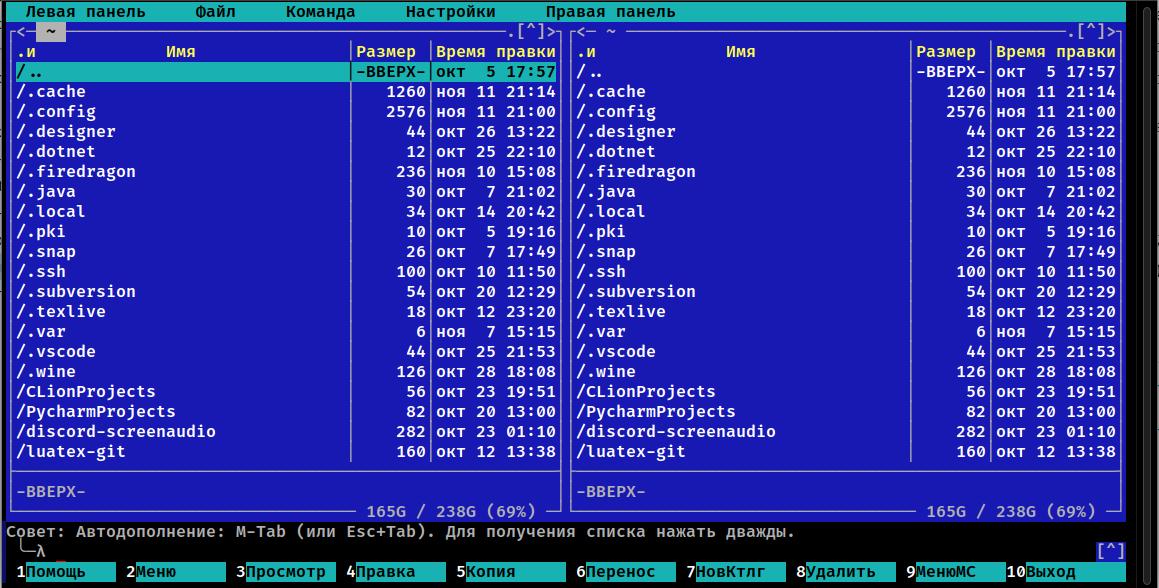
# 3 Порядок выполнения лабораторной работы

1. Откройте Midnight Commander

print(mishanya4u@Legenda in ~ via C v13.2.1-gcc)  
>> mc

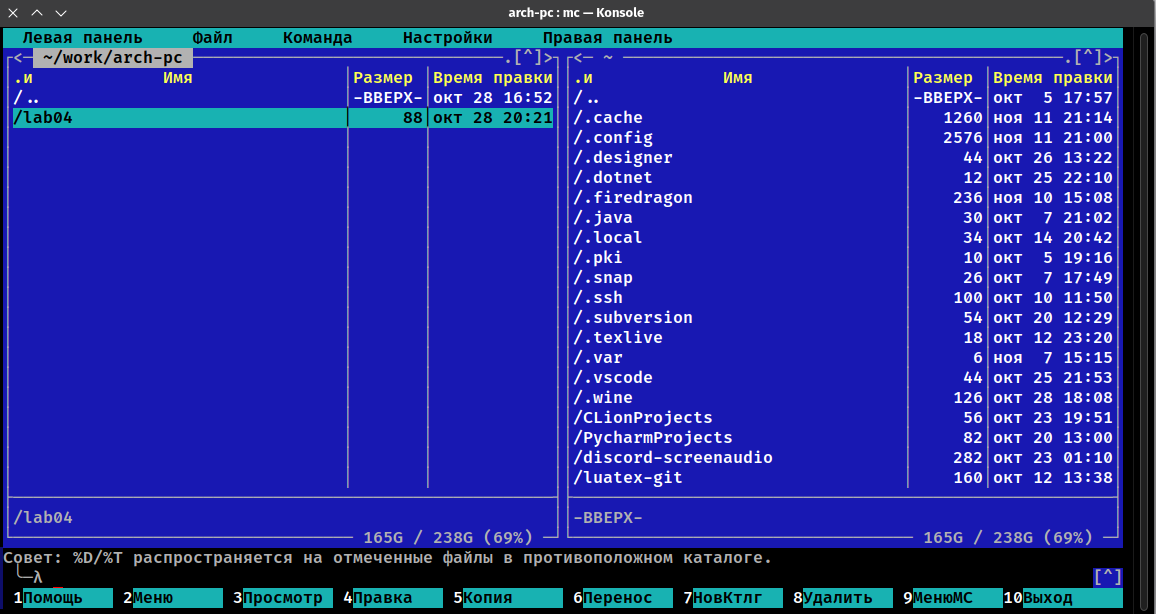


команда чтобы открыть Midnight Commander



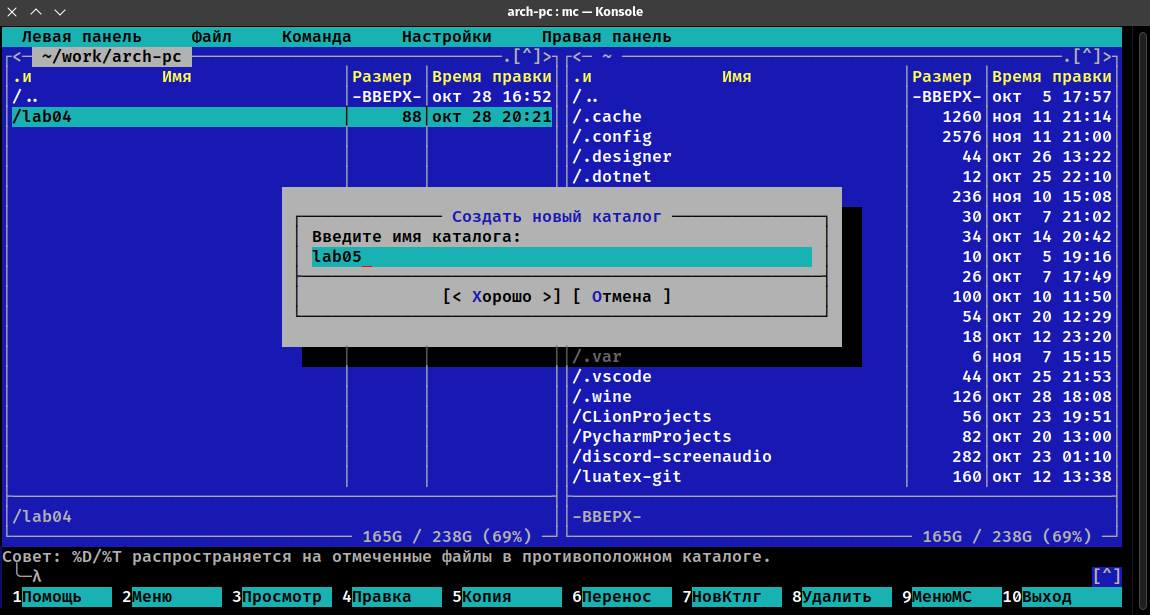
Редактор Midnight Commander

1. Пользуясь клавишами ↑ , ↓ и Enter перейдите в каталог ~/work/arch-pc созданный при выполнении лабораторной работы №4.

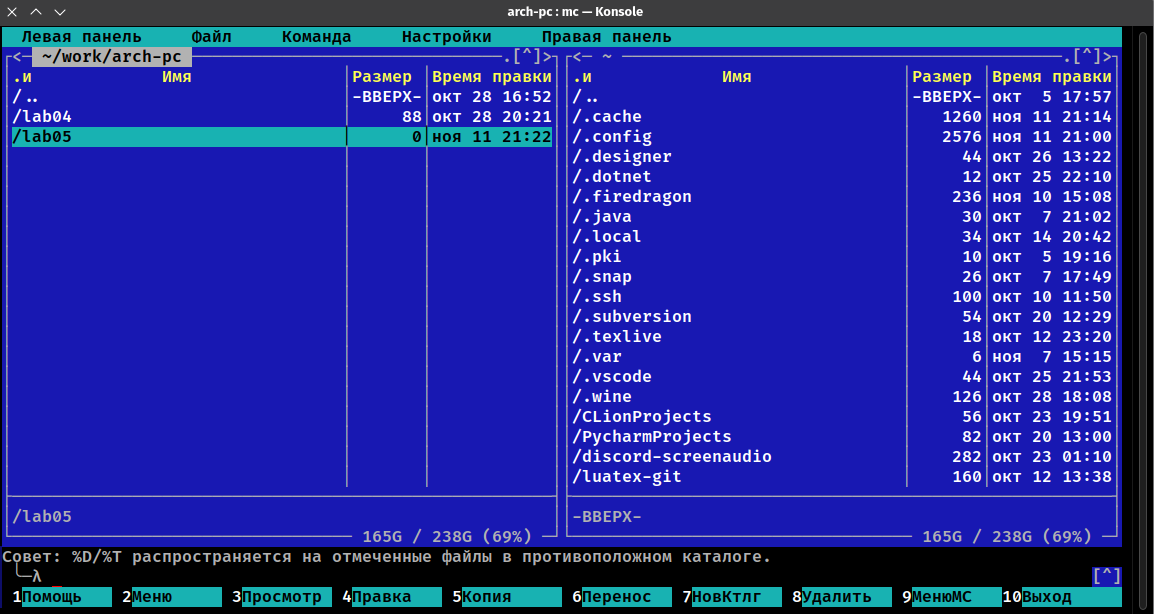


Перешёл в каталол ~/work/arch-pc

1. С помощью функциональной клавиши F7 создайте папку lab05 и перейдите в созданный каталог.

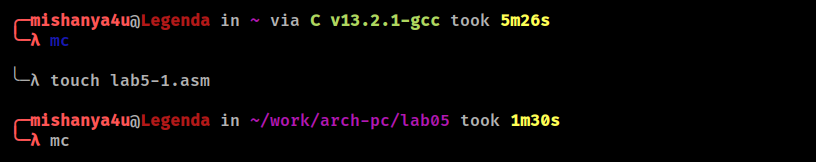


Создаем lab05

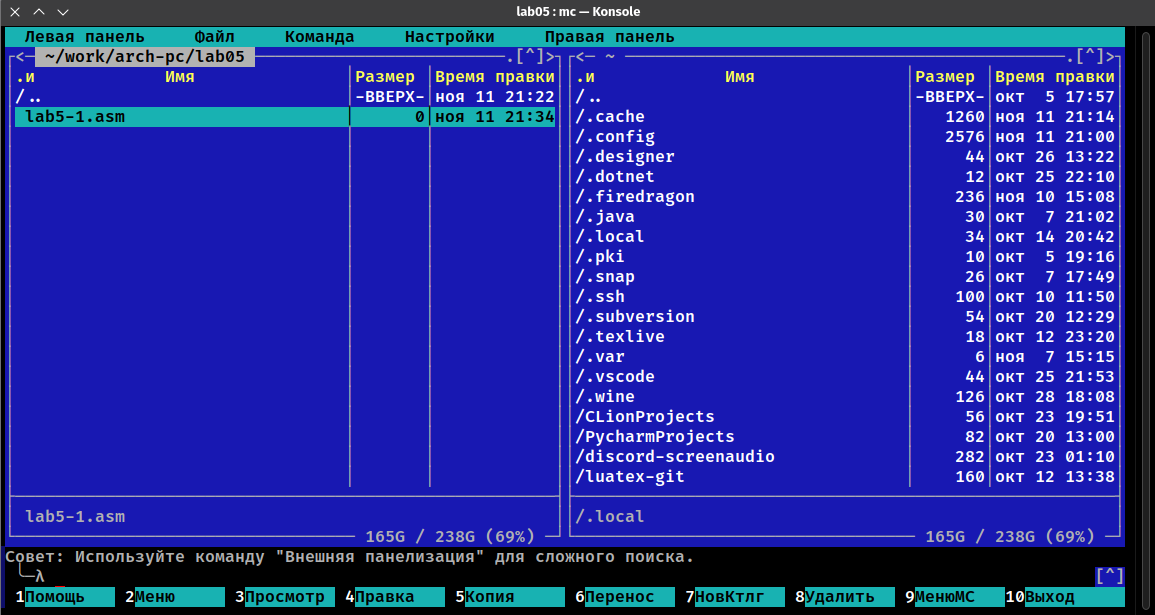


Создан lab05

1. Пользуясь строкой ввода и командой touch создайте файл lab5-1.asm.



Создаем lab5-1.asm с помощью touch



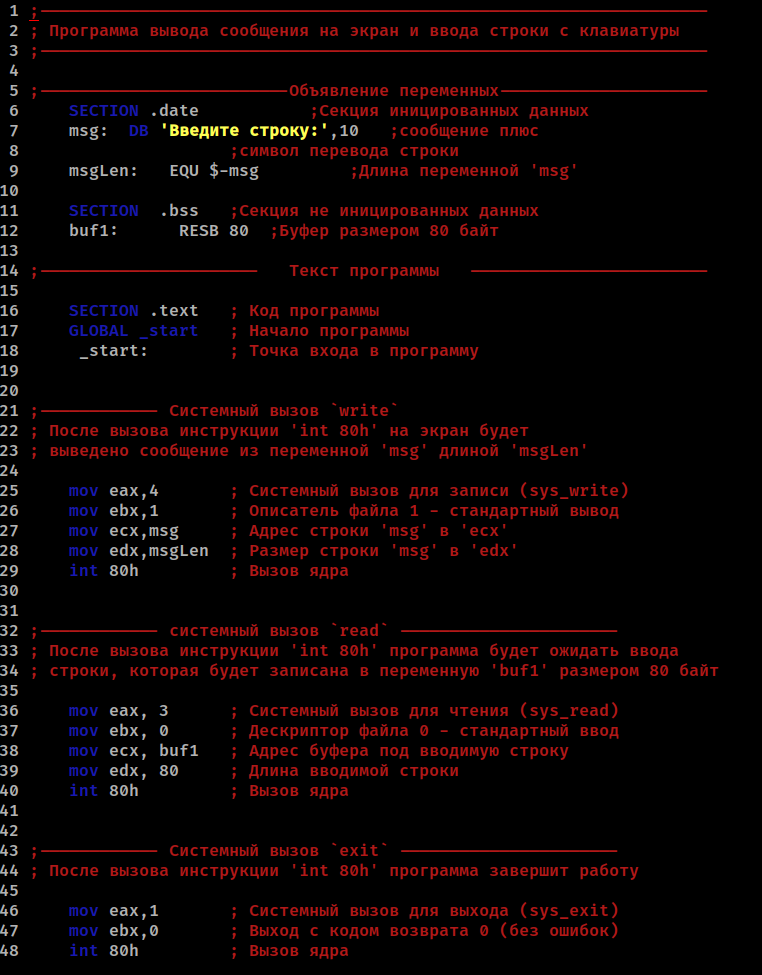
Создан lab05-1.asm

1. С помощью функциональной клавиши F4 откройте файл lab5-1.asm для редактирования во встроенном редакторе. Как правило в качестве встроенного редактора Midnight Commander используется редактор mcedit.

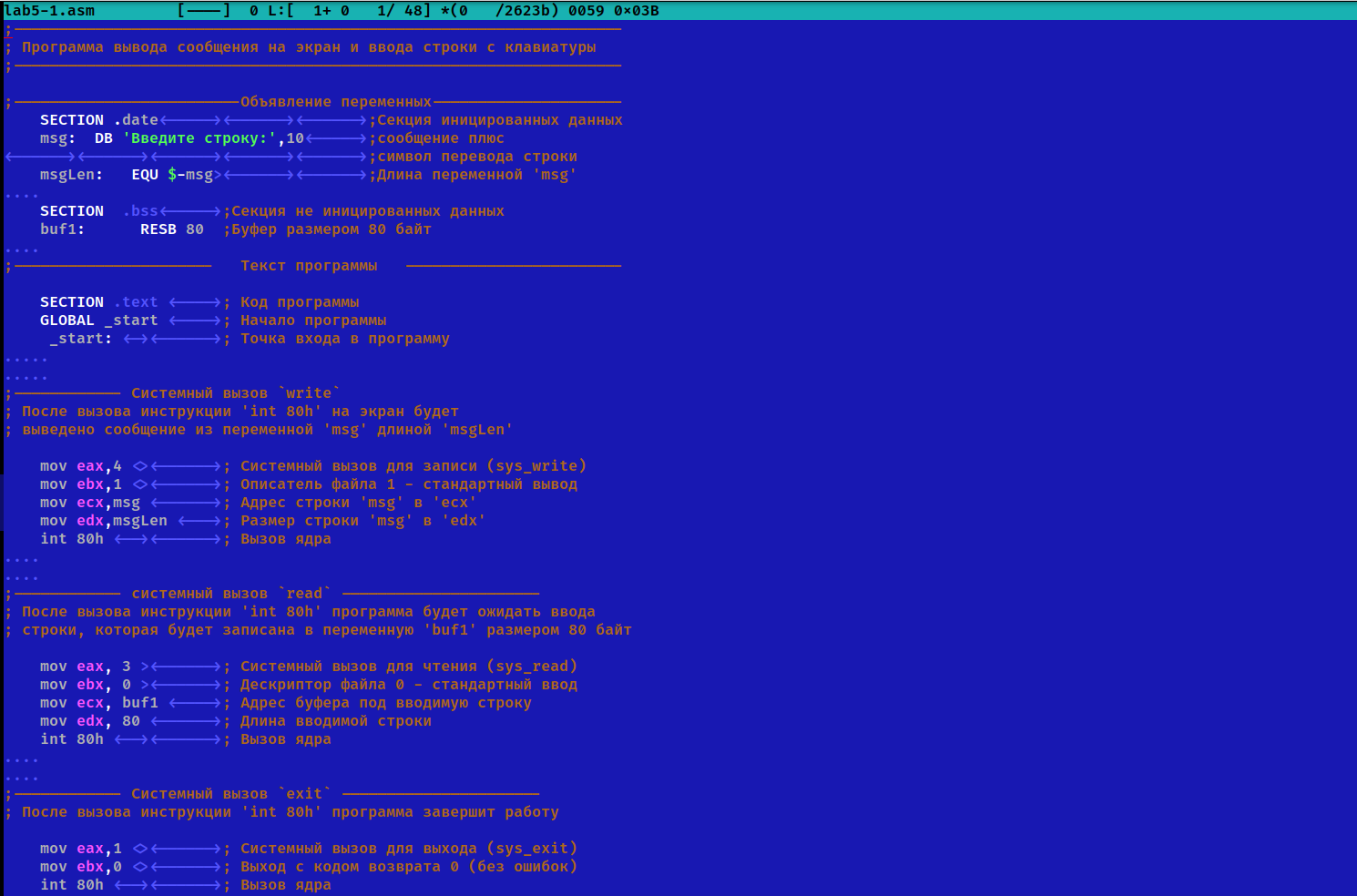


lab05-1.asm используя mcedit

1. Введите текст программы из листинга 5.1, сохраните изменения и закройте файл.

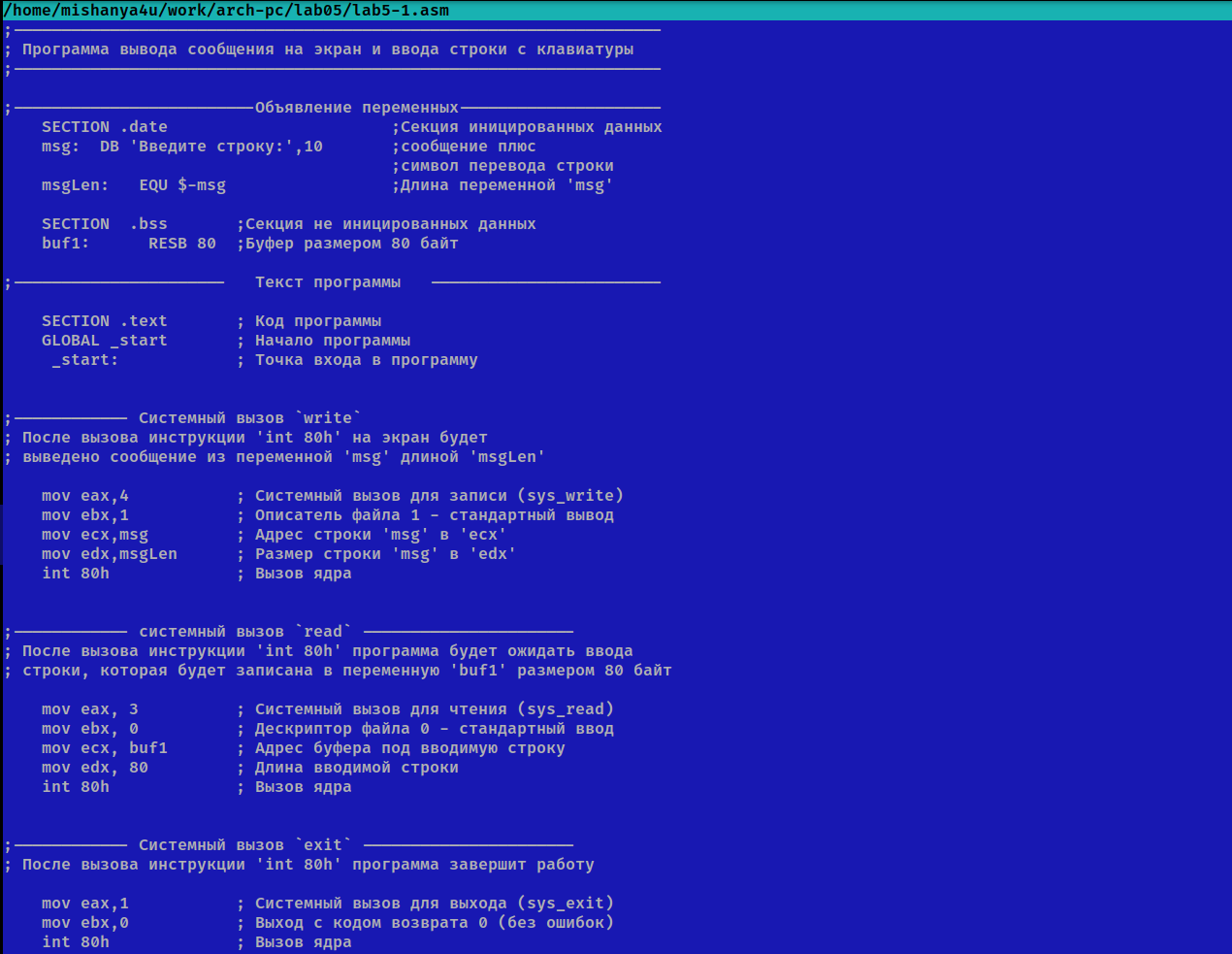


Текст программы из листинга 5.1 в nano



Текст программы из листинга 5.1 в mcedit

1. С помощью функциональной клавиши F3 откройте файл lab5-1.asm для просмотра. Убедитесь, что файл содержит текст программы.



lab5-1.asm

1. Оттранслируйте текст программы lab5-1.asm в объектный файл. Выполните компоновку объектного файла и запустите получившийся исполняемый файл. Программа выводит строку ‘Введите строку:’ и ожидает ввода с клавиатуры. На запрос введите Ваши ФИО.

mishanya4u@Legenda in ~ via C v13.2.1-gcc

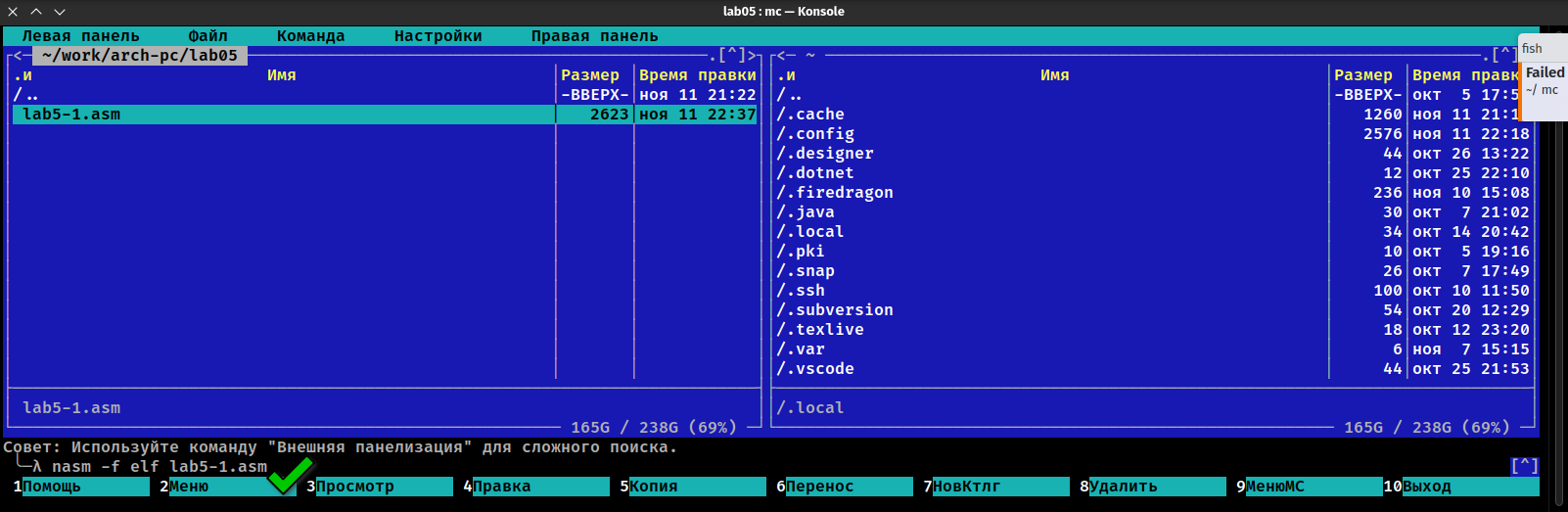
print(nasm -f elf lab5-1.asm)

mishanya4u@Legenda in ~ via C v13.2.1-gcc

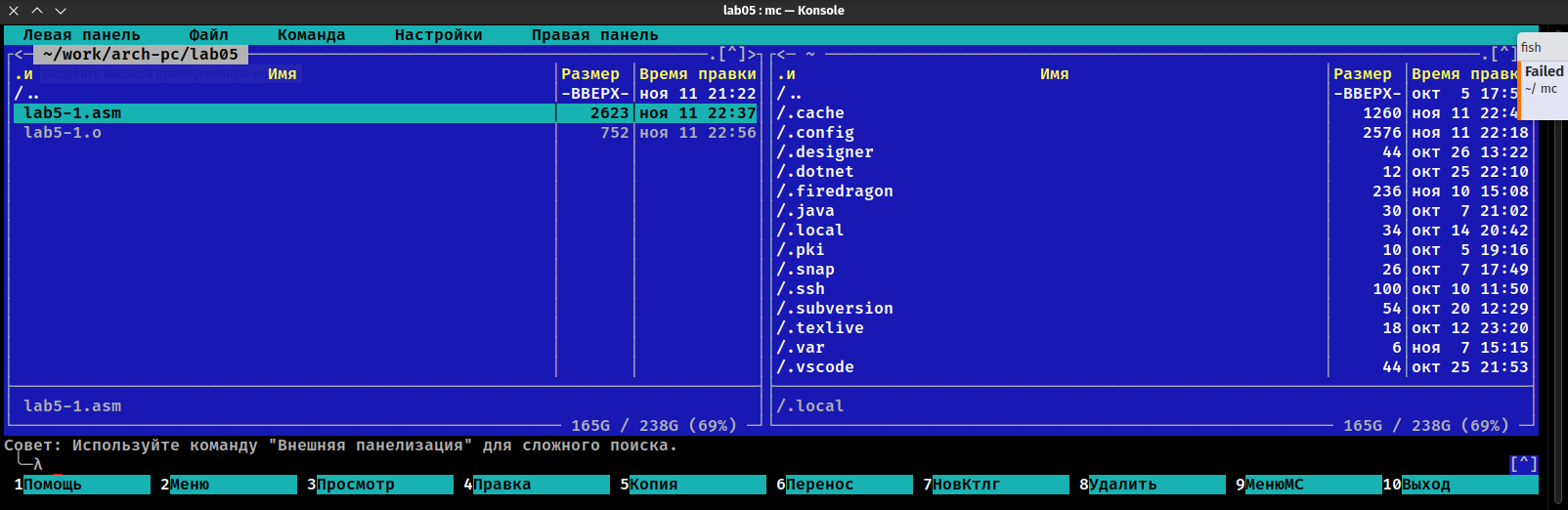
print(ld -m elf\_i386 -o lab5-1 lab5-1.o)

mishanya4u@Legenda in ~ via C v13.2.1-gcc

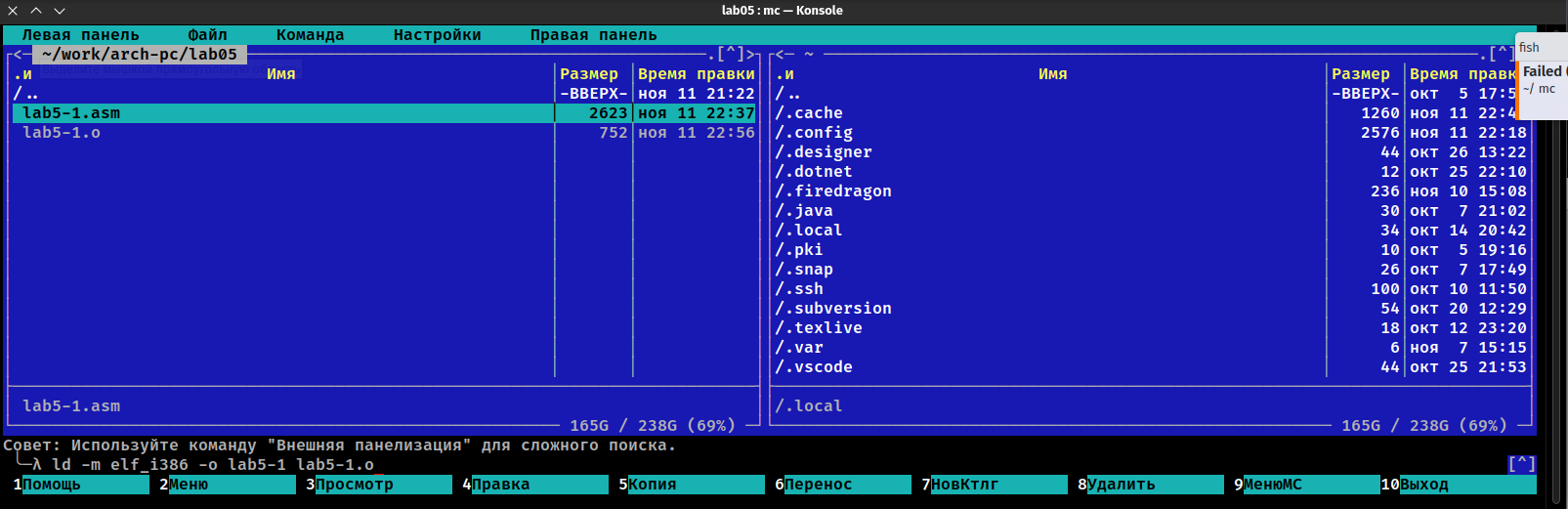
print(./lab5-1)



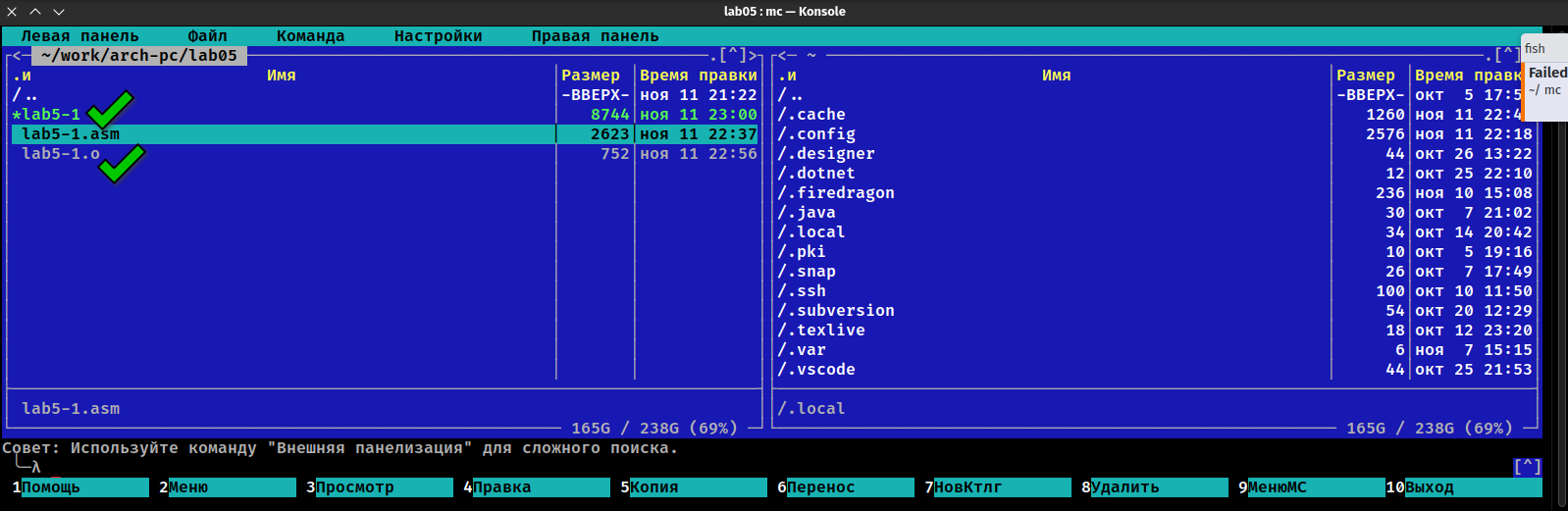
Команда nasm -f elf lab5-1.asm



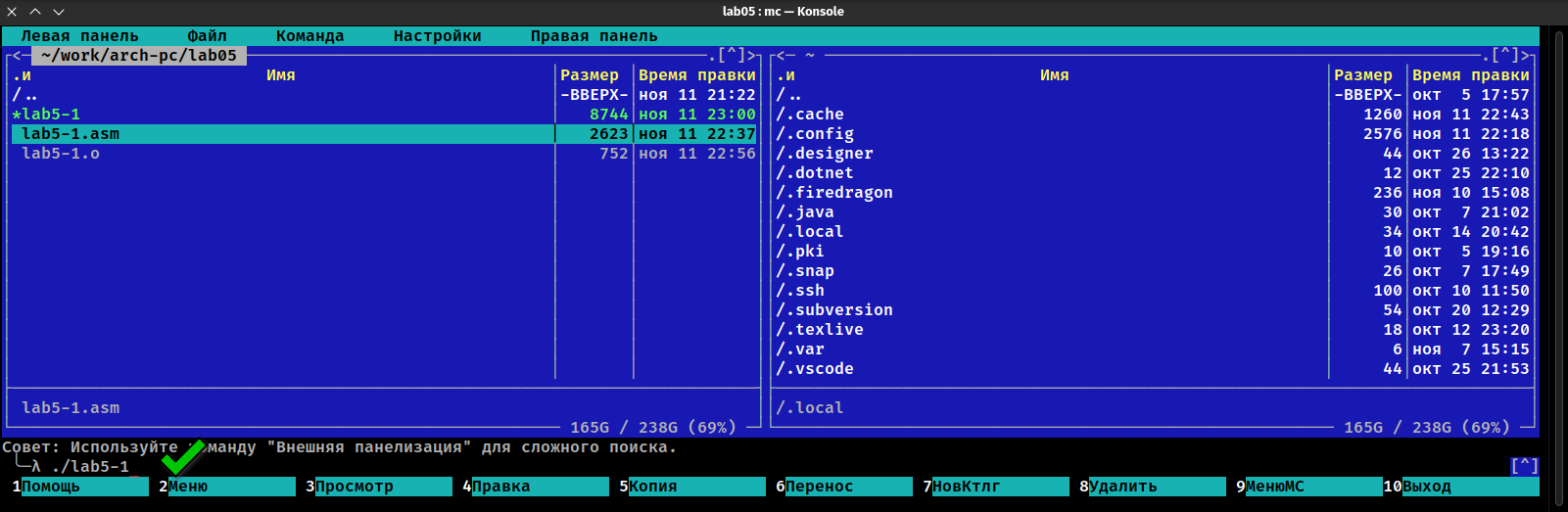
lab5-1.o был создан



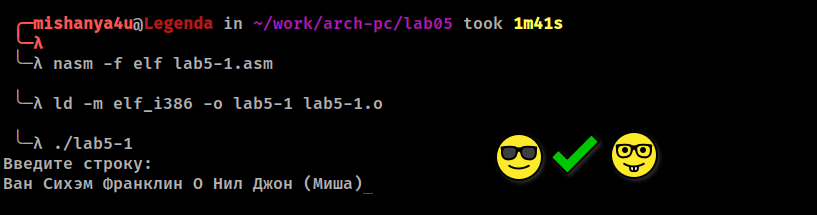
ld -m elf\_i386 -o lab5-1 lab5-1.o



\*lab5-1 был создан

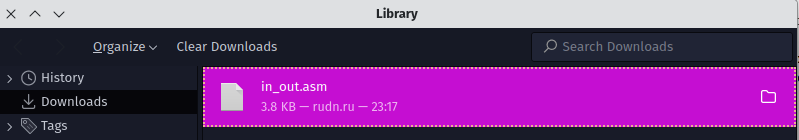


Команда ./lab5-1

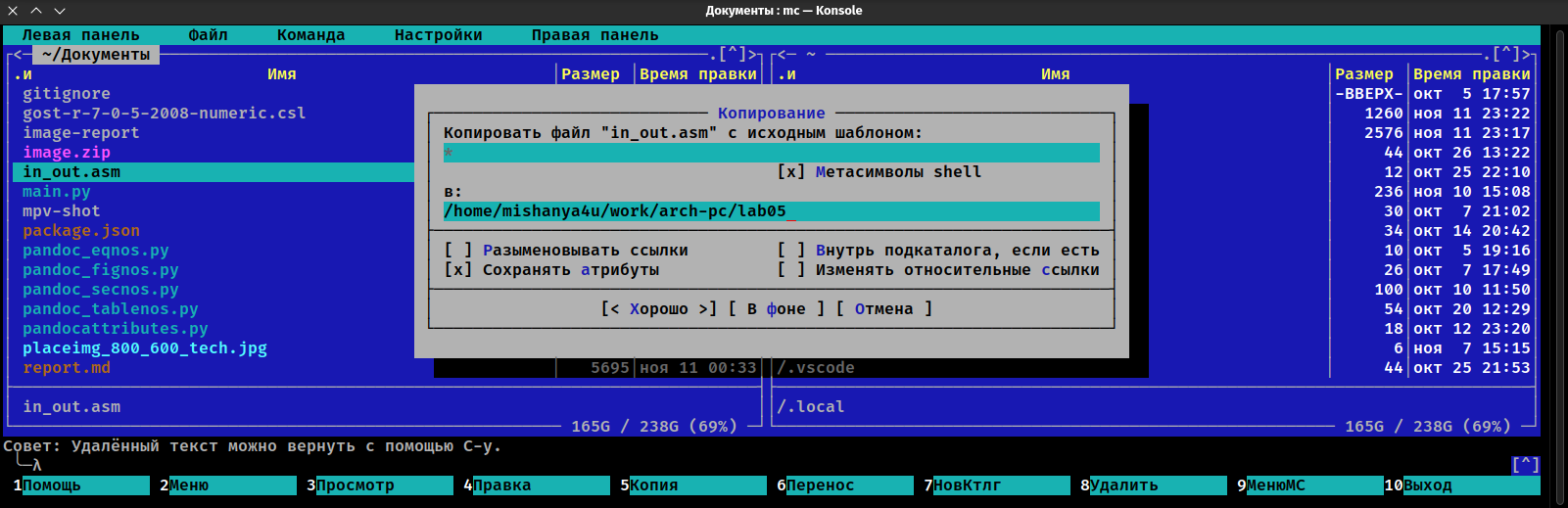


ФИО ввел

1. Скачайте файл in\_out.asm со страницы курса в ТУИС

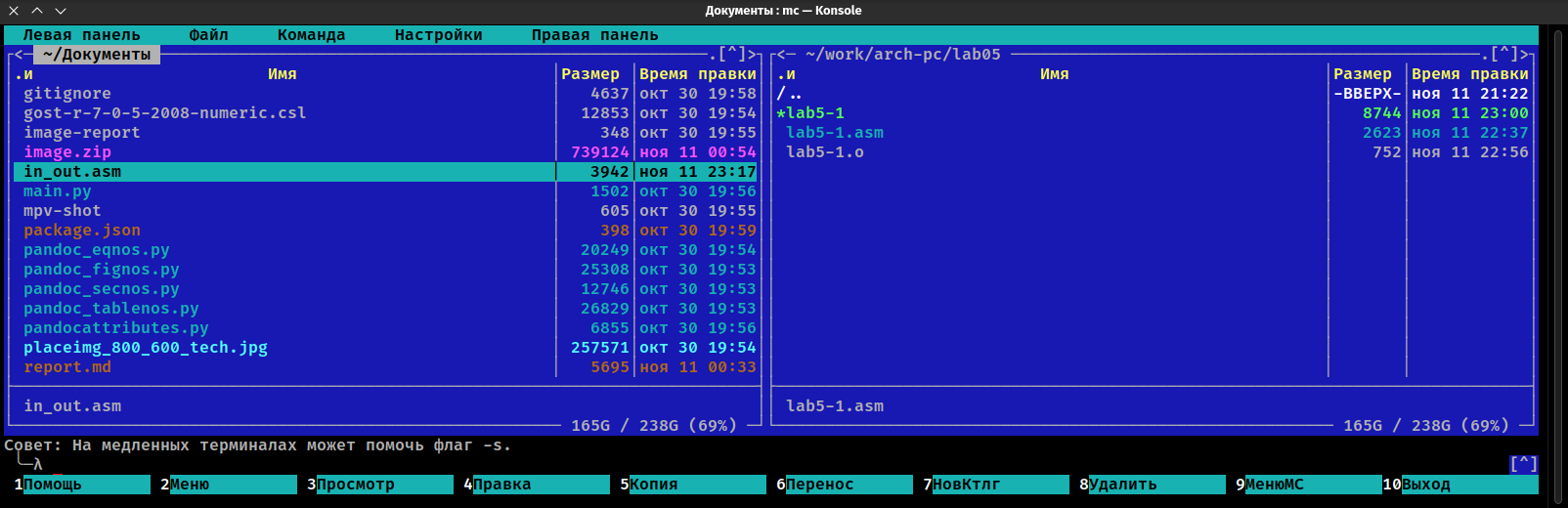


in\_out.asm файл скачанный

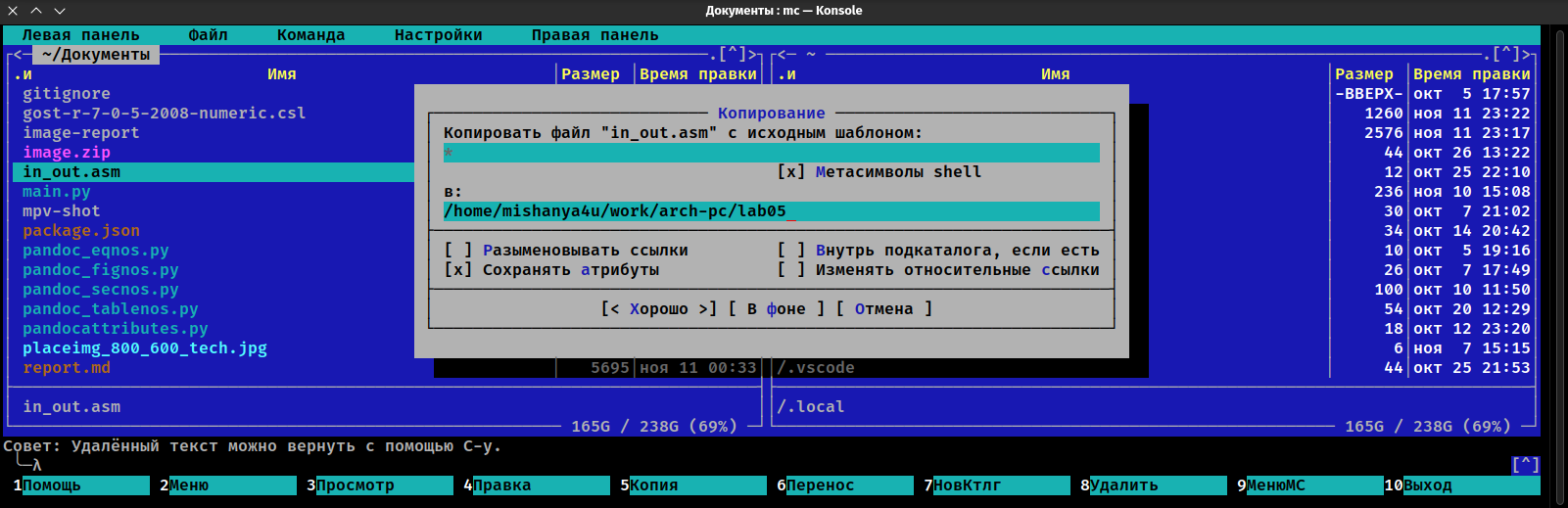


in\_out.asm файл скопирован в каталол ~/work/arch-pc/lab05

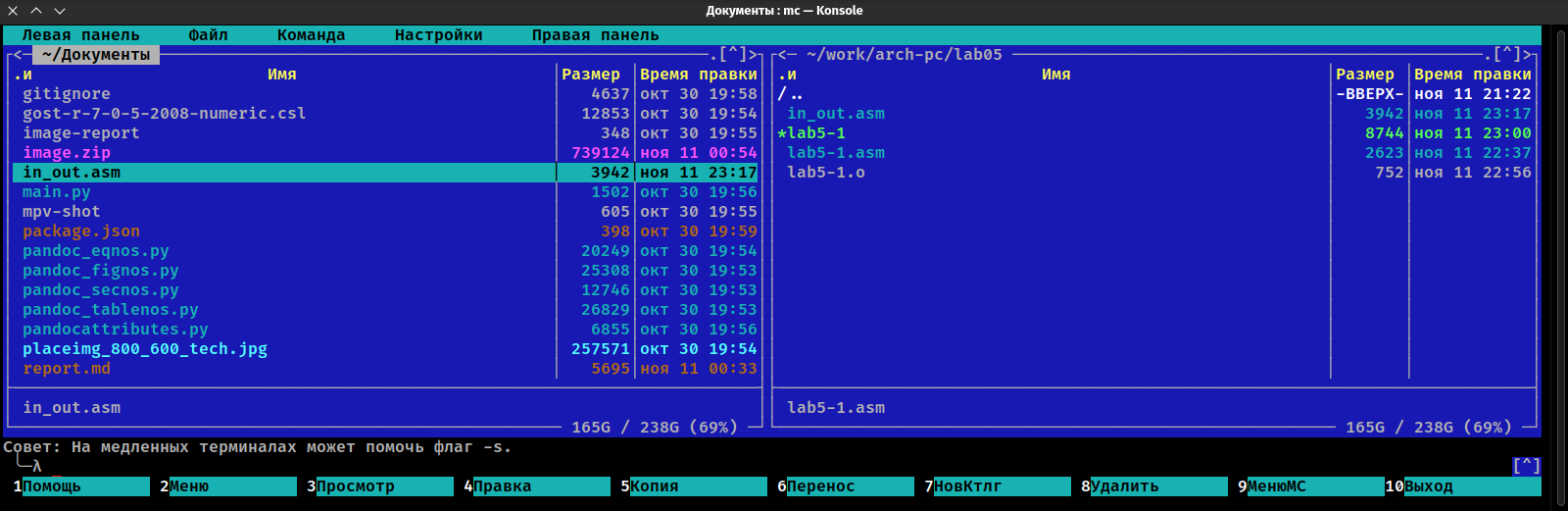
1. Подключаемый файл in\_out.asm должен лежать в том же каталоге, что и файл с программой, в которой он используется. В одной из панелей mc откройте каталог с файлом lab5-1.asm. В другой панели каталог со скаченным файлом in\_out.asm (для перемещения между панелями используйте Tab. Скопируйте файл in\_out.asm в каталог с файлом lab5-1.asm с помощью функциональной клавиши F5.



in\_out.asm файл в каталол ~/Документы

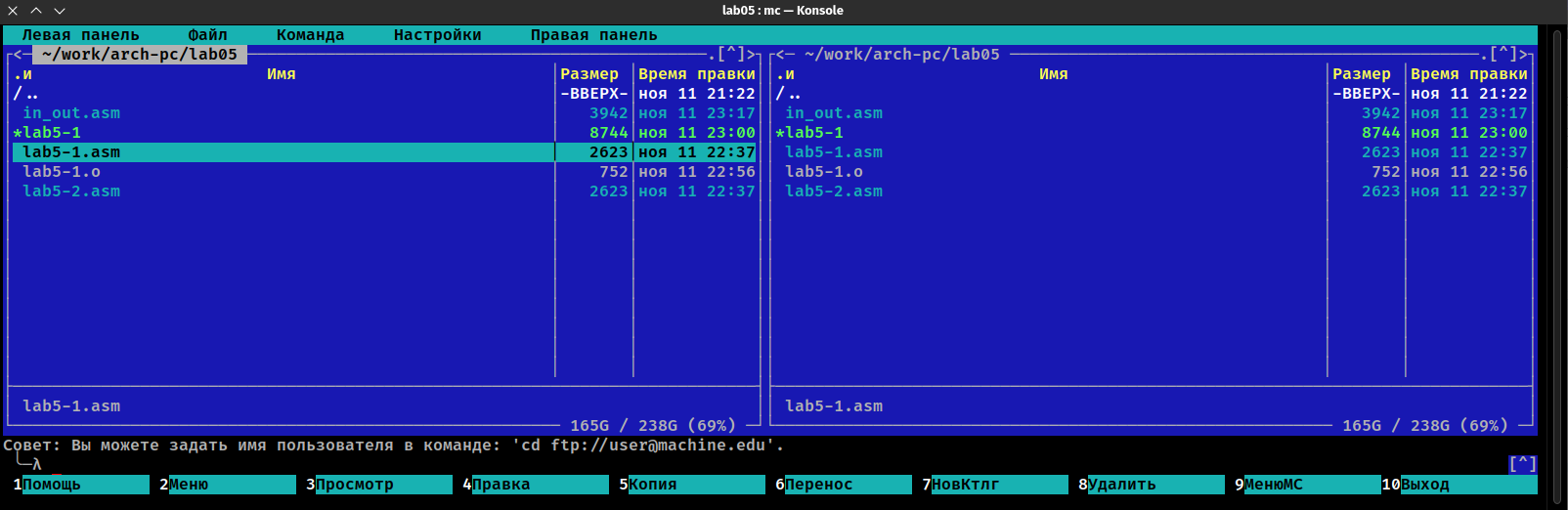


in\_out.asm файл копируется в каталол ~/work/arch-pc/lab05



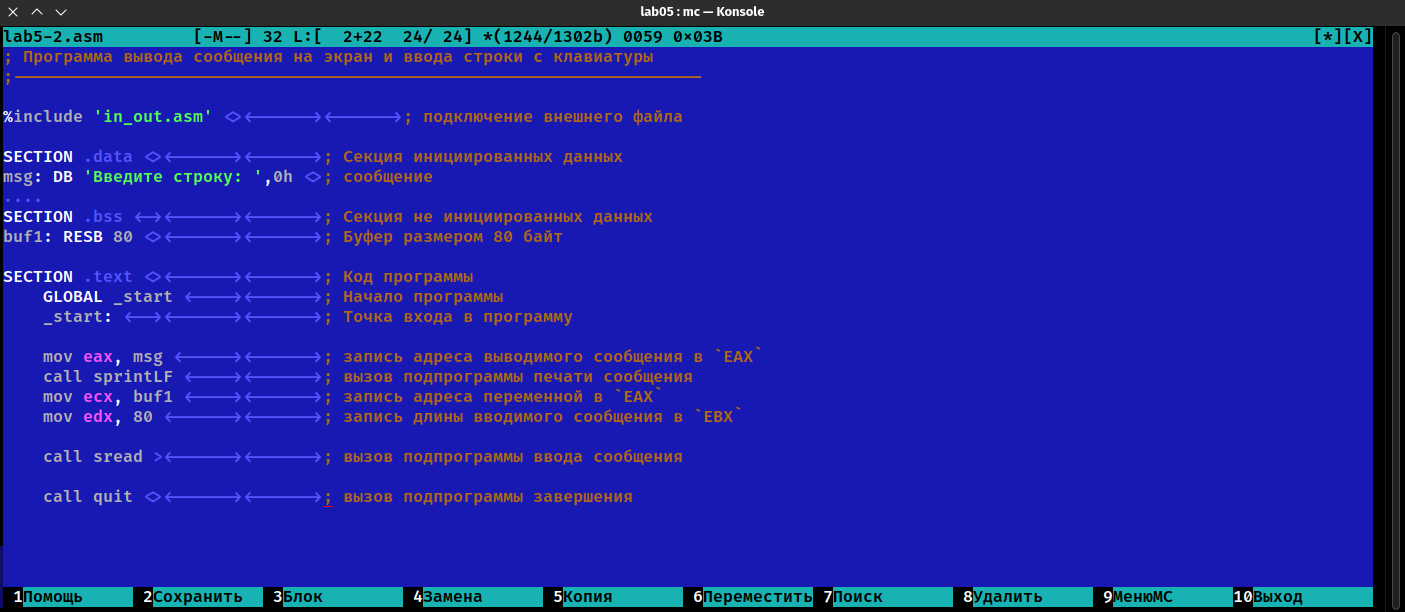
in\_out.asm файл скопирован в каталол ~/work/arch-pc/lab05

1. С помощью функциональной клавиши F6 создайте копию файла lab5-1.asm с именем lab5-2.asm.



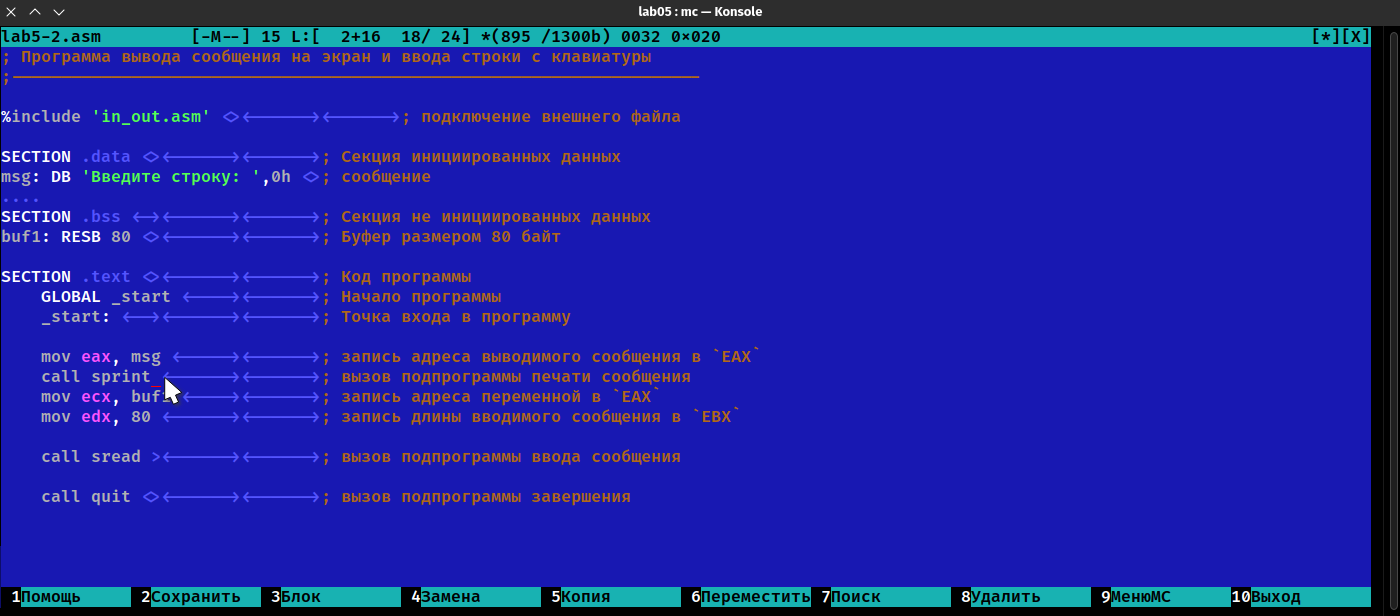
lab5-2.asm был создан

1. Исправьте текст программы в файле lab5-2.asm с использование подпрограмм из внешнего файла in\_out.asm (используйте подпрограммы sprintLF, sread и quit) в соответствии с листингом 5.2. Создайте исполняемый файл и проверьте его работу.



lab5-2.asm с использование подпрограмм из внешнего файла in\_out.asm

1. В файле lab5-2.asm замените подпрограмму sprintLF на sprint. Создайте исполняемый файл и проверьте его работу. В чем разница?



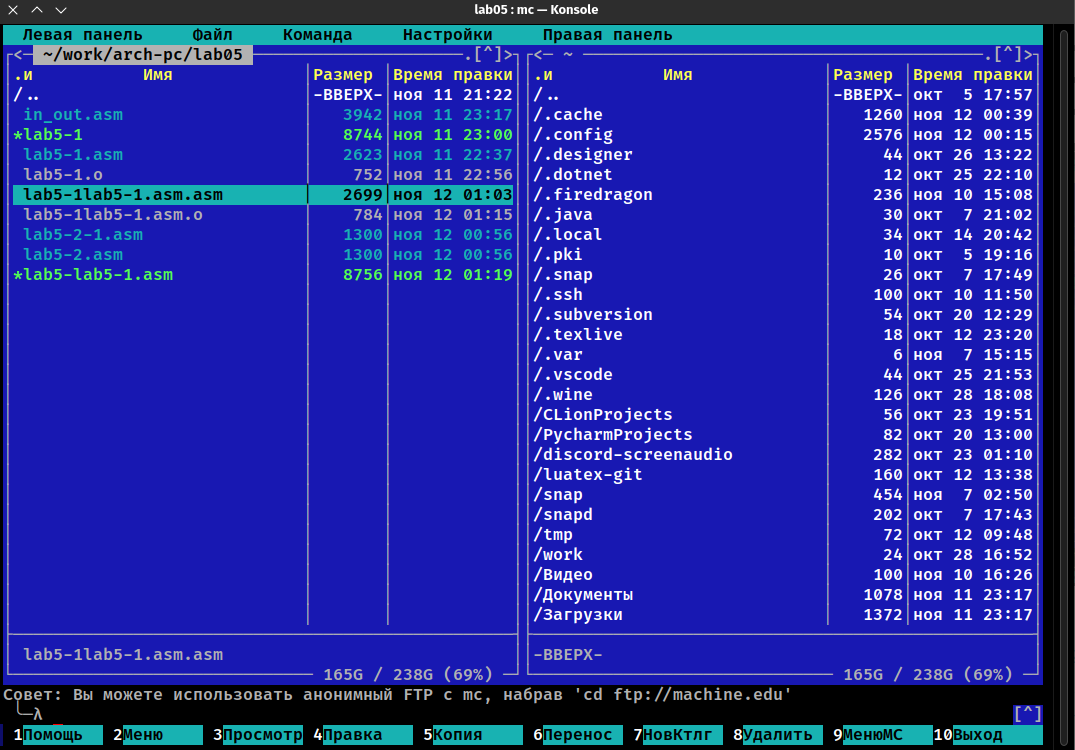
заменил подпрограмму sprintLF на sprint

sprint – вывод сообщения на экран, перед вызовом sprint в регистр eax необходимо записать выводимое сообщение (mov eax, message) а sprintLF также работает но при выводе на экран добавляет к сообщению символ перевода строки.

## 3.1 Задание для самостоятельной работы

1. Создайте копию файла lab5-1.asm. Внесите изменения в программу (без использования внешнего файла in\_out.asm), так чтобы она работала по следующему алгоритму:

* вывести приглашение типа “Введите строку:”;
* ввести строку с клавиатуры;
* вывести введённую строку на экран.

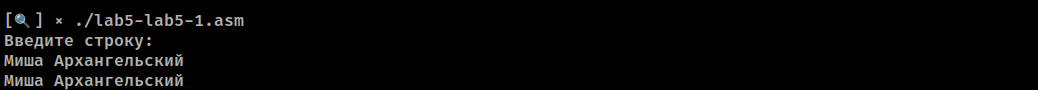


копия файла lab5-1.asm



редактировал файл lab5-1.asm

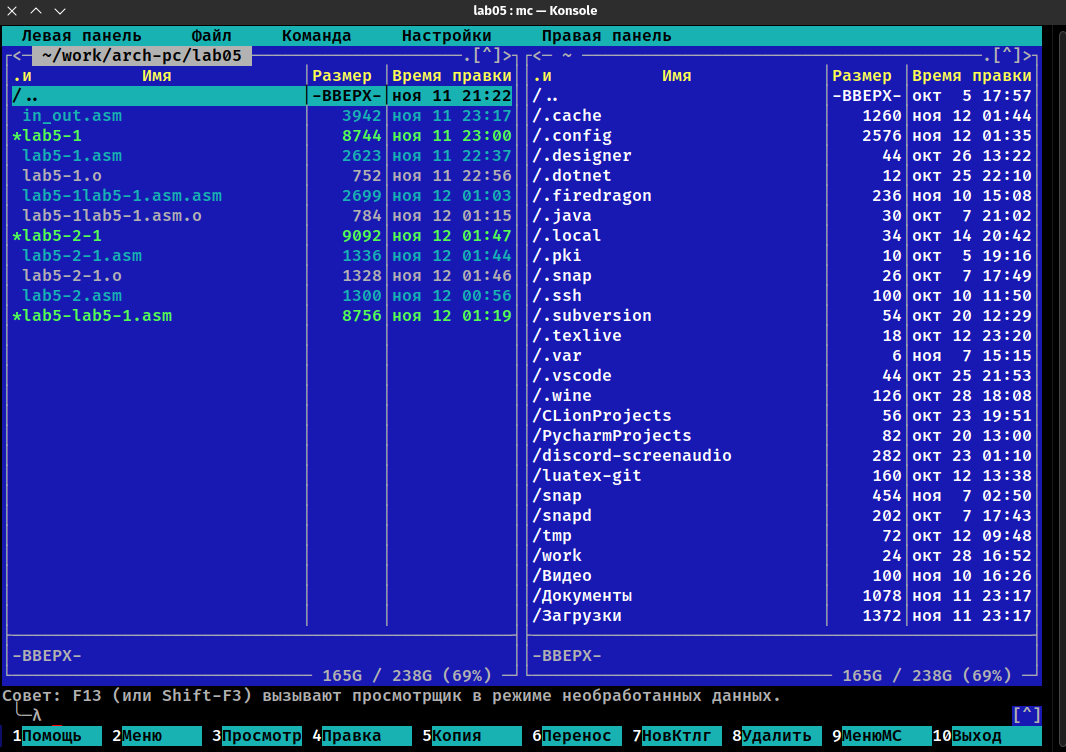
1. Получите исполняемый файл и проверьте его работу. На приглашение ввести строку введите свою фамилию.



вывод lab5-lab5-1.asm

1. Создайте копию файла lab5-2.asm. Исправьте текст программы с использование подпрограмм из внешнего файла in\_out.asm, так чтобы она работала по следующему алгоритму:

* вывести приглашение типа “Введите строку:”;
* ввести строку с клавиатуры;
* вывести введённую строку на экран.

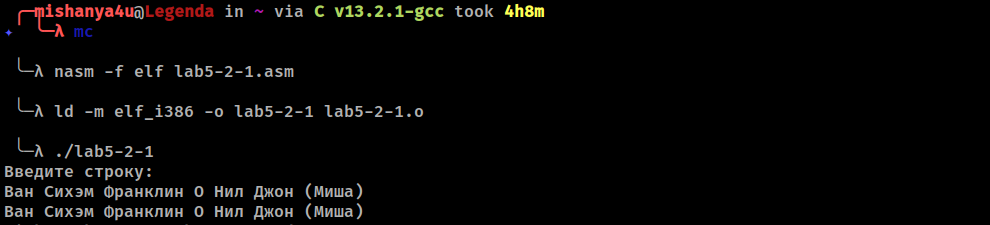


вывод lab5-lab5-1.asm



вывод lab5-lab5-1.asm

1. Создайте исполняемый файл и проверьте его работу.



вывод lab5-lab5-1.asm

## 3.2 Листинг 5.1

;------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;------------------------------------------------------------------  
;------------------- Объявление переменных ----------------  
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку:',10 ; сообщение плюс  
 ; символ перевода строки  
msgLen: EQU $-msg ; Длина переменной 'msg'  
  
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  
  
;------------ Cистемный 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,1 ; Системный вызов для выхода (sys\_exit)  
 mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)  
 int 80h ; Вызов ядра

## 3.3 Листинг 5.2

;--------------------------------------------------------------------  
; Программа вывода сообщения на экран и ввода строки с клавиатуры  
;---------------------------------------------------------------------  
  
%include 'in\_out.asm' ; подключение внешнего файла  
  
SECTION .data ; Секция инициированных данных  
msg: DB 'Введите строку: ',0h ; сообщение  
  
SECTION .bss ; Секция не инициированных данных  
buf1: 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 sprintLF ; вызов подпрограммы печати сообщения  
 mov ecx, buf1 ; запись адреса переменной в `EAX`  
 mov edx, 80 ; запись длины вводимого сообщения в `EBX`  
  
 call sread ; вызов подпрограммы ввода сообщения  
  
 call quit ; вызов подпрограммы завершения

# 4 Заключение

После работы с Midnight Commander, могу сказать что МС очень простой и интуитивно понятный интерфейс, который позволяет легко навигировать по файловой системе и выполнять нужные операции. Он также поддерживает работу с архивами, просмотр и редактирование текстовых файлов, а также поддерживает множество других фунций, которые делают его очень полезным инструментом для системного администратора или разработчика.