Отчёт по лабораторной работе №5

Дисциплина: Архитектура компьютера

Студент: Верниковская Екатерина Андреевна

Содержание

# Цель работы

Приобрести практические навыки работы в Midnight Commander и освоить инструкции языка ассемблера mov и int.

# Задание

1. Открыть Midnight Commander.
2. Создать папку lab05, где будут храниться файлы лабортаорной работы №5.
3. Создать файл «lab5-1.asm», ввести текст программы. Оттранслировать текст программы, выполнить компоновку объектного файла и запустить получившийся исполняемый файл.
4. Скачать с ТУИС файл «in\_out.asm» и переместить его в каталог lab05.
5. Скопировать файл «lab5-1.asm» с именем «lab5-2.asm» и исправить текст программы так чтобы использовались программы из внешнего файла «in\_out.asm».
6. Создать исполняемый файл и проверить его работу.
7. Создать копию файла «lab5-1.asm». Внести измененя в программу (без использования внешнего файла «in\_out.asm»), так чтобы она работа по определённуму алгоритму.
8. Создать копию файла «lab5-2.asm». Также исправить текст программы, но уже с использованием подпрограмм из внешнего файла «in\_out.asm», так чтобы она работала по определённуму алгоритму.
9. Создать исполняемые файлы и проверить их работу.

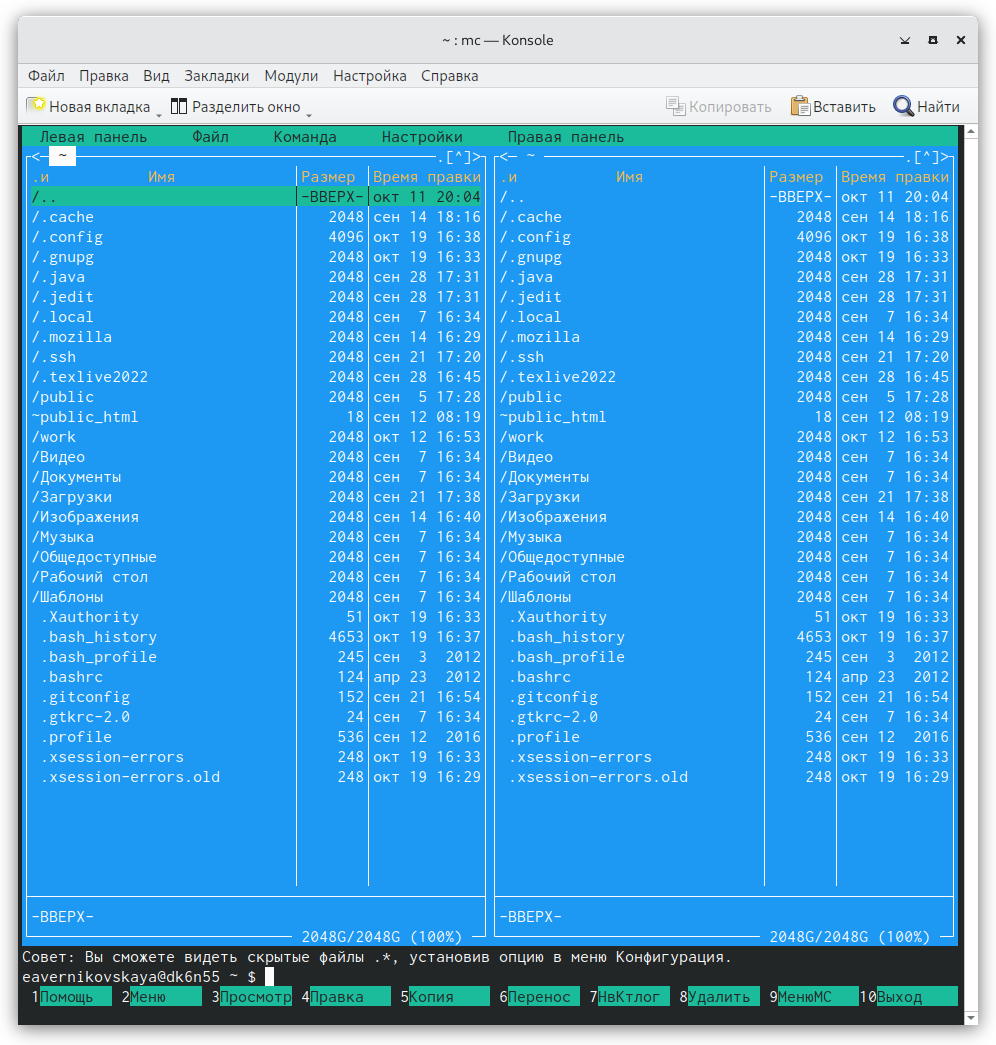
# Выполнение лабораторной работы

## Изучение Midnight Commander

Открываем Midnight Commander с помощью команды ‘mc’ (рис. [-@fig:001]), (рис. [-@fig:002])

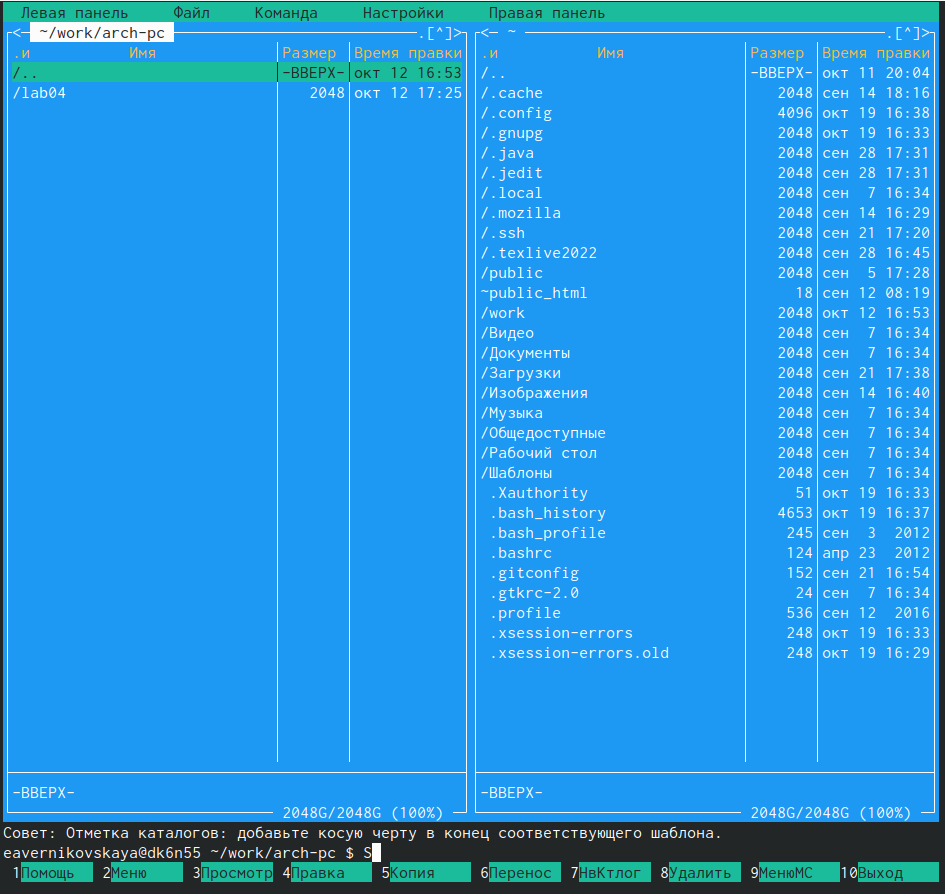
Открытие Midnight Commander

Открытие Midnight Commander



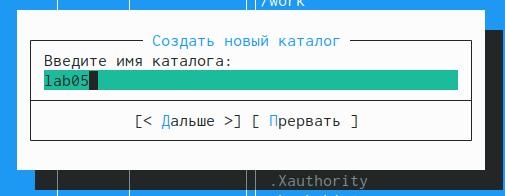
Открытый Midnight Commander

Пользуясь клавишами на клавиатуре ‘вверх’, ‘вниз’ и ‘enter’ переходим в каталог «~/work/arch-pc» (рис. [-@fig:003])



Переход в нужный каталог

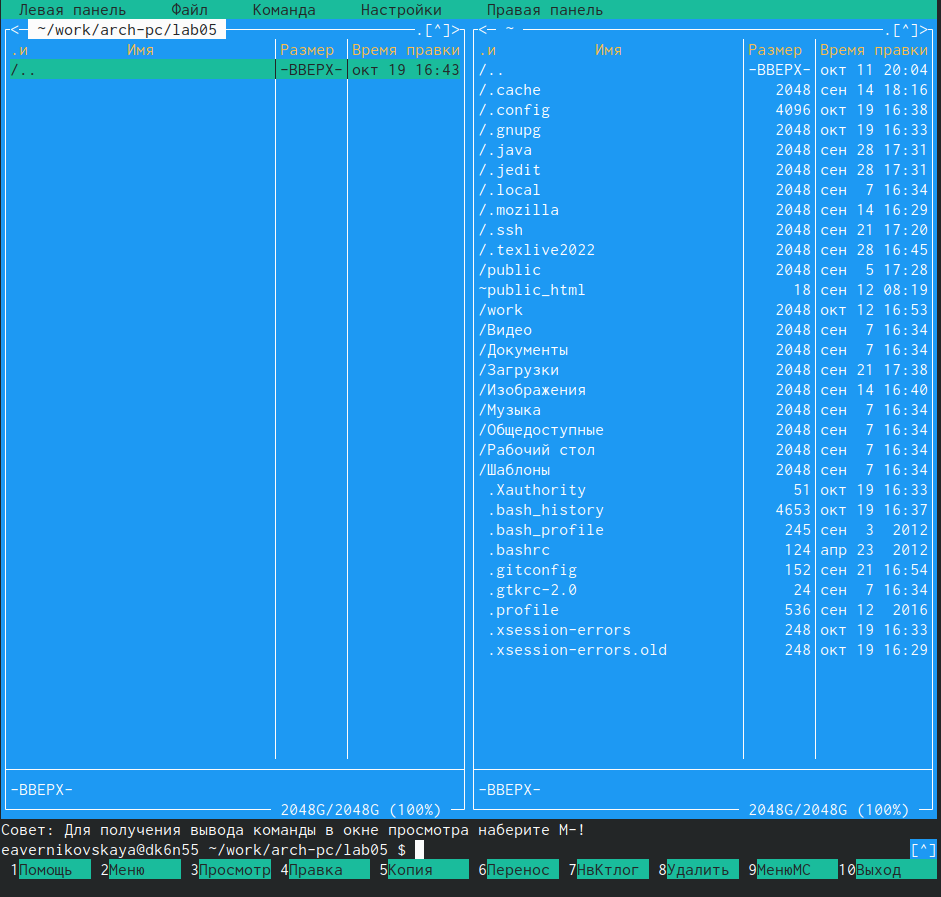
С помощью функциональной клавиши ‘F7’ создаём папку «lab05» и переходим в неё рис. [-@fig:004]), (рис. [-@fig:005]), рис. [-@fig:006])



Создание папки

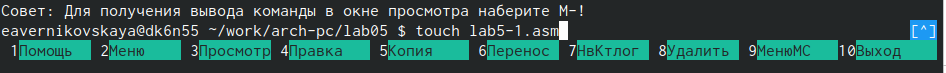


Проверка

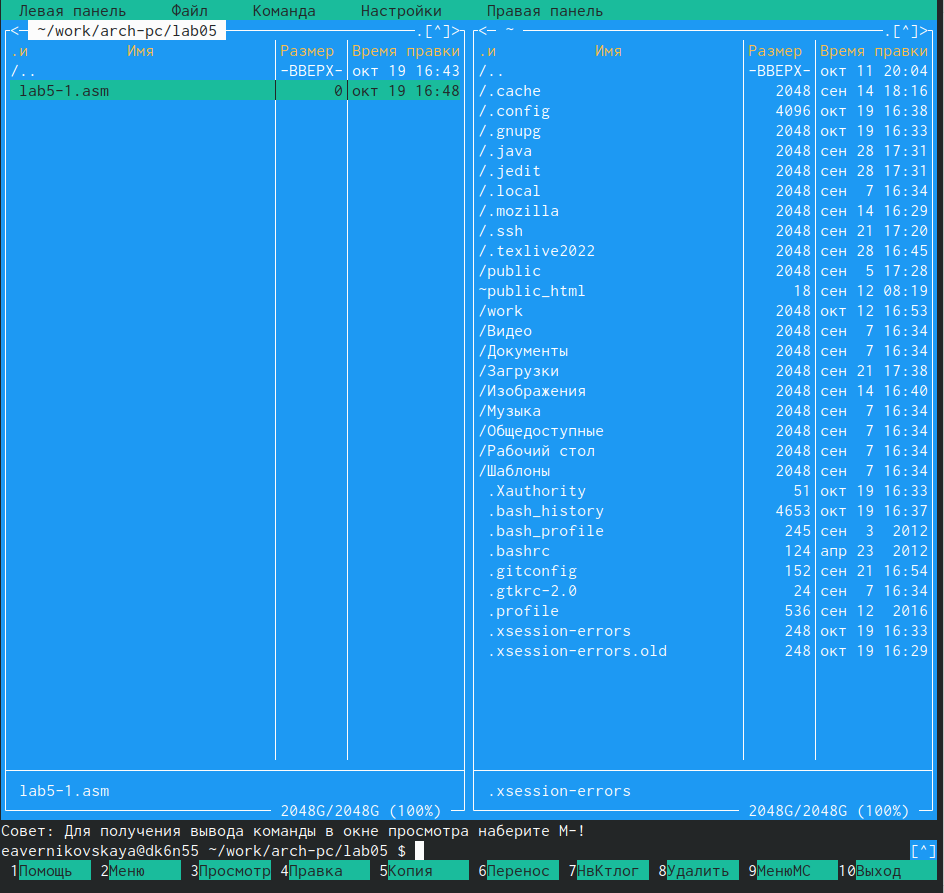


Переход в созданную папку

Пользуясь строкой ввода и командой ‘touch’ создаём файл «lab5-1.asm» рис. [-@fig:007]), (рис. [-@fig:008])

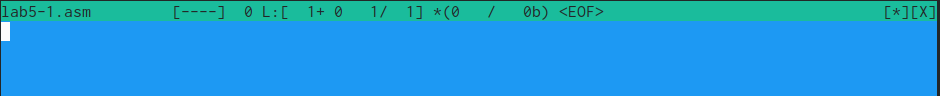


Создание файла



Проверка

С помощью функциональной клавиши ‘F4’ открываем файл «lab5-1.asm» для редактирования во встроенном редакторе. В нашем случае это mcedit (рис. [-@fig:009])

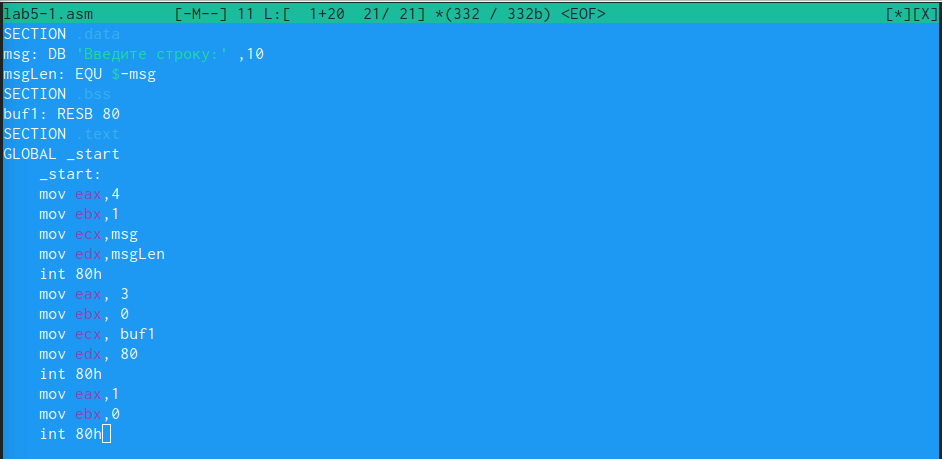


Открытие файла

Вводим текст программы (рис. [-@fig:010])

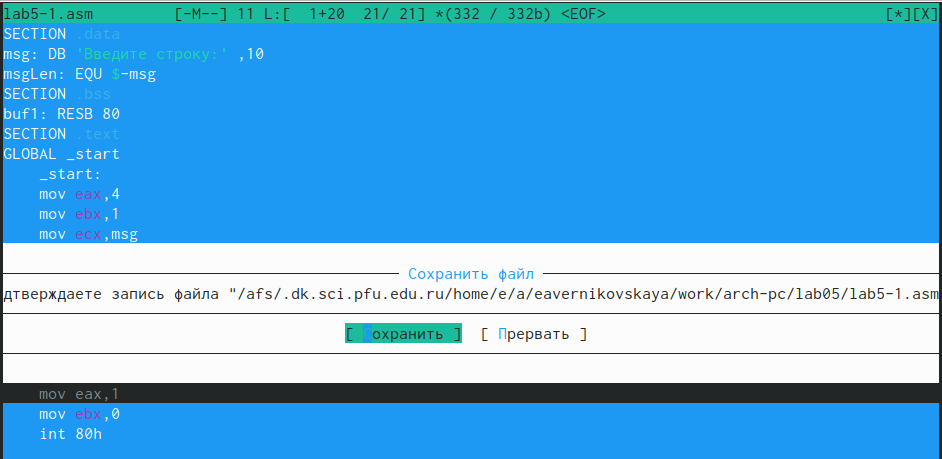
Текст программы:

SECTION .data   
msg: DB 'Введите строку:',10   
msgLen: EQU $-msg   
SECTION .bss   
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,1   
 mov ebx,0   
 int 80h



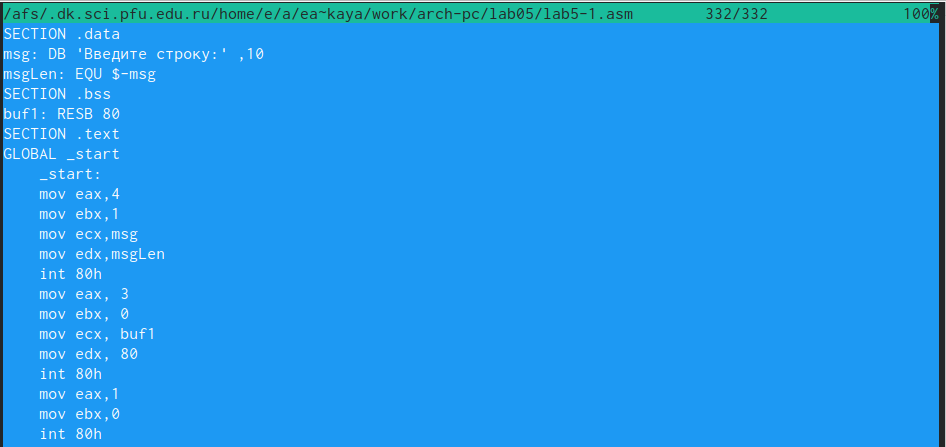
Ввод текста программы

С помощью функциональной клавиши ‘F2’ сохраняем изменения и выходим с помощью ‘F10’ (рис. [-@fig:011])



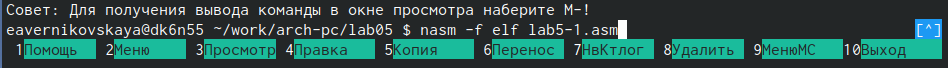
Сохранение файла

С помощью функциональной клавиши ‘F3’ открываем изменённый файл, чтобы убедиться, что он содержит текст программы (рис. [-@fig:012])

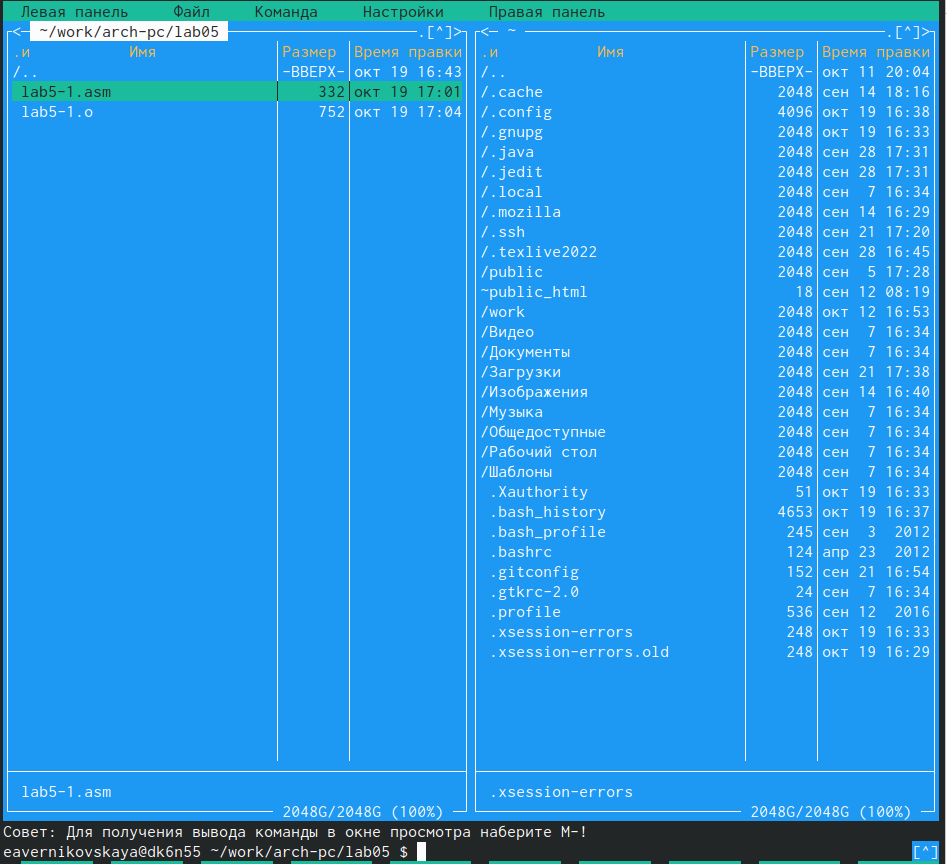


Проверка

Транслируем текст программы в объектрный файл, компилируем исходный файл, передаём объектный файл на обработку компоновщику (рис. [-@fig:013]), (рис. [-@fig:014]), (рис. [-@fig:015]), (рис. [-@fig:016]), (рис. [-@fig:017]), (рис. [-@fig:018]), (рис. [-@fig:019]), (рис. [-@fig:020])



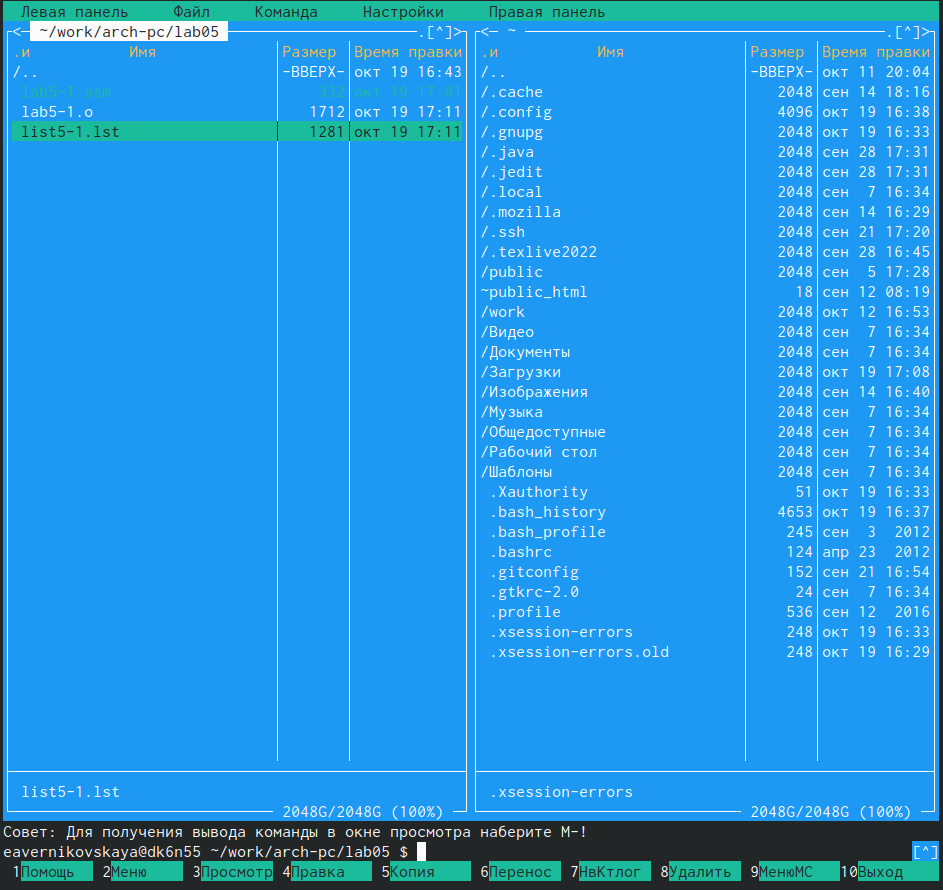
Транслирование



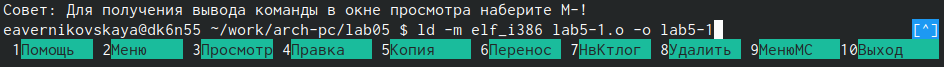
Проверка 1

Компиляция

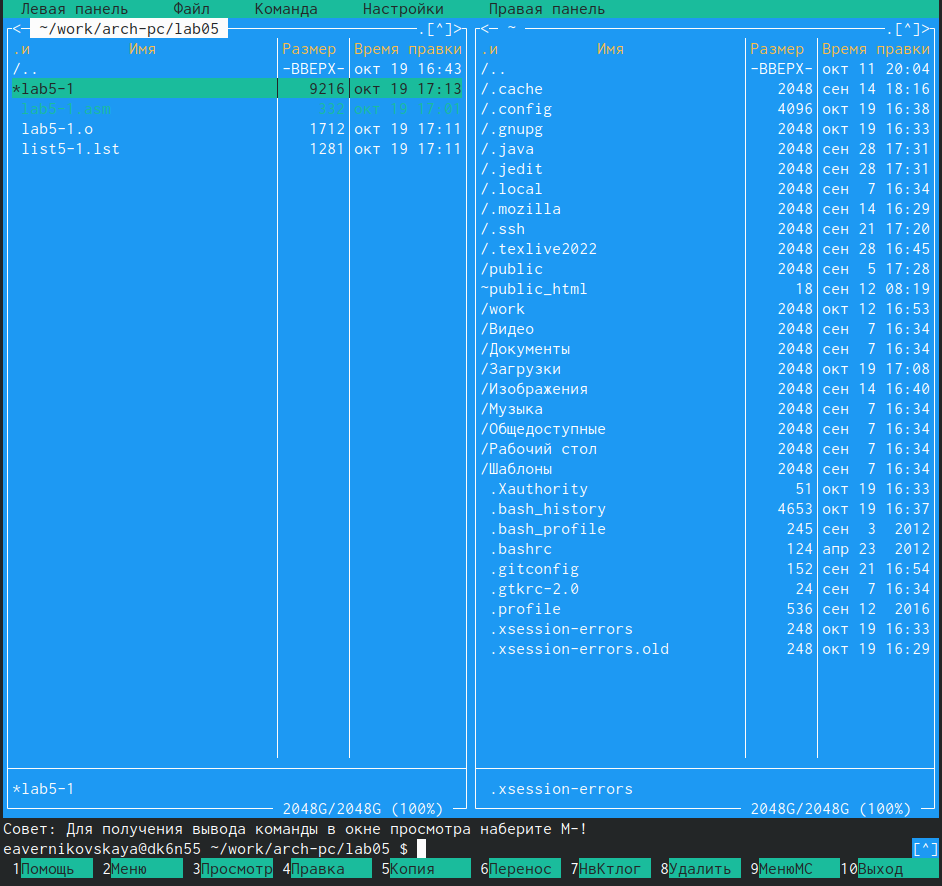
Компиляция



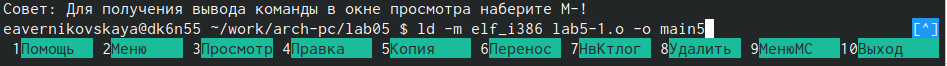
Проверка 2



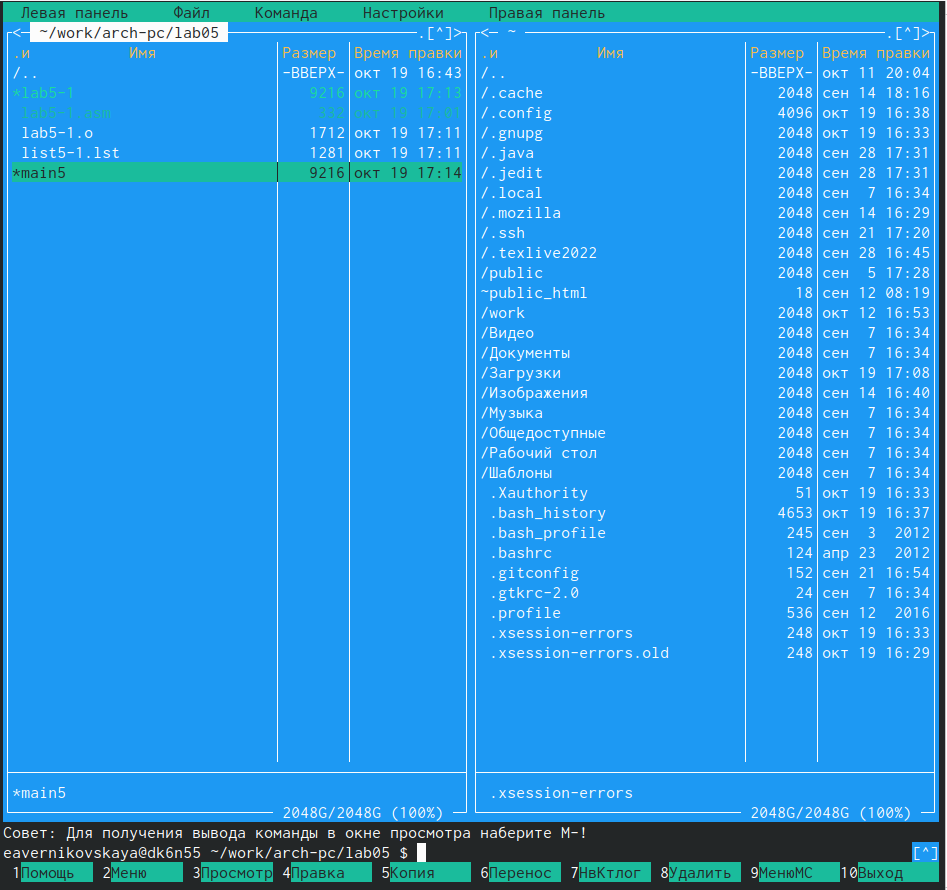
Компоновка



Проверка 3



Компоновка 2



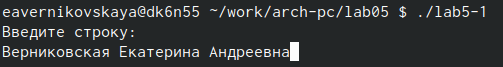
Проверка 4

Запускаем получившийся исполняемый файл (рис. [-@fig:021])

Запуск файла

Запуск файла

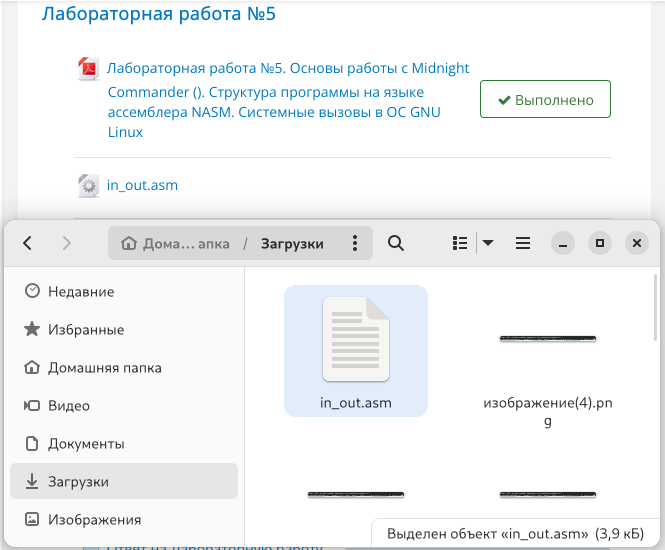
На запрос вводим наше ФИО (рис. [-@fig:022])



Ввод с клавиатуры

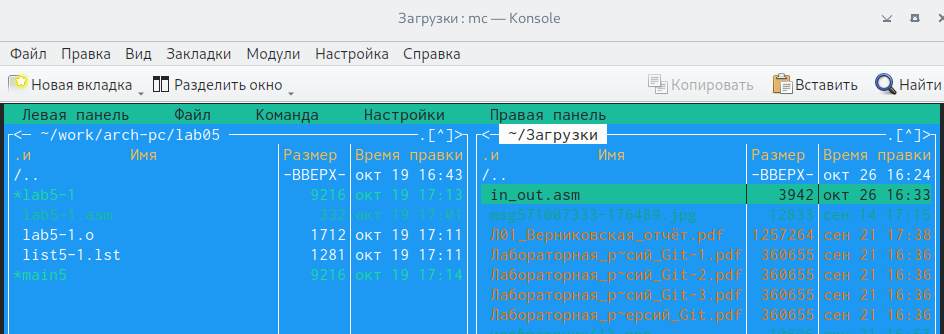
## Подключение внешнего файла in\_out.asm

Скачиваем файл «in\_out.asm» с ТУИС (рис. [-@fig:023])



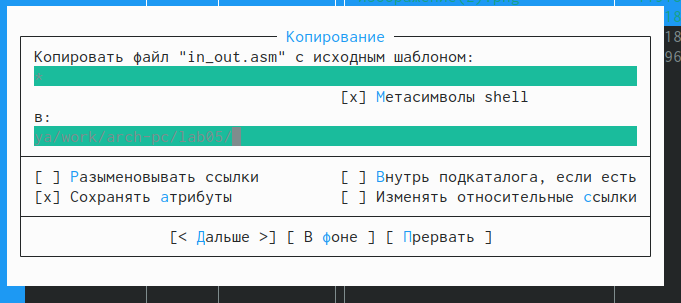
Скачивание файла

В одной из панелей mc открываем каталог с файлом «lab5-1.asm», в другой - каталог со скачанным файлом «in\_out.asm» (рис. [-@fig:024])

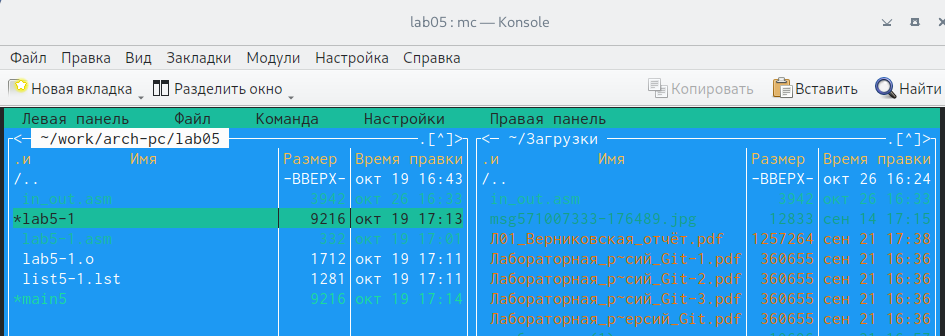


Открытие двух панелей

С помощью функциональной клавиши ‘F5’ копируем файл «in\_out.asm» в каталог с файлом «lab5-1.asm» (рис. [-@fig:025]), (рис. [-@fig:026])

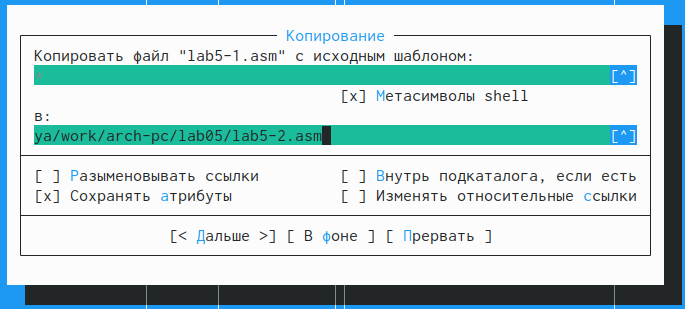


Копирование файла в нужный каталог

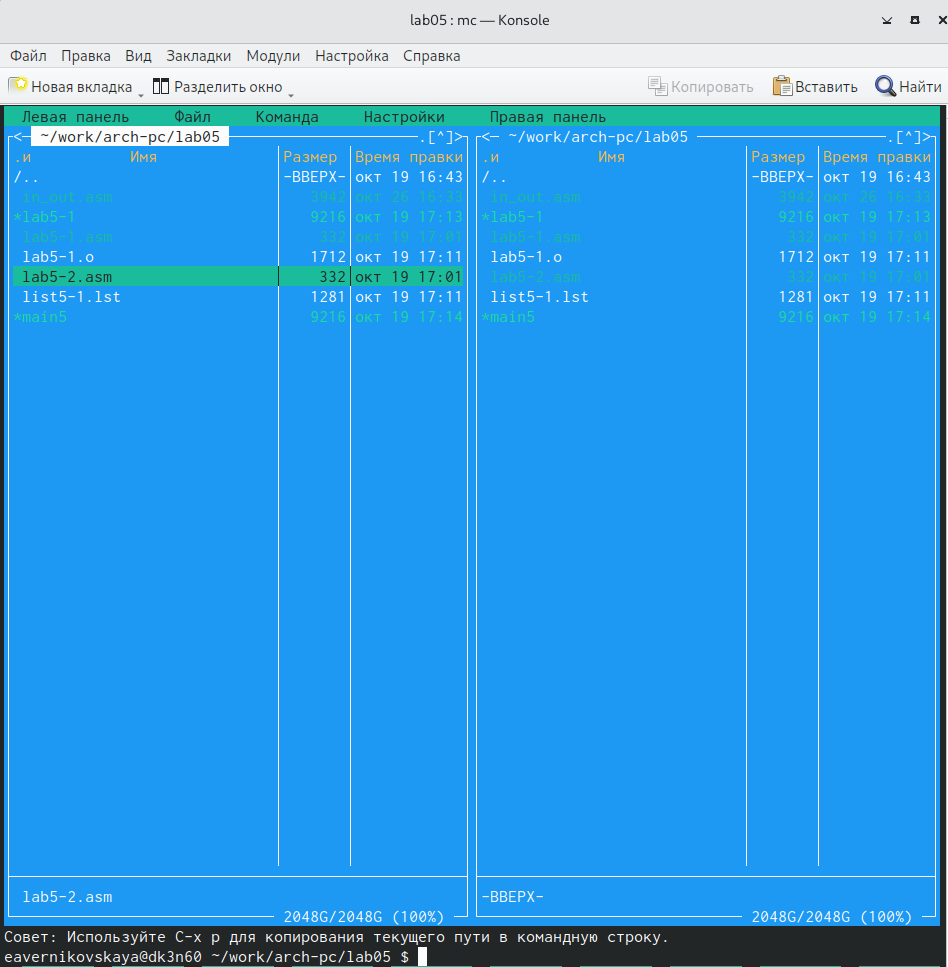


Проверка

С помощью той же функциональной клавиши ‘F5’ создаём копию файла «lab5-1.asm» с именем «lab5-2.asm» (рис. [-@fig:027]), (рис. [-@fig:028])



Создание копии файла с другим именем

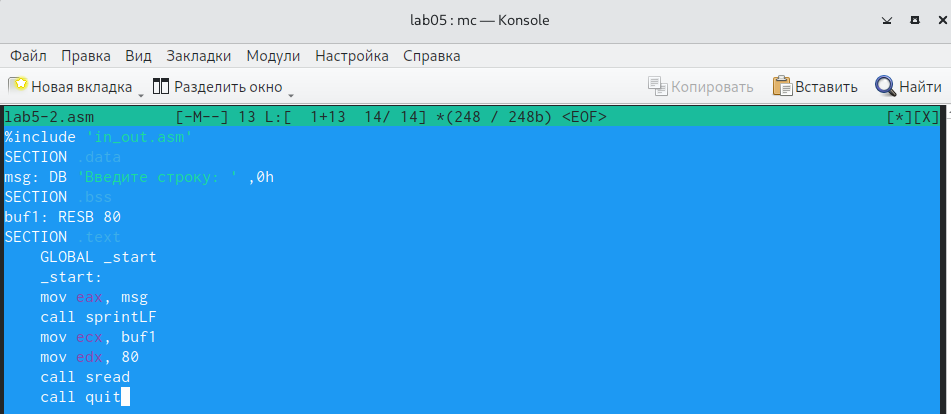


Проверка

С помощью функцональной клавиши ‘F4’ открываем файл «lab5-2.asm» для редактирования и исправляем текст программы с использованием подпрограмм из внешнего файла «in\_out.asm» (рис. [-@fig:029])

Текст программы:

%include 'in\_out.asm'  
SECTION .data   
msg: DB 'Введите строку:',10   
SECTION .bss   
buf1: RESB 80  
SECTION .text   
GLOBAL \_start   
 \_start:  
 mov eax, msg   
 call sprintLF  
 mov ecx, buf1  
 mov edx, 80  
 call sread  
 call quit

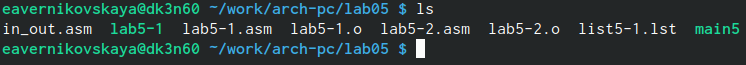


Исправленная программа

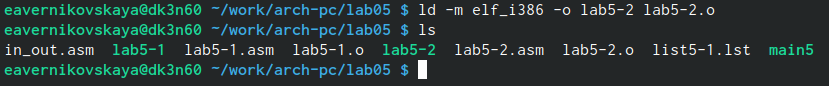
Транслируем текст программы в объектный файл, выполняем компоновку объектного файла и запускаем получившийся исполняемый файл (рис. [-@fig:030]), (рис. [-@fig:031]), (рис. [-@fig:032]), (рис. [-@fig:033])

Транслирование

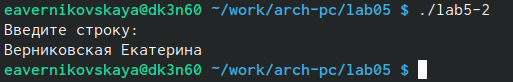
Транслирование



Проверка



Компоновка

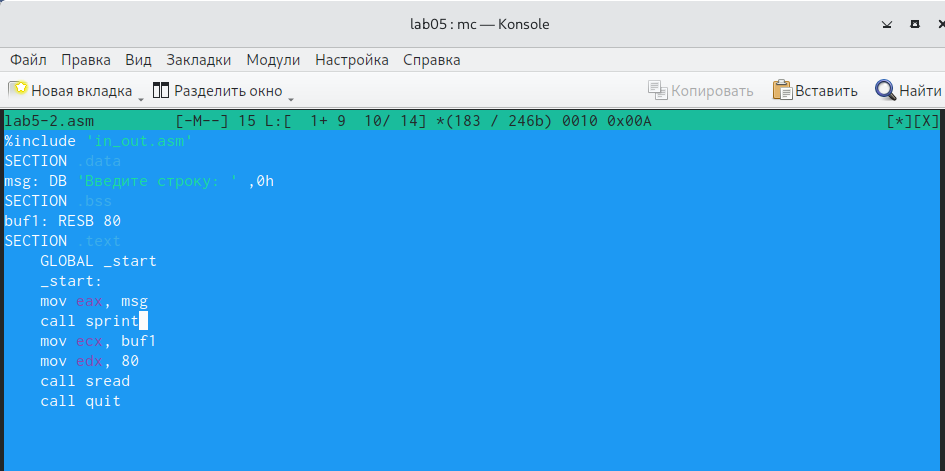


Запуск программы

В файле «lab5-2.asm» заменяем подпрограмму ‘sprintLF’ на ‘sprint’ (рис. [-@fig:034])

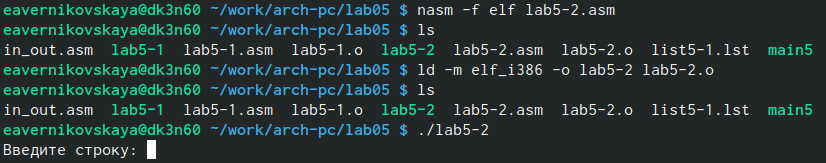
Текст исправленной программы:

%include 'in\_out.asm'  
SECTION .data   
msg: DB 'Введите строку:',10   
SECTION .bss   
buf1: RESB 80  
SECTION .text   
GLOBAL \_start   
 \_start:  
 mov eax, msg   
 call sprint  
 mov ecx, buf1  
 mov edx, 80  
 call sread  
 call quit



Исправление программы

Создаём исполняемый файл и проверяем его работу (рис. [-@fig:035])

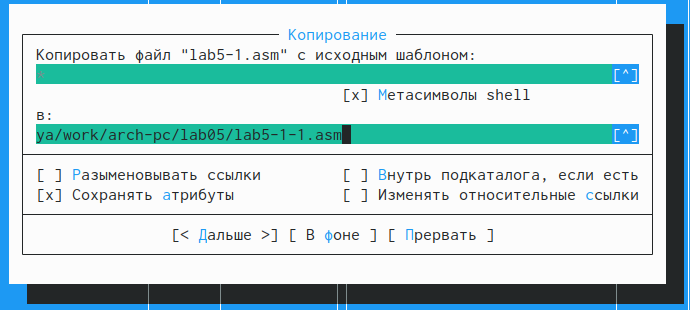


Проверка

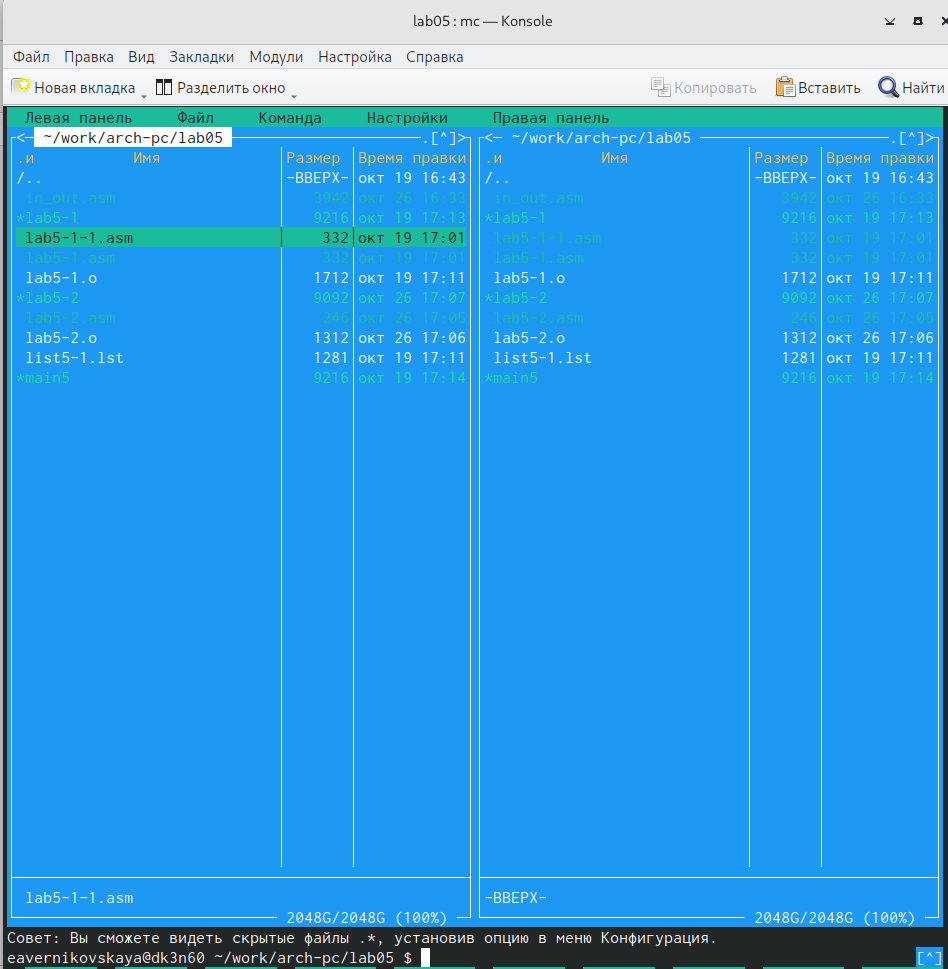
Ответ на вопрос: в первом случае программа просила нас ввести текст на следующей строке, во втором - сразу после знака двоеточия.

## Задание для самостоятельной работы

Создаём копию файла «lab5-1.asm» с именем «lab5-1-1.asm» (рис. [-@fig:036]), (рис. [-@fig:037])



Создание копии файла



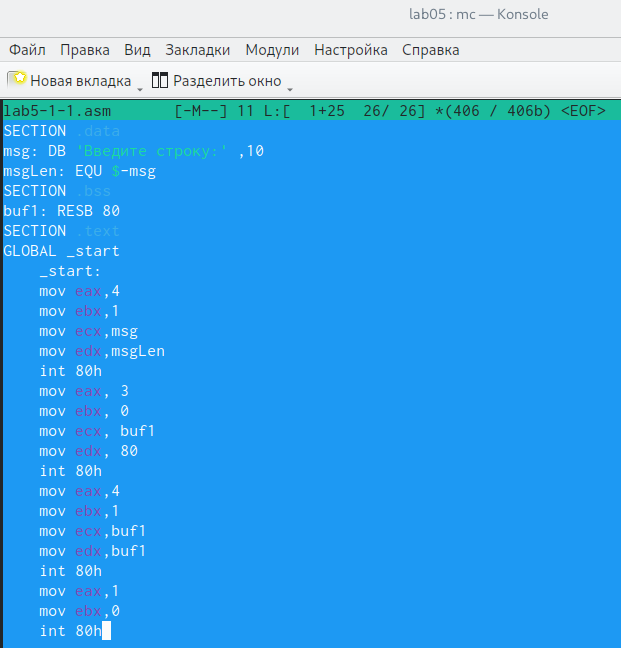
Проверка

Вносим 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; (рис. [-@fig:038])

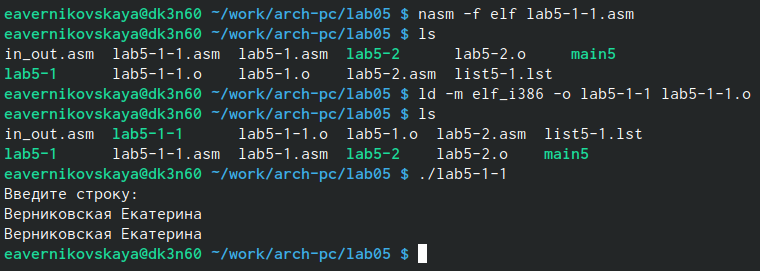
Текст программы:

SECTION .data   
msg: DB 'Введите строку:',10  
msgLen: EQU $-msg   
SECTION .bss   
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,1  
 mov ebx,0  
 int 80h



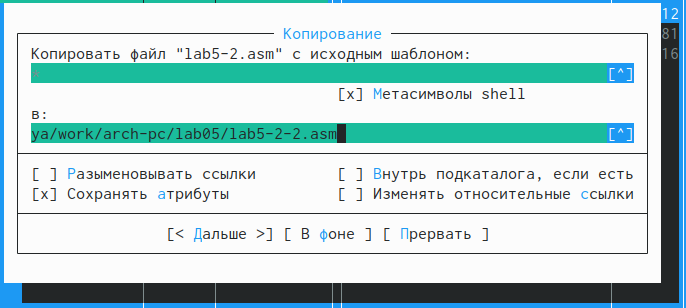
Текст программы

Получаем исполняемый файл и проверяем его работу. На приглашение вводим свою фамилию и имя (рис. [-@fig:039])

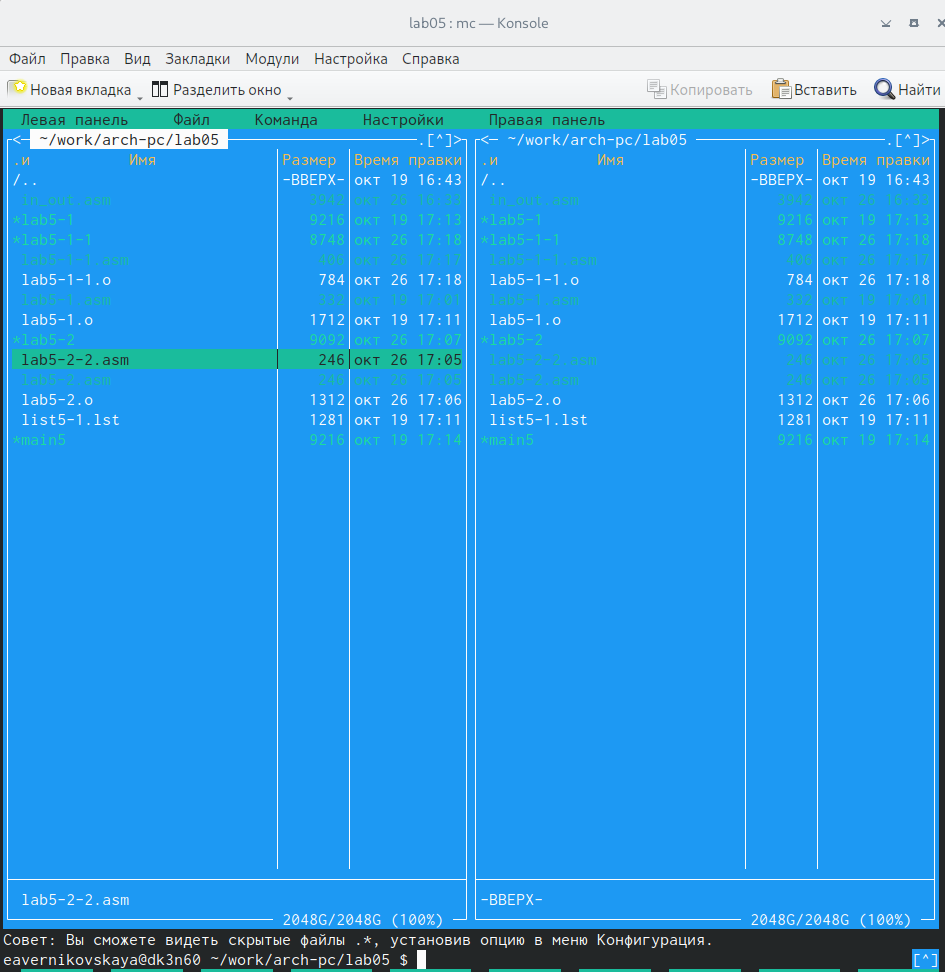


Проверка

Создаём копию файла «lab5-2.asm» с именем «lab5-2-2.asm» (рис. [-@fig:040]), (рис. [-@fig:041])



Создание копии файла



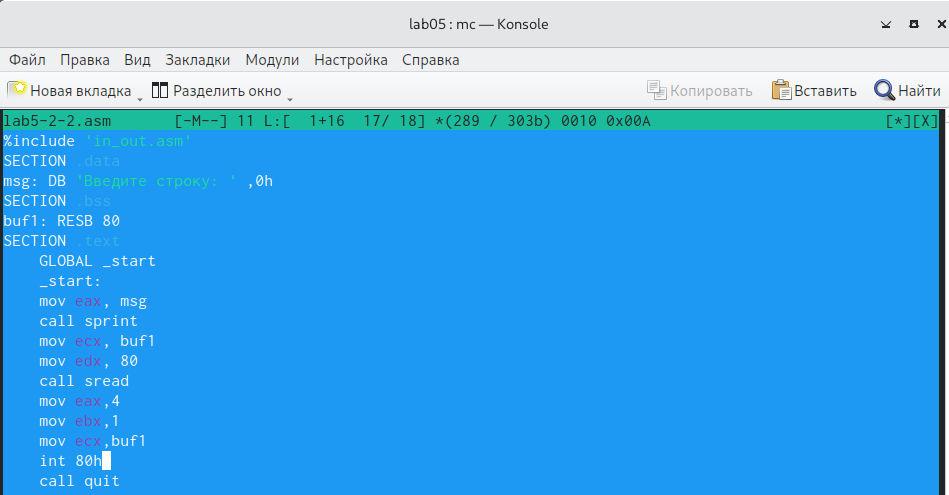
Проверка

Вносим изменения в программу (уже с использованием 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; (рис. [-@fig:042])

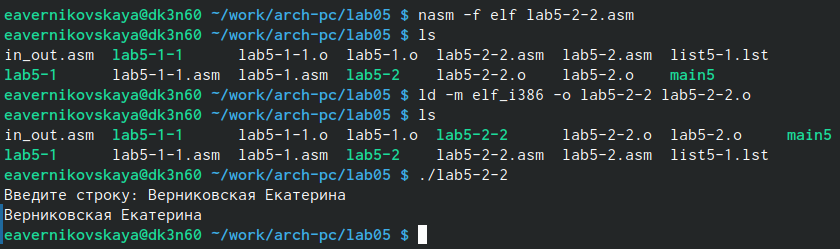
Текст программы:

%include 'in\_out.asm'   
SECTION .data   
msg: DB 'Введите строку:',10  
SECTION .bss   
buf1: RESB 80  
SECTION .text   
GLOBAL \_start   
 \_start:  
 mov eax, msg  
 call sprint  
 mov ecx, buf1  
 mov edx, 80  
 call sread  
 mov eax,4  
 mov ebx,1  
 mov ecx,buf1  
 int 80h  
 call quit



Текст программы

Создаём исполняемый файл и проверяем его работу (рис. [-@fig:043])



Проверка

# Выводы

В ходе выполнения лабораторной работы мы приобрели навыки работы в Midnight Commander, а так же освоили инструкции языка ассемблера mov и int.