

Лабораторная работа №5

Седохин Даниил Алексеевич

Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Задание для самостоятельной работы	16
4	Выводы	21

Список иллюстраций

2.1	Открытие Midnight Commander	5
2.2	Переход в каталог work/arch-pc	6
2.3	Создание папки lab05	7
2.4	Создание файла lab5-1.asm	8
2.5	Открытие файла lab5-1.asm для дальнейшего редактирования . .	9
2.6	Редактирование файла lab5-1.asm	10
2.7	Открытие файла lab5-1.asm, проверка содержания файла	11
2.8	Оттранслирование текста программы lab5-1.asm в объектный файл	11
2.9	Проверка файла	11
2.10	Копирование файла in_out.asm в каталог с файлом lab5-1.asm . .	12
2.11	Создание копии файла lab5-1.asm	13
2.12	Исправление текста программы в файле lab5-2.asm с использова- нием подпрограмм из in_out.asm	14
2.13	Создание и проверка исполняемого файла	14
2.14	Замена подпрограммы sprintLF на sprint в файле lab5-2.asm . . .	15
2.15	Создание и проверка исполняемого файла	15
3.1	Редактирование копии файла lab5-1.asm без использования in_out.asm согласно алгоритму	17
3.2	Создание и проверка исполняемого файла	18
3.3	Редактирование копии файла lab5-2.asm с использованием под- программ из файла in_out.asm согласно алгоритму	19
3.4	Создание и проверка исполняемого файла	20
3.5	Загрузка файлов на github	20

1 Цель работы

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

2 Выполнение лабораторной работы

1) Откроем Midnight Commander. (рис. 2.1).

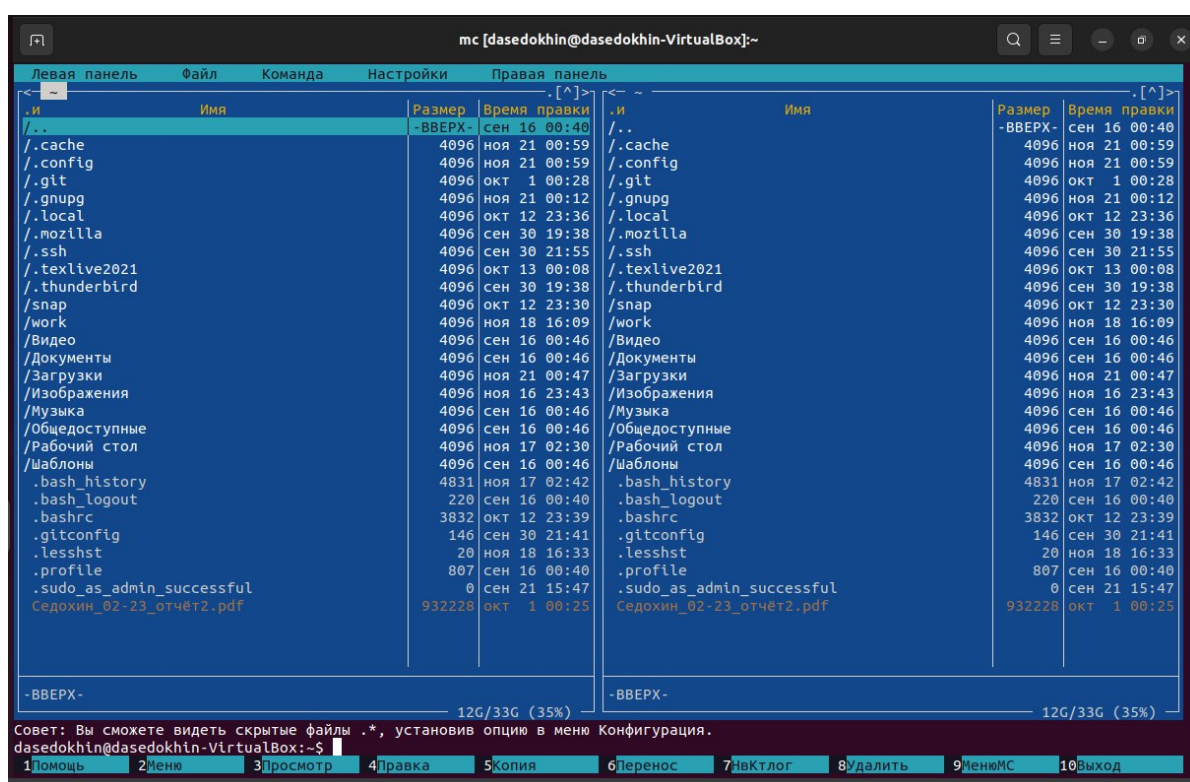


Рис. 2.1: Открытие Midnight Commander

2) Пользуясь клавишами **⌘**, **⌘** и Enter перейдём в каталог `~/work/arch-рс` созданный при выполнении лабораторной работы №4. (рис. 2.2).

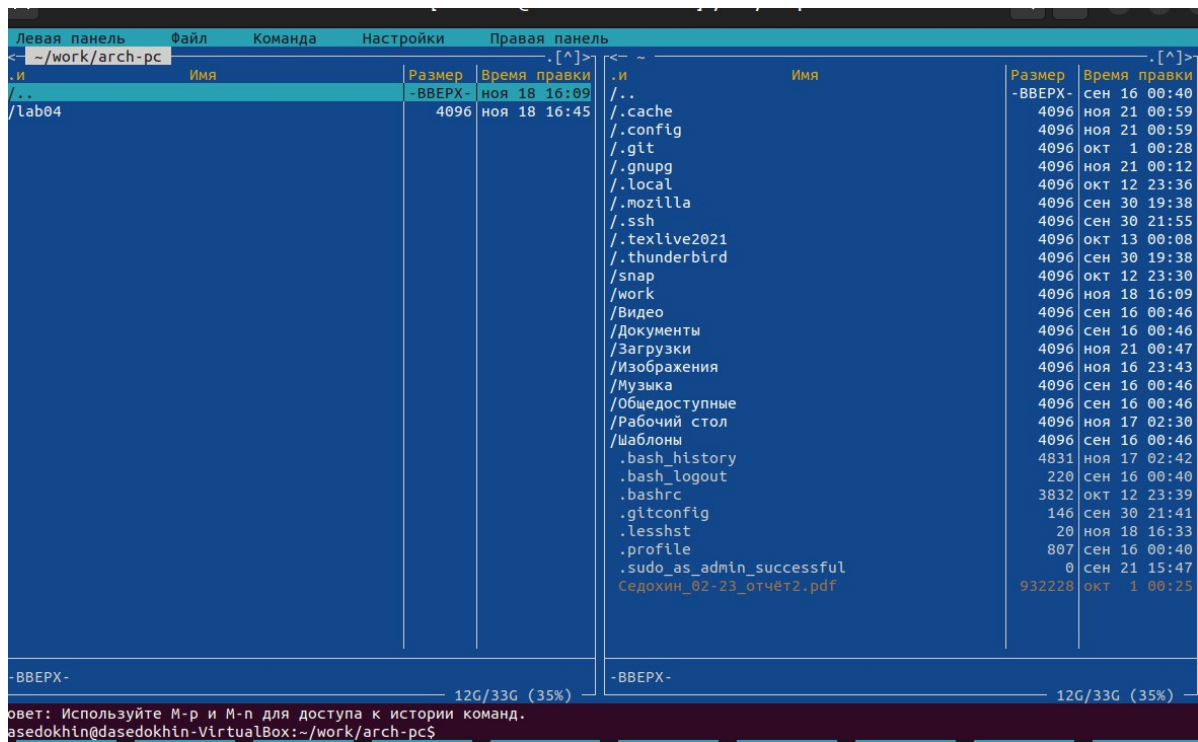


Рис. 2.2: Переход в каталог work/arch-pc

- 3) С помощью функциональной клавиши F7 создадим папку lab05 и перейдём в созданный каталог. (рис. 2.3).

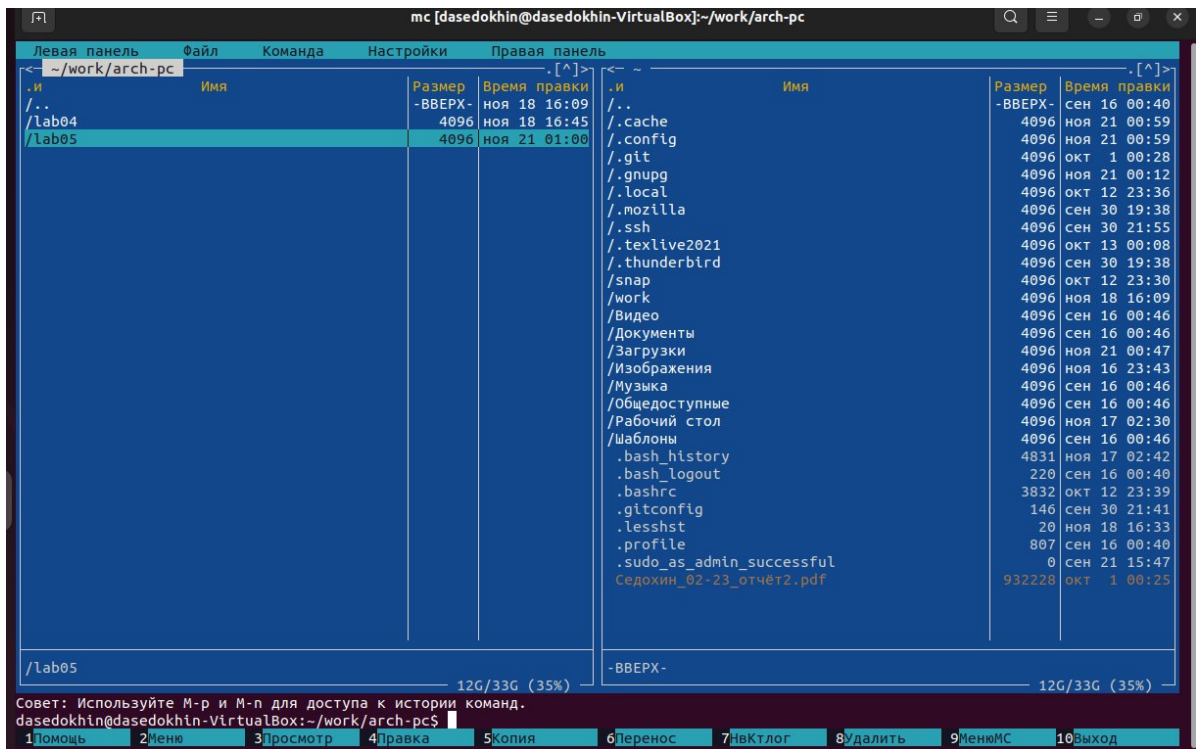


Рис. 2.3: Создание папки lab05

- 4) Пользуясь строкой ввода и командой touch создадим файл lab5-1.asm (рис. 2.4).

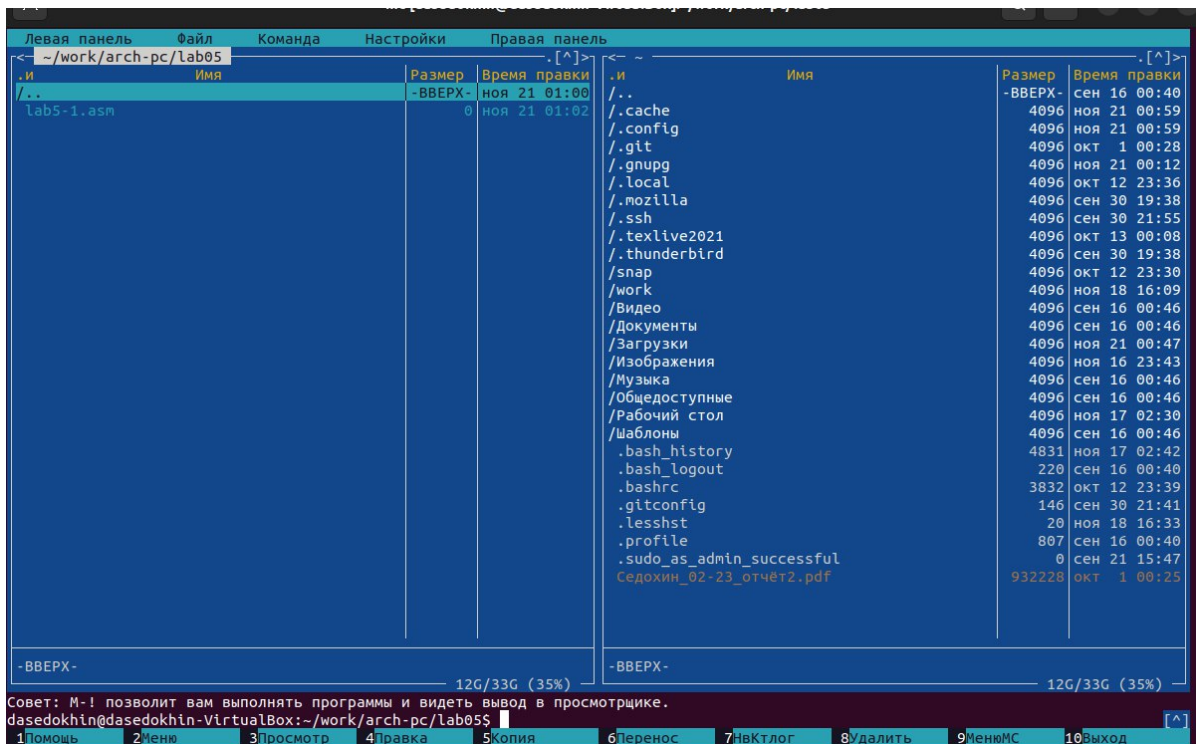


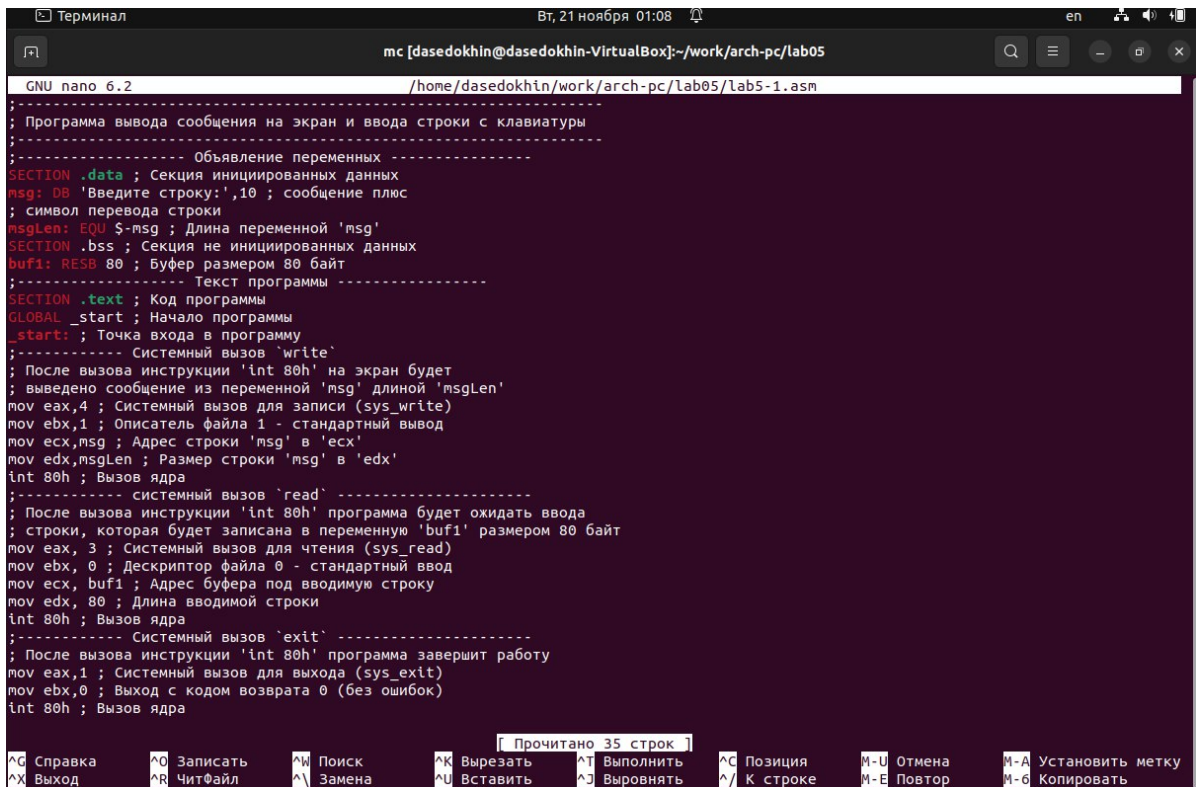
Рис. 2.4: Создание файла lab5-1.asm

- 5) С помощью функциональной клавиши F4 откроем файл lab5-1.asm для редактирования во встроенном редакторе. Как правило в качестве встроенного редактора Midnight Commander используется редакторы nano или mcedit. (рис. 2.5).



Рис. 2.5: Открытие файла lab5-1.asm для дальнейшего редактирования

- 6) Введём текст программы из листинга 5.1 , сохраним изменения и закроем файл. (рис. 2.6).

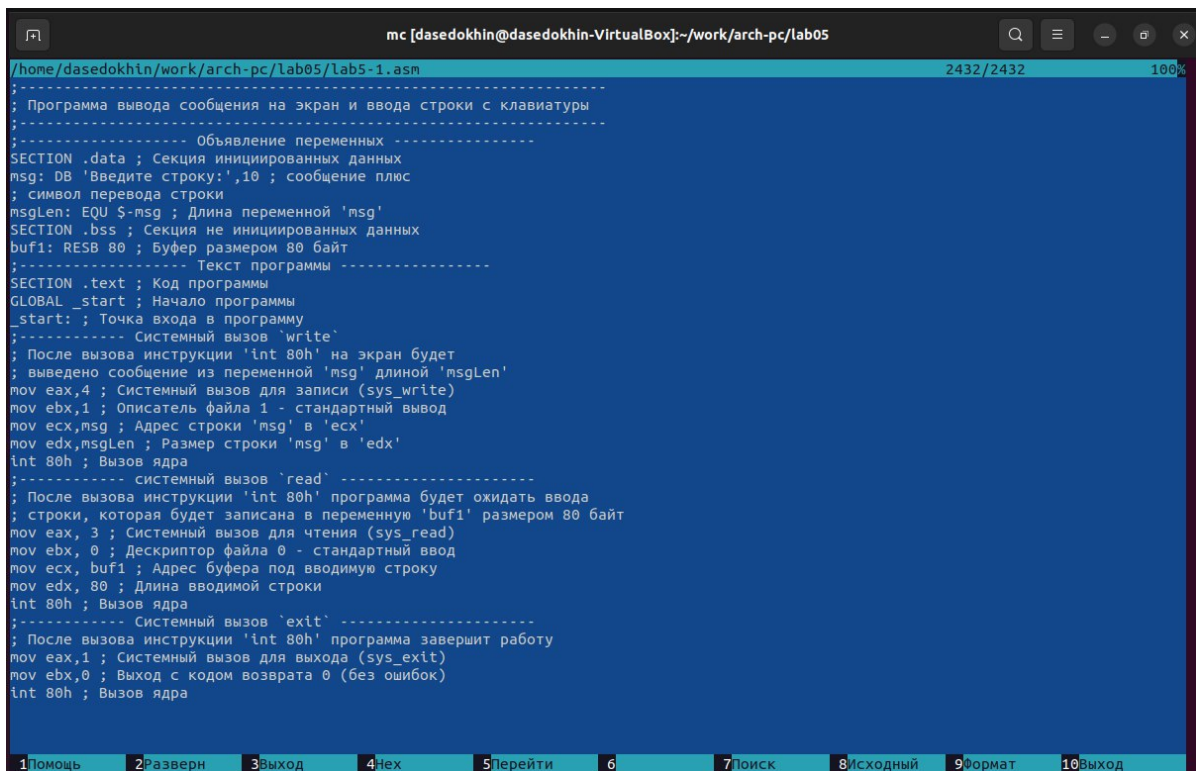


```
GNU nano 6.2 /home/dasedokhin/work/arch-pc/lab05/lab5-1.asm
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .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
;----- 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,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра

Прочитано 35 строк
^G Справка      ^O Записать    ^W Поиск       ^K Вырезать    ^T Выполнить   ^C Позиция     M-U Отмена     M-A Установить метку
^X Выход        ^R ЧитФайл    ^\ Замена      ^U Вставить    ^J Вывернуть   ^/_ К строке    M-E Повтор     M-6 Копировать
```

Рис. 2.6: Редактирование файла lab5-1.asm

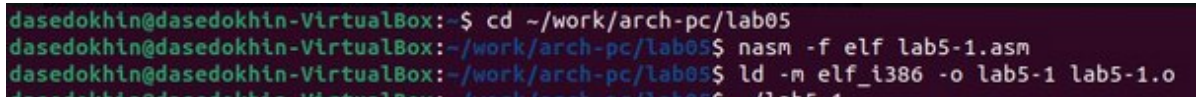
7) С помощью функциональной клавиши F3 откроем файл lab5-1.asm для просмотра. Убедимся, что файл содержит текст программы. (рис. 2.7).



```
mc [dasedokhin@dasedokhin-VirtualBox]:~/work/arch-pc/lab05
/home/dasedokhin/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 .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
;----- 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,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра
```

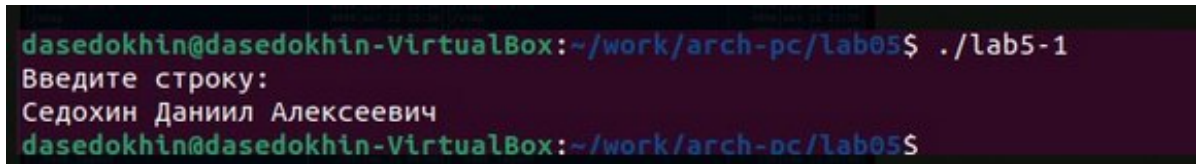
Рис. 2.7: Открытие файла lab5-1.asm, проверка содержания файла

8) Оттранслируем текст программы lab5-1.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл. Программа выводит строку ‘Введите строку:’ и ожидает ввода с клавиатуры. На запрос вводим ФИО (рис. 2.8 2.9).



```
dasedokhin@dasedokhin-VirtualBox:~$ cd ~/work/arch-pc/lab05
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
```

Рис. 2.8: Оттранслирование текста программы lab5-1.asm в объектный файл



```
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Седохин Даниил Алексеевич
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$
```

Рис. 2.9: Проверка файла

- 9) Скачаем файл `in_out.asm` со страницы курса в ТУИС. 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` откроем каталог с файлом `lab5-1.asm`. В другой панели каталог со скаченным файлом `in_out.asm` (для перемещения между панелями используем `Tab`). Скопируем файл `in_out.asm` в каталог с файлом `lab5-1.asm` с помощью функциональной клавиши `F5` (рис. 2.10).

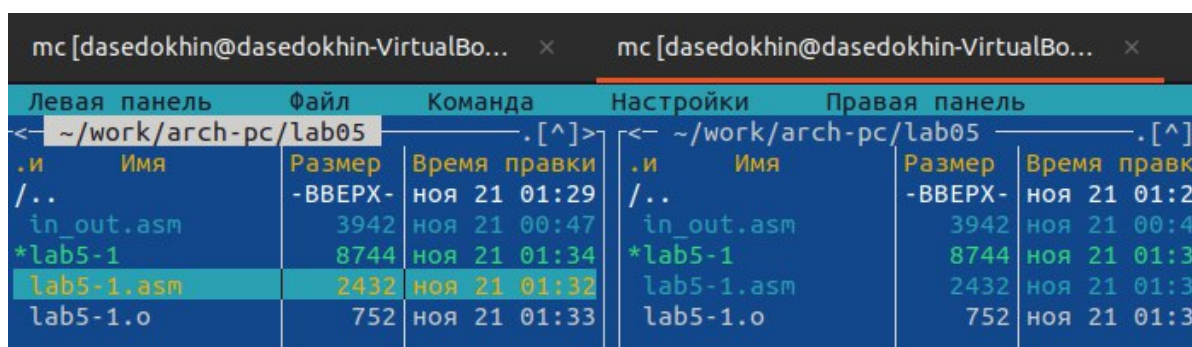


Рис. 2.10: Копирование файла `in_out.asm` в каталог с файлом `lab5-1.asm`

- 10) С помощью функциональной клавиши `F6` создадим копию файла `lab5-1.asm` с именем `lab5-2.asm`. (рис. 2.11).

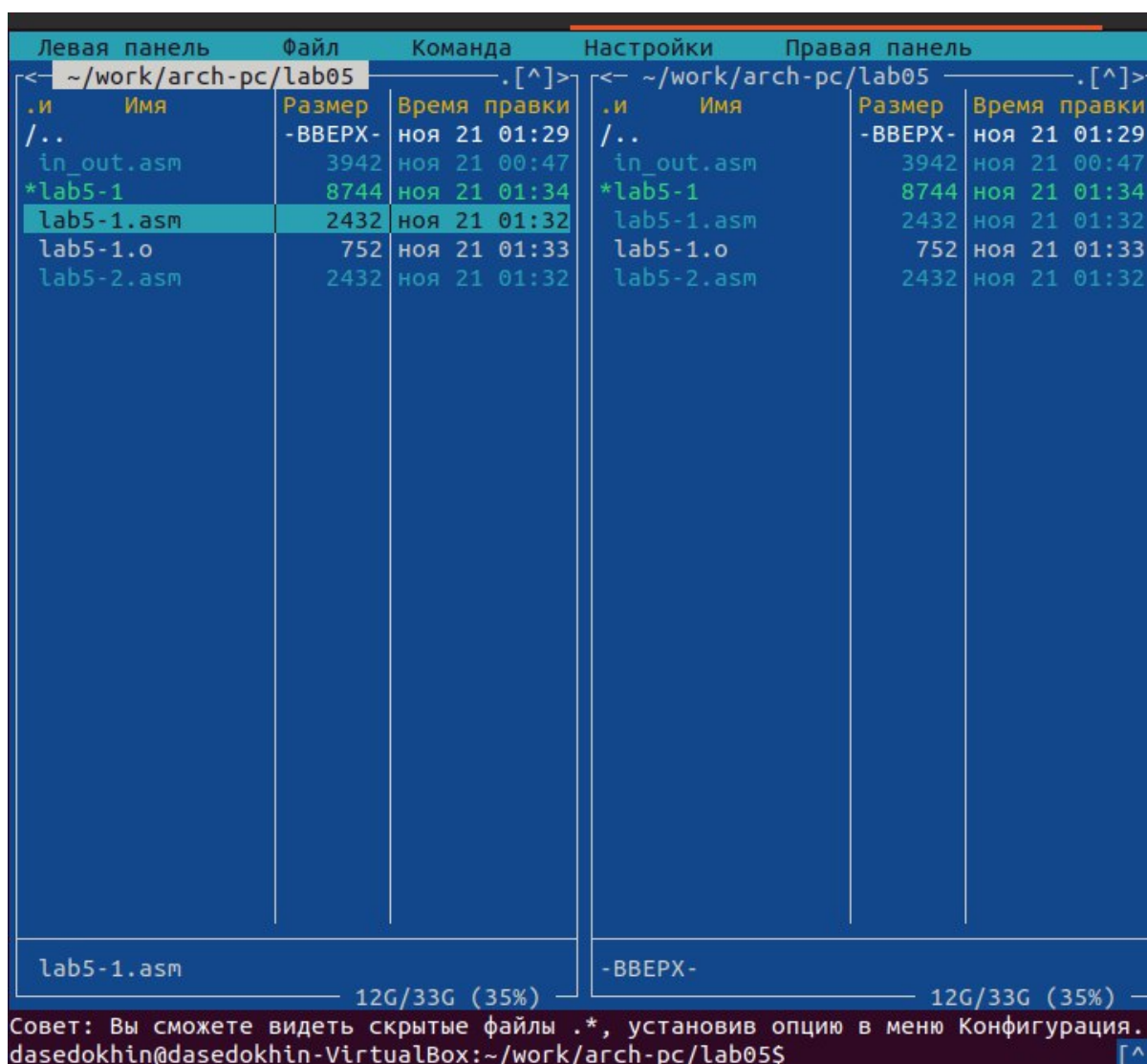
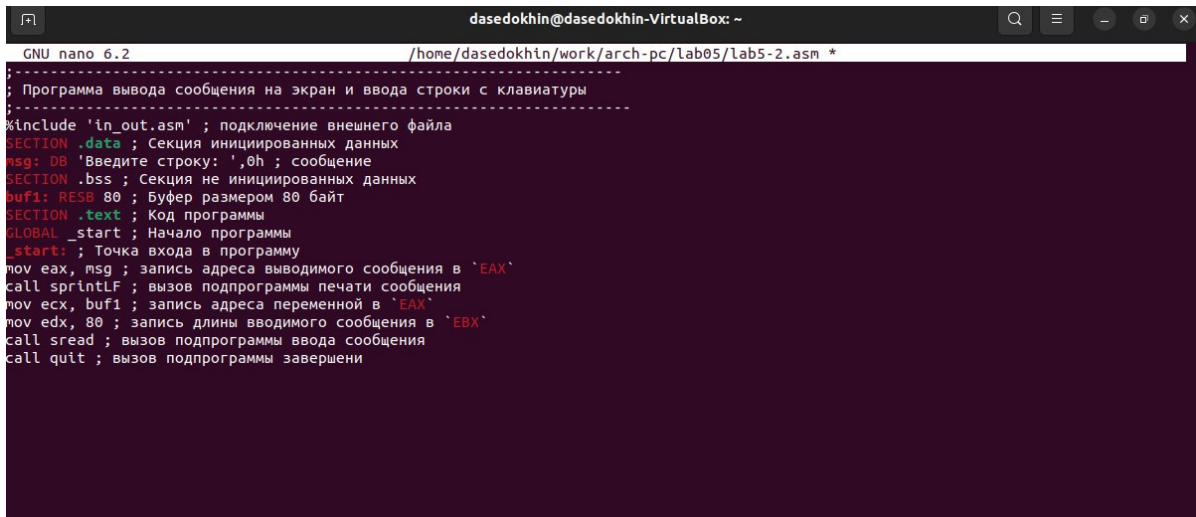


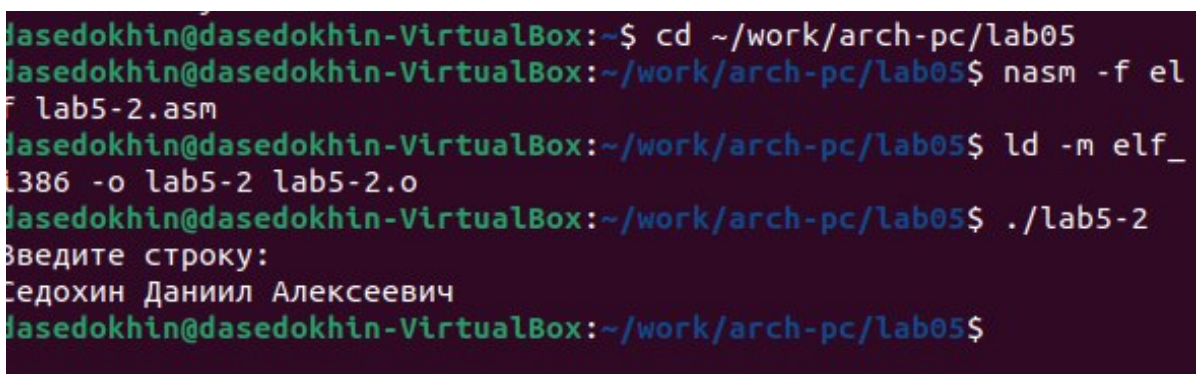
Рис. 2.11: Создание копии файла lab5-1.asm

- 11) Исправим текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in_out.asm (используем подпрограммы sprintLF, sread и quit) в соответствии с листингом 5.2. Создадим исполняемый файл и проверим его работу. (рис. 2.12 2.13).



```
GNU nano 6.2 /home/dasedokhin/work/arch-pc/lab05/lab5-2.asm *
; 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 .bss ; Секция не инициализированных 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 sprintf ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'EAX'
mov edx, 80 ; запись длины вводимого сообщения в 'EBX'
call read ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения
```

Рис. 2.12: Исправление текста программы в файле lab5-2.asm с использованием подпрограмм из in_out.asm



```
dasedokhin@dasedokhin-VirtualBox:~$ cd ~/work/arch-pc/lab05
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ nasm -f elf
lab5-2.asm
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ld -m elf_
386 -o lab5-2 lab5-2.o
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Седохин Даниил Алексеевич
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$
```

Рис. 2.13: Создание и проверка исполняемого файла

- 12) В файле lab5-2.asm заменим подпрограмму sprintf на sprint. Создадим исполняемый файл и проверим его работу. Разница sprintf и sprint заключается в том, что подпрограмма sprintf запрашивает ввод текста на следующей строке от выводимого сообщения, а sprint запрашивает ввод на той же строке. (рис. 2.14 2.15).

```
dasedokhin@dasedokhin-VirtualBox: ~/wo...
/home/dasedokhin/work/arch-pc/lab05/lab5-2.asm *
;----->
; 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 sprint ; 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 завершени
```

Рис. 2.14: Замена подпрограммы sprintLF на sprint в файле lab5-2.asm

```
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ nasm -f elf_
lab5-2.asm
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ld -m elf_
i386 -o lab5-2 lab5-2.o
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Седохин Даниил Алексеевич
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$
```

Рис. 2.15: Создание и проверка исполняемого файла

3 Задание для самостоятельной работы

1) Создадим копию файла lab5-1.asm. Внесём 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 введенную строку на экран.

Получим исполняемый файл и проверим его работу. (рис. 3.1 3.2).


```
mc [dasedokhin@dasedokhin-VirtualBox]:...
/home/dasedokhin/work/arch-pc/lab05/lab5-01.asm *
;-----
; 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
;----- 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
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
mov eax, 4 ;
mov ebx, 1 ;
mov ecx, buf1 ;
mov edx, 80 ;
int 80h ;
;----- Системный вызов `exit` -----
```

Рис. 3.1: Редактирование копии файла lab5-1.asm без использования in_out.asm согласно алгоритму

```
[5] 14 Остановлен MC
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ nasm -f elf
lab5-01.asm
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ld -m elf_
i386 -o lab5-01 lab5-01.o
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ./lab5-01
Введите строку:
Седохин Даниил Алексеевич
Седохин Даниил Алексеевич
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$
```

Рис. 3.2: Создание и проверка исполняемого файла

2) Создадим копию файла lab5-2.asm. Исправим 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 введенную строку на экран.

Создадим исполняемый файл и проверим его работу. (рис. 3.3 3.4).

```

/home/dasedokhin/work/arch-pc/lab05/lab5-02.asm *
;----->
; Программа вывода сообщения на экран и ввода строки с клавиат>
;----->
%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 прогЕрамму
mov eax, msg ; запись адреса выводимого сообщения в `EAX`
call sprint ; 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
mov eax, buf1 ;
call sprint ;
call quit ; вызов подпрограммы завершени

```

Рис. 3.3: Редактирование копии файла lab5-2.asm с использованием подпрограмм из файла in_out.asm согласно алгоритму


```

dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ nasm -f
elf lab5-02.asm
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ld -m el
f_i386 -o lab5-02 lab5-02.o
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$ ./lab5-0
2
Введите строку: Седохин Даниил Алексеевич
Седохин Даниил Алексеевич
dasedokhin@dasedokhin-VirtualBox:~/work/arch-pc/lab05$

```

Рис. 3.4: Создание и проверка исполняемого файла

3) Загрузим файлы на github. (рис. 3.5).

```

create mode 100644 labs/lab05/report/image/11.jpeg
create mode 100644 labs/lab05/report/image/12.jpeg
create mode 100644 labs/lab05/report/image/13.jpeg
create mode 100644 labs/lab05/report/image/14.jpeg
create mode 100644 labs/lab05/report/image/2.jpeg
create mode 100644 labs/lab05/report/image/20.jpeg
create mode 100644 labs/lab05/report/image/21.jpeg
create mode 100644 labs/lab05/report/image/22.jpeg
create mode 100644 labs/lab05/report/image/23.jpeg
create mode 100644 labs/lab05/report/image/24.jpeg
create mode 100644 labs/lab05/report/image/3.jpeg
create mode 100644 labs/lab05/report/image/4.jpeg
create mode 100644 labs/lab05/report/image/5.jpeg
create mode 100644 labs/lab05/report/image/6.jpeg
create mode 100644 labs/lab05/report/image/7.jpeg
create mode 100644 labs/lab05/report/image/8.jpeg
create mode 100644 labs/lab05/report/image/9.jpeg
create mode 100644 labs/lab05/report/report.docx
rewrite labs/lab05/report/report.md (72%)
create mode 100644 labs/lab05/report/report.pdf
dasedokhin@dasedokhin-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/a
rch-pc$ git push
Перечисление объектов: 60, готово.
Подсчет объектов: 100% (59/59), готово.
При сжатии изменений используется до 5 потоков
Сжатие объектов: 100% (52/52), готово.
Запись объектов: 100% (52/52), 5.70 МиБ | 3.41 МиБ/с, готово.
Всего 52 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:Danil2234/study_2023-2024_arh-pc.git
 0d0b661..394ded3 master -> master
dasedokhin@dasedokhin-VirtualBox:~/work/study/2023-2024/Архитектура компьютера/arch-pc$

```

Рис. 3.5: Загрузка файлов на github

4 Выводы

Я Приобрёл практические навыки работы в Midnight Commander. Освоил инструкции языка ассемблера mov и int.