

Лабораторная работа №5

Дисциплина: Архитектура компьютера

Первий Анастасия Андреевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
4.1	Подготовка пространства для выполнения лабораторной работы .	9
4.2	Программа вывода сообщения на экран и ввода строки с клавиатуры	10
4.3	Подключение внешнего файла in_out.asm	13
4.4	Программа вывода сообщений с использованием подключенного ранее файла	16
4.5	Самостоятельная работа. Создание исполняемого файла	19
5	Выводы	22
6	Листинги	23
	Список литературы	25

Список иллюстраций

4.1	Открываю Midnight Commander	9
4.2	Создаю папку lab05	10
4.3	Создаю файл lab5-1.asm	10
4.4	Выбираю редактор	11
4.5	Редактор nano	11
4.6	Текст программы в редакторе nano	12
4.7	Сохранение изменений	12
4.8	Проверка	13
4.9	Транслирую текст, выполняю компоновку и запускаю исполняемый файл lab5-1.asm	13
4.10	Копирование файла n_out.asm из домашнего каталога в каталог ~/work/arch-pc/lab05	15
4.11	Проверка копирования файла n_out.asm из домашнего каталога в каталог ~/work/arch-pc/lab05	15
4.12	Копирование файла lab5-1.asm с другим именем	16
4.13	Файл lab5-2.asm	17
4.14	Редактирование текста программы	17
4.15	Проверка текста программы	18
4.16	Транслирую текст, выполняю компоновку и запускаю исполняемый файл lab5-2.asm	18
4.17	Запускаю исправленный исполняемый файл lab5-2.asm	18
4.18	копирую файл lab5-2.asm	19
4.19	Измененный файл lab5-1.asm	20
4.20	Измененный файл lab5-2.asm	20
4.21	Программа из файла lab5-1.asm	21
4.22	Программа из файла lab5-2.asm	21

Список таблиц

1 Цель работы

Целью данной лабораторной работы является приобретение практических навыков работы в **Midnight Commander**. А также освоение инструкций языка ассемблера **mov** и **int**.

2 Задание

0. Подготовка пространства для выполнения лабораторной работы
1. Программа вывода сообщения на экран и ввода строки с клавиатуры
2. Подключение внешнего файла `in_out.asm`
3. Программа вывода сообщений с использованием подключенного ранее файла
4. Самостоятельная работа. Создание исполняемого файла

3 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.

Структура:

Программа на языке ассемблера **NASM**, как правило, состоит из трёх секций: секция кода программы (**SECTION .text**), секция инициализированных (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**).

Для объявления инициализированных данных в секции **.data** используются директивы **DB**, **DW**, **DD**, **DQ** и **DT**, которые резервируют память и указывают, какие значения должны храниться в этой памяти: • **DB** (define byte) — определяет переменную размером в 1 байт; • **DW** (define word) — определяет переменную размером в 2 байта (слово); • **DD** (define double word) — определяет переменную размером в 4 байта (двойное слово); • **DQ** (define quad word) — определяет переменную размером в 8 байт (учетверённое слово); • **DT** (define ten bytes) — определяет переменную размером в 10 байт.

Директивы используются для объявления простых переменных и для объявления массивов. Для определения строк принято использовать директиву **DB** в связи с особенностями хранения данных в оперативной памяти. Синтаксис директив определения данных следующий:

DB [,][,]

4 Выполнение лабораторной работы

Чтобы начать работать над выполнением лабораторной работы, целью которой является приобретение практических навыков работы в **Midnight Commander**, необходимо его открыть.

4.1 Подготовка пространства для выполнения лабораторной работы

Откроем терминал и впишем команду **mc**, необходимую для открытия **Midnight Commander**. (Рис.1 4.1)

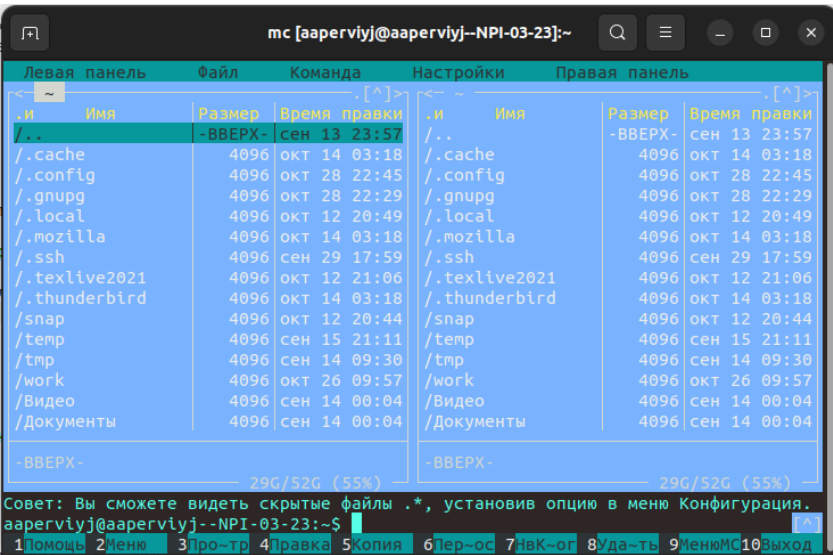


Рис. 4.1: Открываю Midnight Commander

Теперь нужно перейти в нужный каталог **~/work/arch-pc** и создать там папку **lab05** (Рис.2 4.2)

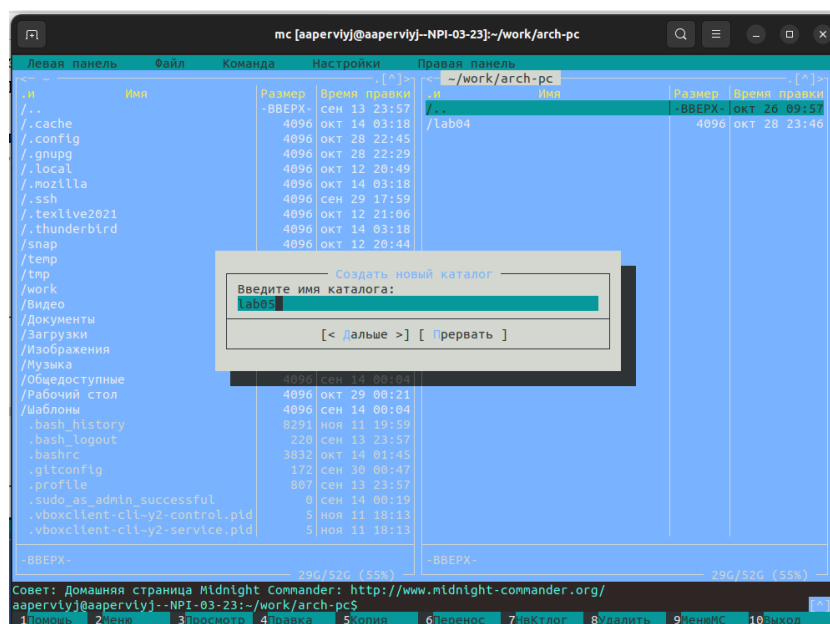


Рис. 4.2: Создаю папку lab05

Пользуясь строкой ввода и командой **touch**, необходимо создать файл **lab5-1.asm** (Рис.3 4.3)

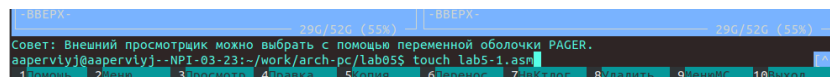


Рис. 4.3: Создаю файл lab5-1.asm

4.2 Программа вывода сообщения на экран и ввода строки с клавиатуры

С помощью функциональной клавиши **F4** необходимо открыть созданный ранее файл **lab5-1.asm** для редактирования во встроенном редакторе. Как правило в качестве встроенного редактора Midnight Commander используется

редакторы **nano** или **mcedit**. Я буду использовать редактор **nano**

Я нажала клавишу **F4** и открылся терминнал, где необходимо выбрать редактор (Рис.4 4.4)

```
mc [aaperviyj@aaperviyj--NPI-03-23]:~/work/arch-pc/lab05
aaperviyj@aaperviyj--NPI-03-23:~$ mc
aaperviyj@aaperviyj--NPI-03-23:~$ mc
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ touch lab5-1.asm
Select an editor. To change later, run 'select-editor'.
 1. /bin/nano          <---- easiest
 2. /usr/bin/mcedit
 3. /usr/bin/vim.tiny
 4. /bin/ed
Choose 1-4 [1]: 1
```

Рис. 4.4: Выбираю редактор

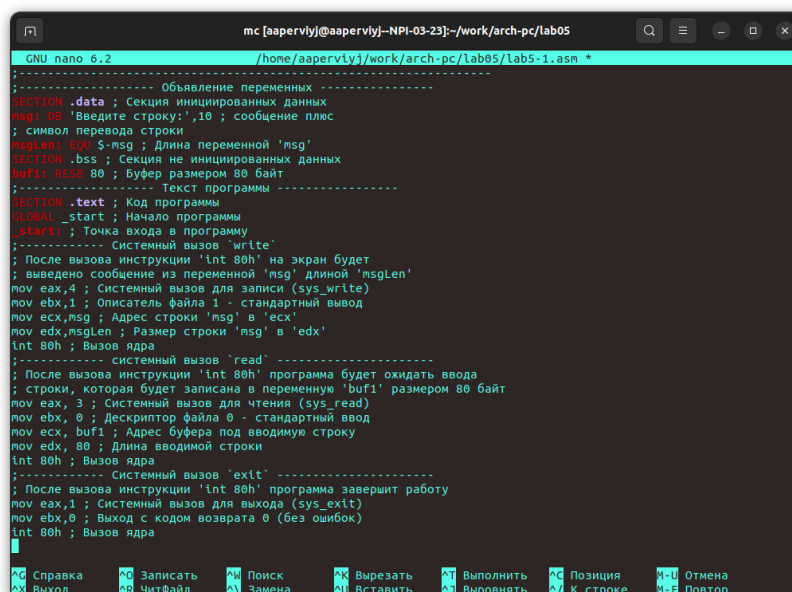
После того, как я ввела **1** и нажала клавишу **Enter** открылся сам файл, с помощью выбранного редактора **nano** (Рис.5 4.5)



Рис. 4.5: Редактор nano

Для создания *программы вывода сообщения на экран и ввода строки с клавиатуры* нужно ввести нужный текст из **листинга 5.1** (Список используемых листингов

приведен ниже). Копирую и вставляю текст программы (Рис.6 4.6)



```
GNU nano 6.2 /home/aapervlyj/work/arch-pc/lab05/lab5-1.asm *
;----- Объявление переменных -----
SECTION .data ; Секция иницированных данных
msg: db "Введите строку:",10 ; сообщение плюс
; символ перевода строки
msglen: equ $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не иницированных 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

^O Справка ^O Записать ^M Поиск ^X Вырезать ^T Выполнить ^C Позиция ^I-U Отмена
^X Выход ^R ЧитФайл ^M Замена ^U Вставить ^J Выводить ^J К строке ^I-E Повтор
```

Рис. 4.6: Текст программы в редакторе nano

Сохраняю изменения и закрываю редактор с помощью последовательности клавиш *Ctrl + X(выход) -> Y(сохранение изменений) -> Enter* (Рис.7 4.7)

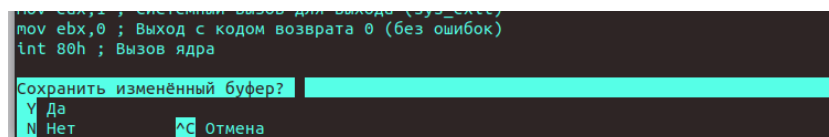


Рис. 4.7: Сохранение изменений

Проверяю сохранился ли изменения, с помощью клавиши **f3** (Рис.8 4.8)

```

mc [aapervlyj@aapervlyj--NPI-03-23]:~/work/arch-pc/lab05
/home/aapervlyj/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
1Помощь 2Заверн 3Выход 4Неч 5Грейти 6 7Поиск 8Исходный 9Формат 10Выход

```

Рис. 4.8: Проверка

Транслирую текст программы **lab5-1.asm** в объектный файл.Выполняю компоновку объектного файла и запускаю Ваши ФИО получившийся исполняемый файл. Программа выводит строку *‘Введите строку:’* и ожидает ввода с клавиатуры. На запрос ввожу ФИО (Рис.9 4.9)

```

aapervlyj@aapervlyj--NPI-03-23:~$ cd /home/aapervlyj/work/arch-pc/lab05
aapervlyj@aapervlyj--NPI-03-23:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
aapervlyj@aapervlyj--NPI-03-23:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
aapervlyj@aapervlyj--NPI-03-23:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Первый Анастасия Андреевна

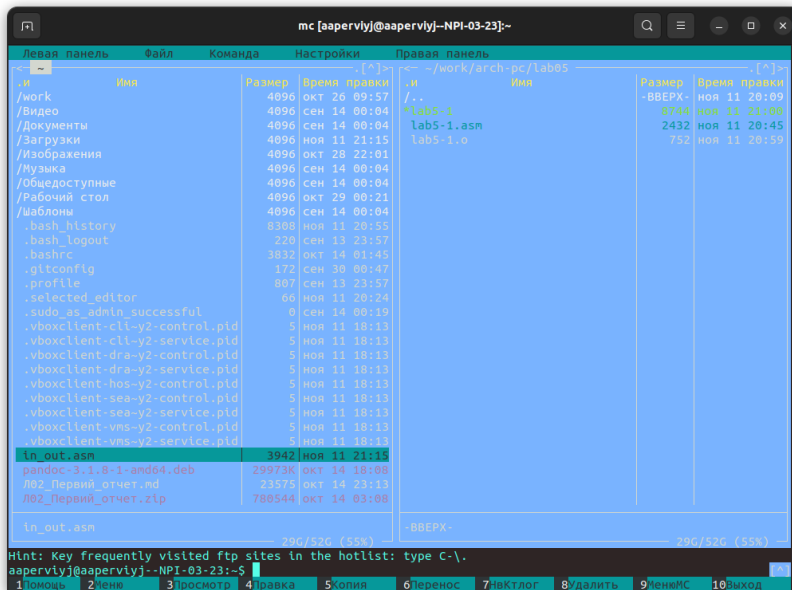
```

Рис. 4.9: Транслирую текст, выполняю компоновку и запускаю иполняемый файл lab5-1.asm

4.3 Подключение внешнего файла in_out.asm

Для выполнения данного пункта лабораторной работы необходимо скачать файл из ТУИС **n_out.asm**. Подключаемый файл должен лежать в том же каталоге, что и файл с программой, в которой он используется. Поэтому опять открываем **Midnight Commander** и на первой панели открываем

каталог, в котором находится файл, а на второй каталог, куда его нужно скопировать (Рис.10 ??) *Ctrl + X(выход) -> Y(сохранение изменений) -> Enter*



Теперь, когда все готово для копирования, нажимаю клавишу **F5**(с помощью этой клавиши файл можно скопировать из одного каталога в другой). (Рис.11 4.10)

4.4 Программа вывода сообщений с использованием подключенного ранее файла

Для выполнения данного пункта лабораторной работы необходимо создать копию файла **lab5-1.asm** под другим именем **lab5-2.asm**. Для этого нужно нажать клавишу **F6** и поменять название (Рис.13 4.12)

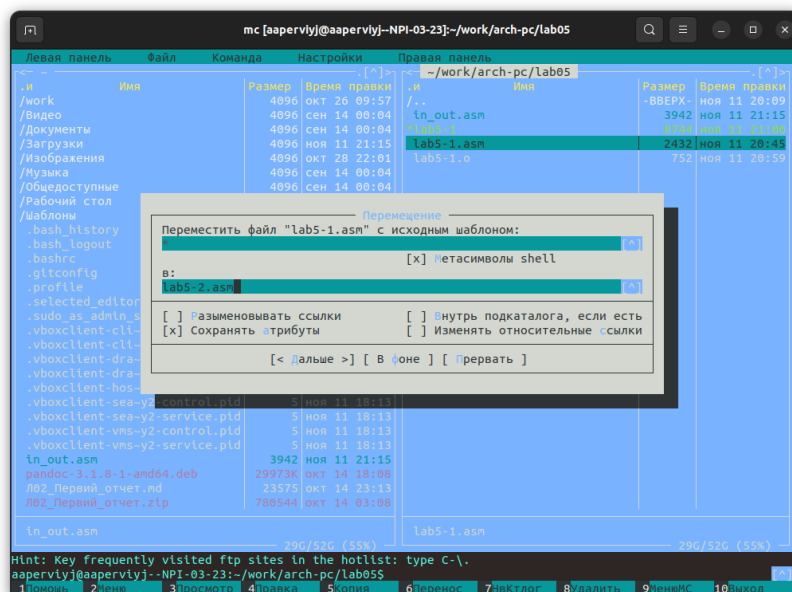


Рис. 4.12: Копирование файла lab5-1.asm с другим именем

Видим появившийся файл с именем **lab5-2.asm**, значит я сделала все правильно (Рис.14 4.13)

Имя	Размер	Время правки
./..	-ВВЕРХ-	ноя 11 20:09
in_out.asm	3942	ноя 11 21:15
*lab5-1	8744	ноя 11 21:00
lab5-1.o	752	ноя 11 20:59
lab5-2.asm	2432	ноя 11 20:45

Рис. 4.13: Файл lab5-2.asm

Так как новый файл **lab5-2.asm** это файл **lab5-1.asm** просто с другим названием, текст программы старый. Его необходимо исправить и вставить туда текст программы из **листинга 5.2**. Снова нажимаю клавишу **F4**, чтобы отредактировать файл и потом сохранить изменения с помощью комбинации клавиш **Ctrl + X(выход) -> Y(сохранение изменений) -> Enter** (Рис.15 4.14)

```

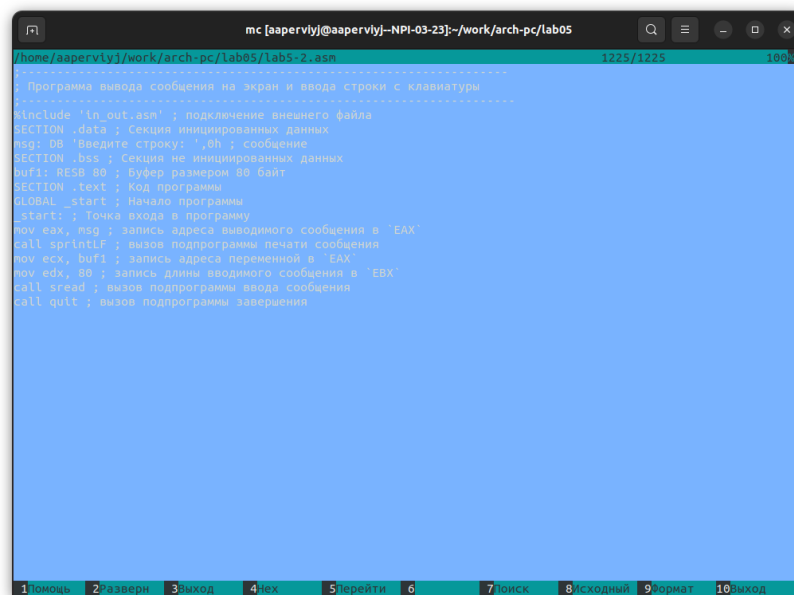
GNU nano 6.2 /home/aaperviy/work/arch-pc/lab05/lab5-2.asm
;.....
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;.....
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: db 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция 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 'ECX'
mov edx, 80 ; запись длины выводимого сообщения в 'EDX'
call sread ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения

Сохранить измененный буфер?
Y Да
N Нет  Ctrl+Q Отмена

```

Рис. 4.14: Редактирование текста программы

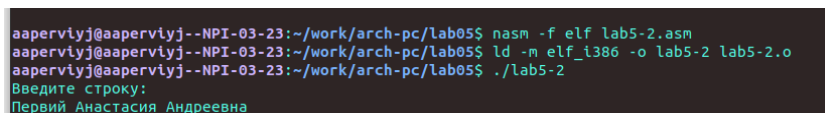
Проверяю сохранились ли изменения (Рис.16 4.15)



```
mc [aaperviyj@aaperviyj--NPI-03-23]~/work/arch-pc/lab05
/home/aaperviyj/work/arch-pc/lab05/lab5-2.asm 1225/1225 100%
; 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 'EDX'
call read ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения
```

Рис. 4.15: Проверка текста программы

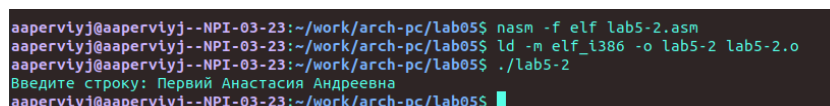
Транслирую текст программы **lab5-2.asm** в объектный файл.Выполняю компоновку объектного файла и запускаю Ваши ФИО получившийся исполняемый файл. Программа выводит строку 'Введите строку:' и ожидает ввода с клавиатуры. На запрос ввожу ФИО (Рис.17 4.16)



```
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Первый Анастасия Андреевна
```

Рис. 4.16: Транслирую текст, выполняю компоновку и запускаю исполняемый файл lab5-2.asm

В файле **lab5-2.asm** заменяю подпрограмму *sprintf* на *sprint*. Создаю исполняемый файл и проверяю его работу(Рис.18 4.17)



```
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ./lab5-2
Введите строку: Первый Анастасия Андреевна
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$
```

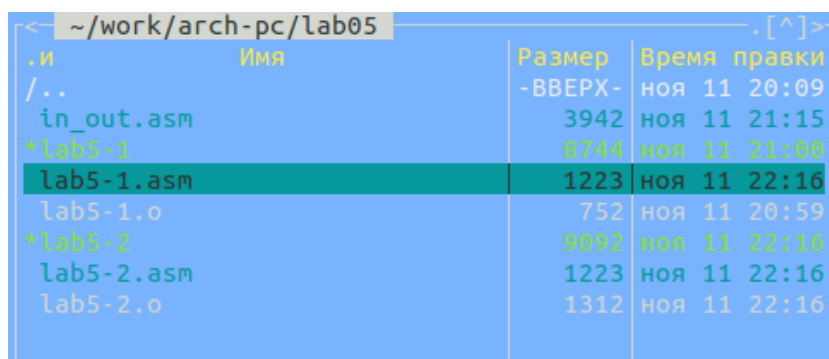
Рис. 4.17: Запускаю исправленный исполняемый файл lab5-2.asm

Сразу заметна разница при вводе сообщения. Подпрограмма *sprintLF* выводит сообщение с новой строки, а подпрограмма *sprint* на той же.

4.5 Самостоятельная работа. Создание исполняемого файла

В задании для самостоятельной работы необходимо создать копии файлов и внести изменения в программы, чтобы они работали следующим образом: • вывести приглашение типа “Введите строку:”; • ввести строку с клавиатуры; • вывести введенную строку на экран

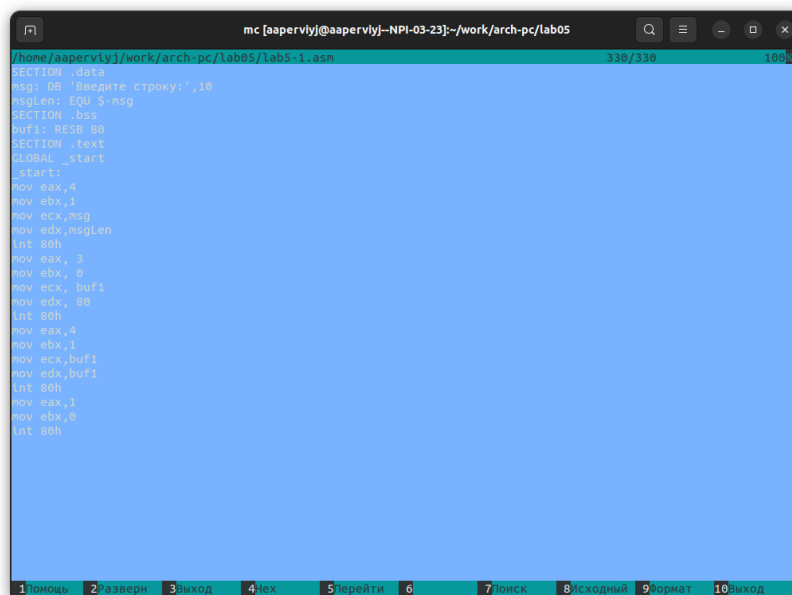
Копирую файл **lab5-2.asm** с именем **lab5-1.asm**. (Рис.19 4.18)



Имя	Размер	Время правки
./..	-ВВЕРХ-	ноя 11 20:09
in_out.asm	3942	ноя 11 21:15
*lab5-1	8744	ноя 11 21:00
lab5-1.asm	1223	ноя 11 22:16
lab5-1.o	752	ноя 11 20:59
*lab5-2	9092	ноя 11 22:16
lab5-2.asm	1223	ноя 11 22:16
lab5-2.o	1312	ноя 11 22:16

Рис. 4.18: копирую файл lab5-2.asm

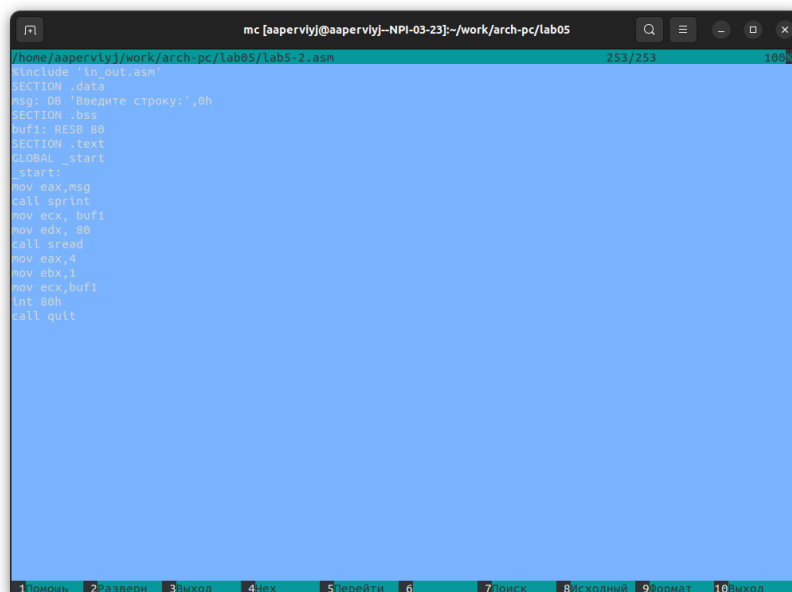
Вношу изменения в программу (без использования внешнего файла in_out.asm) и сохраняю. Делаю проверку с помощью клавиши **F3** (Рис.20 4.19)



```
mc [aapervlyj@aapervlyj-NPI-03-23]-~/work/arch-pc/lab05
/home/aapervlyj/work/arch-pc/lab05/lab5-1.asm 330/330 100%
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,4
mov ebx,1
mov ecx,buf1
mov edx,buf1
int 80h
mov eax,1
mov ebx,0
int 80h
1Помощь 2Заверн 3Выход 4lex 5Перейти 6 7Инск 8Сходный 9Формат 10Выход
```

Рис. 4.19: Измененный файл lab5-1.asm

То же самое делаю и с файлом **lab5-2.asm** (Рис.21 4.20)



```
mc [aapervlyj@aapervlyj-NPI-03-23]-~/work/arch-pc/lab05
/home/aapervlyj/work/arch-pc/lab05/lab5-2.asm 253/253 100%
#include 'fn_out.asm'
SECTION .data
msg: DB 'Введите строку:',0h
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
1Помощь 2Заверн 3Выход 4lex 5Перейти 6 7Инск 8Сходный 9Формат 10Выход
```

Рис. 4.20: Измененный файл lab5-2.asm

Создаю исполняемые файлы и проверяю их работу (Рис.22 4.21) и (Рис.23 4.22)

```
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Первый
Первый
```

Рис. 4.21: Программа из файла lab5-1.asm

```
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:Первый
Первый
aaperviyj@aaperviyj--NPI-03-23:~/work/arch-pc/lab05$
```

Рис. 4.22: Программа из файла lab5-2.asm

Программы работают исправно, заданий больше нет, лабораторная работа выполнена.

5 Выводы

В ходе выполнения лабораторной работы я приобрела практические навыки работы в **MidnightCommander** и освоила инструкции языка ассемблера *mov* и *int*

6 Листинги

Листинг 5.1. Программа вывода сообщения на экран и ввода строки с клавиатуры ;----- ; 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

Листинг 5.2. Программа вывода сообщения на экран и ввода строки с клавиатуры с использованием файла in_out.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 sprintf ; 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

```


Список литературы

Архитектура ЭВМ