

# **Лабораторная работа №6**

**НКАбд-03-22**

Шубнякова Дарья

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
<b>5</b>	<b>Выводы</b>	<b>14</b>
	<b>Список литературы</b>	<b>15</b>

## Список иллюстраций

4.1	Начало работы с Midnight Commander . . . . .	9
4.2	Редактор mcedit . . . . .	10
4.3	Запуск файла lab6-1 . . . . .	10
4.4	Внешний файл с подпрограммами . . . . .	11
4.5	Работа в Midnight Commander . . . . .	11
4.6	Редактор nano . . . . .	12
4.7	Запуск lab6-2 с подпрограммой sprintLF . . . . .	12
4.8	Запуск lab6-2 с подпрограммой sprint . . . . .	13

## **Список таблиц**

# 1 Цель работы

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

## 2 Задание

Создать два исполняемых файла. Один с использованием подпрограмм из внешнего файла, второй – без.

### 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.

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).

Для объявления инициированных данных в секции .data используются директивы DB, DW, DD, DQ и DT, которые резервируют память и указывают, какие значения должны храниться в этой памяти. 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. Для объявления неинициированных данных в секции .bss используются директивы resb, resw, resd и другие, которые сообщают ассемблеру, что необходимо зарезервировать заданное количество ячеек памяти.

Инструкция языка ассемблера mov предназначена для дублирования данных источника в приёмнике.

Инструкция языка ассемблера int предназначена для вызова прерывания с указанным номером. В общем виде она записывается в виде int n.

Простейший диалог с пользователем требует наличия двух функций — выво-

да текста на экран и ввода текста с клавиатуры. Простейший способ вывести строку на экран — использовать системный вызов `write`. Этот системный вызов имеет номер 4, поэтому перед вызовом инструкции `int` необходимо поместить значение 4 в регистр `eax`. Первым аргументом `write`, помещаемым в регистр `ebx`, задаётся дескриптор файла. Для вывода на экран в качестве дескриптора файла нужно указать 1 (это означает «стандартный вывод», т. е. вывод на экран). Вторым аргументом задаётся адрес выводимой строки (помещаем его в регистр `ecx`, например, инструкцией `mov ecx, msg`). Строка может иметь любую длину. Последним аргументом (т.е. в регистре `edx`) должна задаваться максимальная длина выводимой строки. Для ввода строки с клавиатуры можно использовать аналогичный системный вызов `read`. Его аргументы – такие же, как у вызова `write`, только для «чтения» с клавиатуры используется файловый дескриптор 0 (стандартный ввод). Системный вызов `exit` является обязательным в конце любой программы на языке ассемблер. Для обозначения конца программы перед вызовом инструкции `int 80h` необходимо поместить в регистр `eax` значение 1, а в регистр `ebx` код завершения 0.



## 4 Выполнение лабораторной работы

Открываем Midnight Commander с помощью команды `mcedit`. Создаем папку `lab06`, а в ней файл `lab6-1.asm` с помощью команды `touch`. (рис. 4.1)

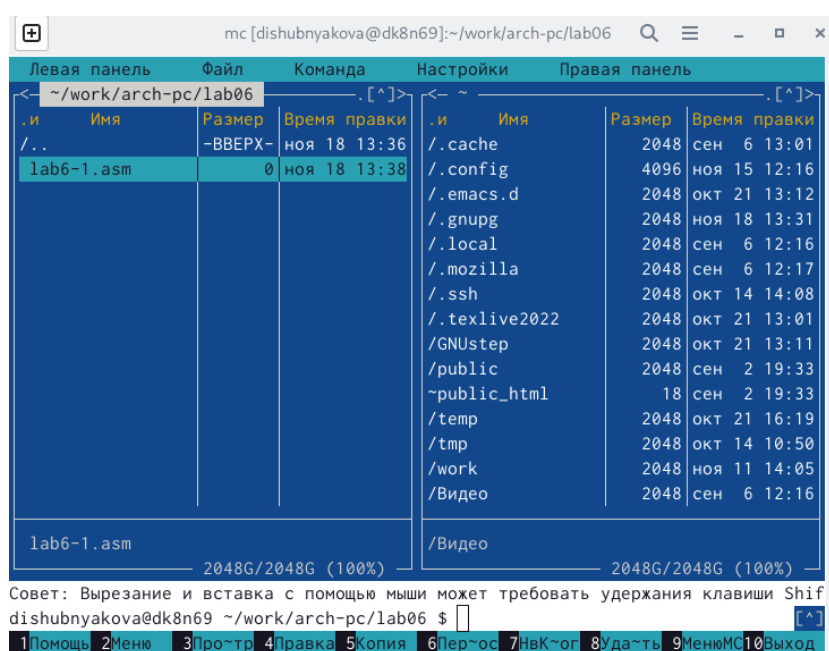
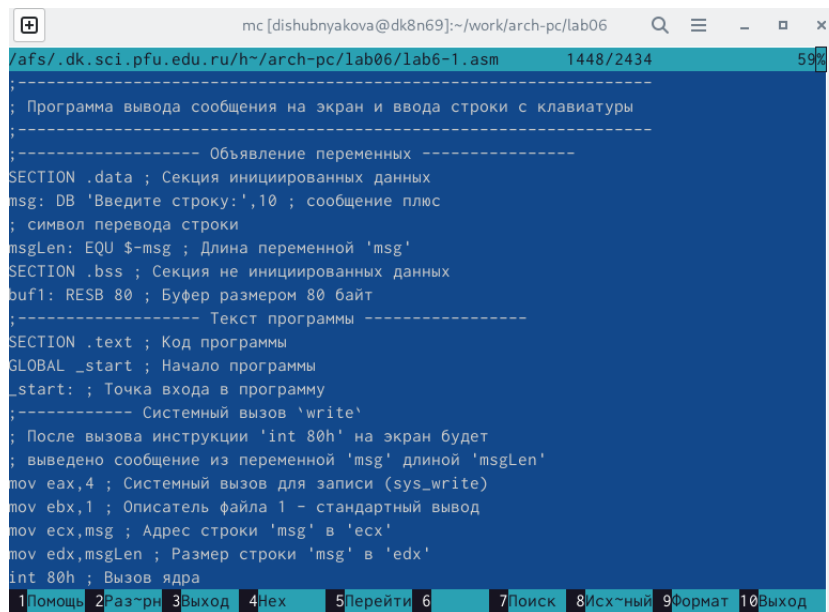


Рис. 4.1: Начало работы с Midnight Commander

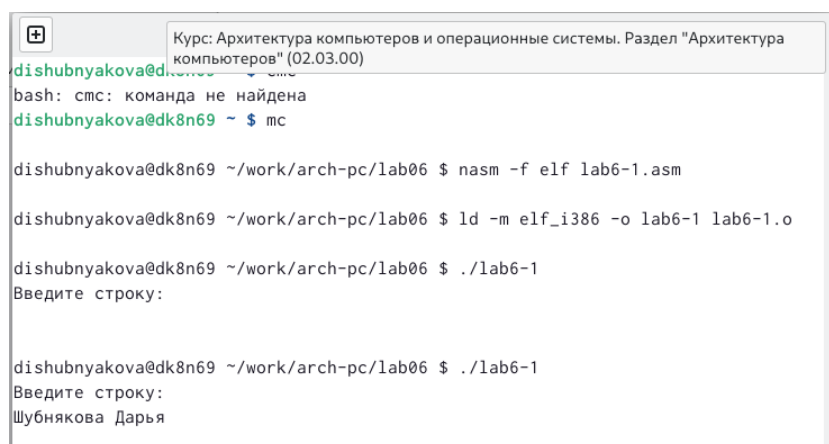
Редактируем файл в соответствии с заданием в редакторе `mcedit`. Сохраняем файл и закрываем редактор. С помощью клавиши `F3` проверяем правильность набранной программы. (рис. 4.2)



```
mc [dishubnyakova@dk8n69]:~/work/arch-pc/lab06
/afs/.dk.sci.pfu.edu.ru/h~/arch-pc/lab06/lab6-1.asm 1448/2434 59%
; 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
1Помощь 2Раз-рн 3Выход 4Нех 5Перейти 6 7Поиск 8Исх-ный 9Формат 10Выход
```

Рис. 4.2: Редактор mcedit

После того, как мы оттранслировали и провели компоновку объектного файла, запускаем исполняемый файл. На запрос “Введите строку” вводим имя и фамилию. (рис. 4.3)



```
Курс: Архитектура компьютеров и операционные системы. Раздел "Архитектура компьютеров" (02.03.00)
dishubnyakova@dk8n69 ~$ mc
bash: mc: команда не найдена
dishubnyakova@dk8n69 ~$ mc

dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm

dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o

dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-1
Введите строку:

dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-1
Введите строку:
Шубнякова Дарья
```

Рис. 4.3: Запуск файла lab6-1

Скачиваем внешний файл in\_out.asm с подпрограммами и копируем его в наш рабочий каталог. (рис. 4.4)

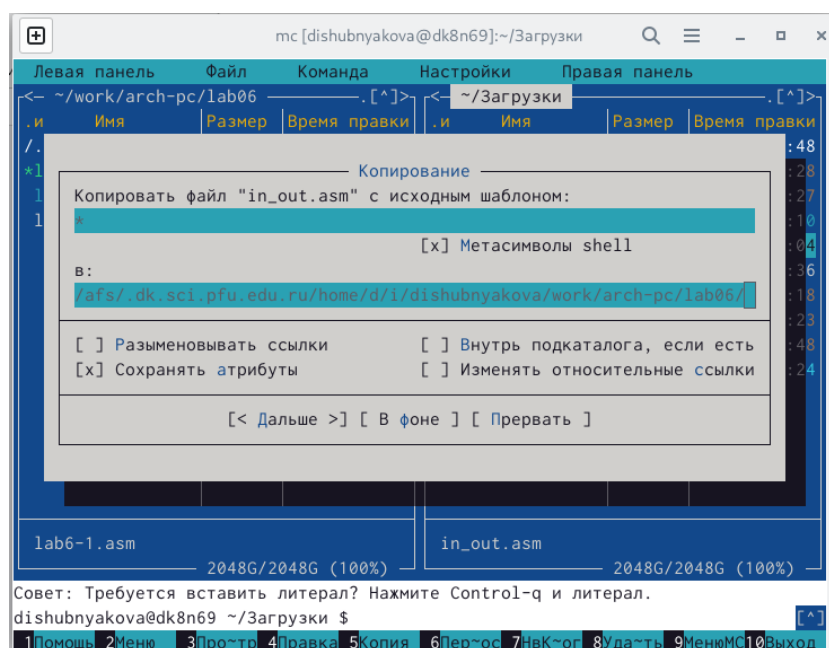


Рис. 4.4: Внешний файл с подпрограммами

Создаем копию нашего файла с названием lab6-2.asm. (рис. 4.5)

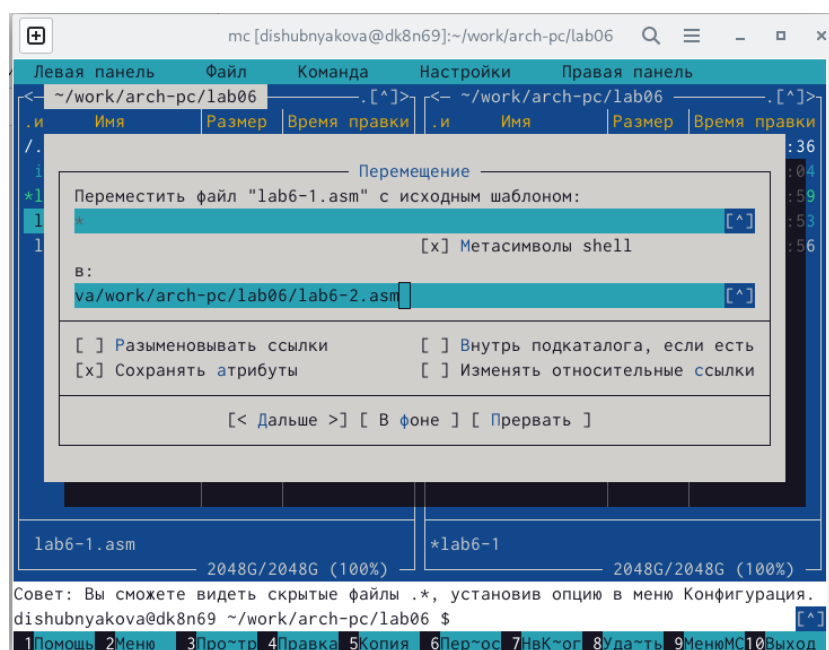


Рис. 4.5: Работа в Midnight Commander

Исправляем текст программы в nano на новый с использованием подпрограмм из скачанного ранее файла. Сохраним, оттранлируем и скомпилируем. (рис. 4.6)

```

mc [dishubnyakova@dk8n69]:~/work/arch-pc/lab06
...k.sci.pfu.edu.ru/home/d/i/dishubnyakova/work/arch-pc/lab06/lab6-2.asm Изменён
;-----
; 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 sread ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения

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

```

Рис. 4.6: Редактор nano

Запускаем исполняемый файл и проверяем, что все правильно работает. (рис. 4.7)

```

mc [dishubnyakova@dk8n69]:~/work/arch-pc/lab06
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-1
Введите строку:

dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-1
Введите строку:
Шубнякова Дарья

dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-1
Введите строку:

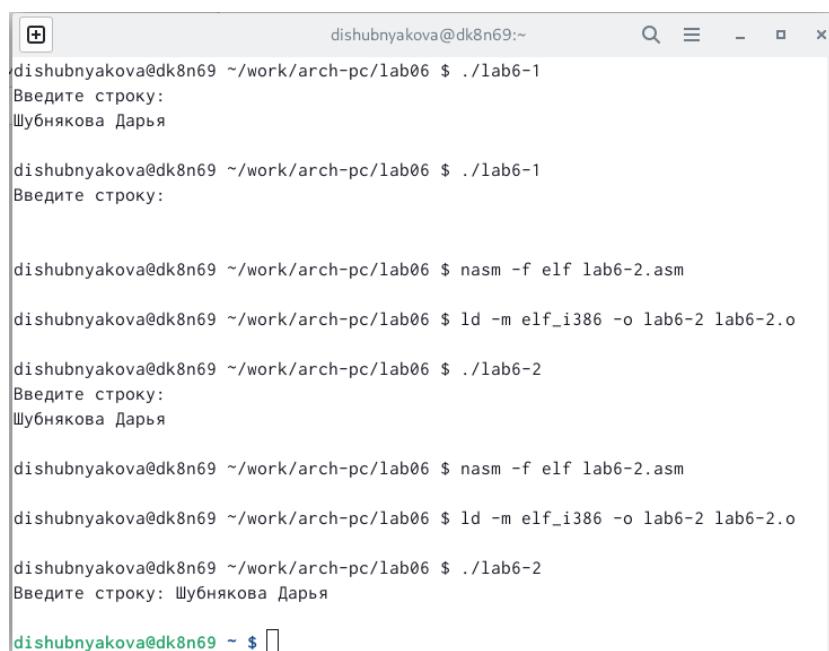
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-2
Введите строку:
Шубнякова Дарья

```

Рис. 4.7: Запуск lab6-2 с подпрограммой sprintf

В этом файле меняем подпрограмму sprintf на sprint. Выполняем те же дей-

ствия для получения исполняемого файла и проверяем его работу. Замечаем небольшую разницу между двумя этими подпрограммами. При использовании `sprintLF` после приглашения “Введите строку” был совершен переход на новую строку для пользовательского ввода данных. В случае `sprint` мы вводим имя и фамилию на этой же строке. (рис. 4.8)



```
dishubnyakova@dk8n69:~  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-1  
Введите строку:  
Шубнякова Дарья  
  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-1  
Введите строку:  
  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm  
  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o  
  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-2  
Введите строку:  
Шубнякова Дарья  
  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm  
  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o  
  
dishubnyakova@dk8n69 ~/work/arch-pc/lab06 $ ./lab6-2  
Введите строку: Шубнякова Дарья  
  
dishubnyakova@dk8n69 ~ $
```

Рис. 4.8: Запуск lab6-2 с подпрограммой `sprint`

## 5 Выводы

Мы познакомились с Midnight Commander (или просто mc). Узнали про структуру программы на языке ассемблера NASM. Прочитали про инструкции `mov` и `int`. Использовали внешний файл с подпрограммами для более удобной работы с NASM.

## **Список литературы**