

# **Отчет по лабораторной работе № 5**

**Дисциплина: Архитектура компьютера**

Дроздова Дарья Игоревна

# Содержание

1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Выводы	12

# 1 Цель работы

Целью данной лабораторной работы является освоение языка ассемблера `mov` и `int` и приобретение практических навыков работы в Midnight Commander.

## 2 Выполнение лабораторной работы

### 1. Порядок выполнения лабораторной работы

- С помощью команды `mc` открываем Midnight Commander и переходим в каталог `~/work/arch-пс` созданный при выполнении лабораторной работы №4:

```
mc [didrozdova@fedora]:~
Левая панель  Файл  Команда  Настройки  Правая панель
~  [^]  [^]
.и  Имя  Размер  Время правки  .и  Имя  Размер  Время правки
/..  -ВВЕРХ-  авг 9 16:57  /..  -ВВЕРХ-  авг 9 16:57
/.cabal  52  окт 22 17:06  /.cabal  52  окт 22 17:06
/.cache  554  ноя 23 10:31  /.cache  554  ноя 23 10:31
/.config  558  ноя 23 10:31  /.config  558  ноя 23 10:31
/.local  20  окт 22 15:42  /.local  20  окт 22 15:42
/.mozilla  48  окт 24 19:59  /.mozilla  48  окт 24 19:59
/.ssh  84  окт 22 21:43  /.ssh  84  окт 22 21:43
/.texlive2021  18  окт 22 21:45  /.texlive2021  18  окт 22 21:45
/work  24  ноя 4 19:52  /work  24  ноя 4 19:52
/Видео  50  окт 26 11:16  /Видео  50  окт 26 11:16
/Документы  0  окт 22 15:42  /Документы  0  окт 22 15:42
/Загрузки  1748  ноя 4 20:37  /Загрузки  1748  ноя 4 20:37
/Изображения  50  окт 24 00:23  /Изображения  50  окт 24 00:23
/Музыка  0  окт 22 15:42  /Музыка  0  окт 22 15:42
-ВВЕРХ-  -ВВЕРХ-
65G/79G (81%)  65G/79G (81%)
Совет: Вы сможете видеть скрытые файлы .*, установив опцию в меню Конфигурация.
didrozdova@fedora ~]$
1Помощь 2Меню 3Про-тр 4Правка 5Копия 6Пер-ос 7НвК-ог 8Уда-ть 9МенюМС 10Выход
```

- Создаем папку `lab06` с помощью командной клавиши `F7` и переходим в нее после создания:

~/work			
.и	Имя	Размер	Время правки
/..	-ВВЕРХ-	ноя 23 10:28	
/arch-pc	10	окт 26 23:04	
/study	18	окт 22 21:43	

~/work/arch-pc			
.и	Имя	Размер	Время правки
/..	-ВВЕРХ-	ноя 4 19:52	
/lab05	158	окт 26 23:21	

~/work/arch-pc			
.и	Имя	Размер	Время правки
/..	-ВВЕРХ-	ноя 4 19:52	
/lab05	158	окт 26 23:21	

Создать новый каталог

Введите имя каталога:

lab06

[< Дальше >] [Прервать]

.и	Имя	Размер	Время правки
/..	-ВВЕРХ-	авг 9 16:57	
/..	52	окт 22 17:06	
/..	554	ноя 23 10:31	
/..	558	ноя 23 10:31	
/..	20	окт 22 15:42	
/..	48	окт 24 19:59	
/..	84	окт 22 21:43	
/..	18	окт 22 21:45	
/..	24	ноя 4 19:52	
/..	50	окт 26 11:16	
/..	0	окт 22 15:42	
/Документы			
/Загрузки	1748	ноя 4 20:37	
/Изображения	50	окт 24 00:23	
/Музыка	0	окт 22 15:42	

- Пользуясь строкой ввода и командой touch, создаем файл lab6-1.asm:

~/work/arch-pc/lab06			
.и	Имя	Размер	Время правки
/..	-ВВЕРХ-	ноя 23 10:59	
lab6-1.asm	0	ноя 23 11:00	

- Открываем только что созданный файл для редактирования во встроенном редакторе *mcedit* и вводим текст программы, представленный в листинге документа, приложенного к данной лабораторной работе:

```

GNU nano 6.0 /home/didrozdova/work/arch-pc/lab06/lab6-1.asm Изменён
-----
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'

G Справка AO Записать AW Поиск AK Вырезать AT Выполнить AS Позиция
X Выход AR ЧитФайл AL Замена AU Вставить AJ Выровнять / К строке

```

- Сохраняем файл lab6-1.asm, закрываем его, а после открываем, чтобы убедиться, что текст программы сохранился.
- Далее оттранслируем текст программы lab6-1.asm в объектный файл, сделаем компоновку объектного файла и запустим получившийся исполняемый файл.

```

[didrozdova@fedora lab06]$ nasm -f elf lab6-1.asm
[didrozdova@fedora lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[didrozdova@fedora lab06]$ ./lab6-1
Введите строку:
Дроздова Дарья Игоревна
[didrozdova@fedora lab06]$

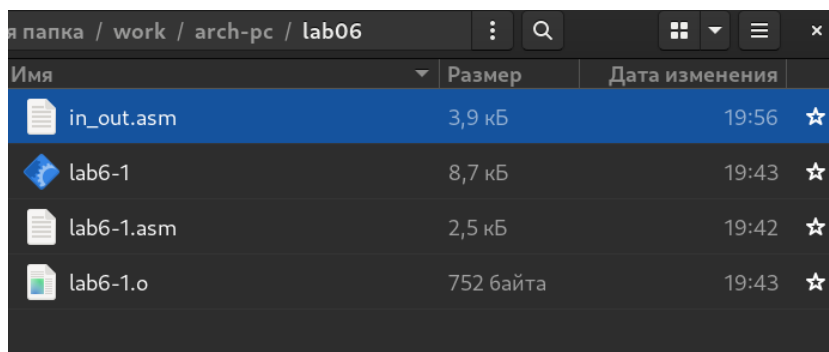
```

Программа выводит строку: 'Введите строку:', и ожидает ввода с клавиатуры. На запрос вводим собственные ФИО.

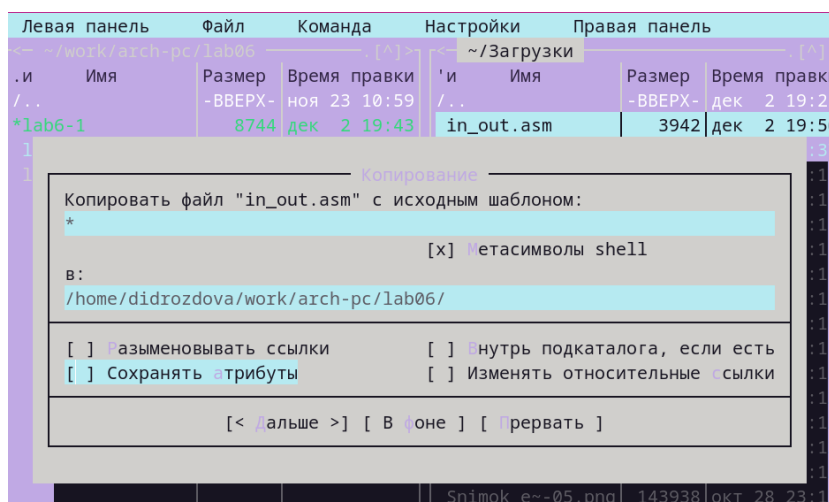
## 2. Подключение внешнего файла in\_out.asm

- Скачиваем файл in\_out.asm со страницы курса в ТУИС.
- Сохраняем подключаемый файл in\_out.asm в одном каталоге с файлом программы, в которой он используется.

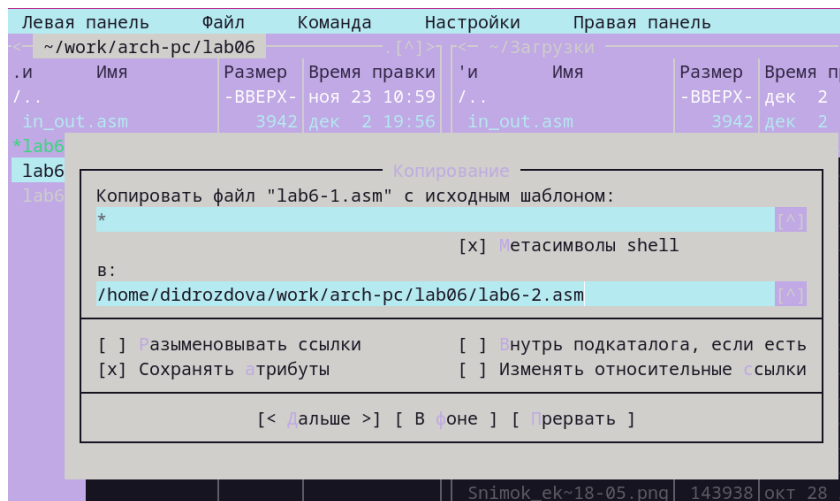
- В одной из панелей mc открываем каталог с файлом lab6-1.asm, в другой - каталог со скаченным файлом in\_out.asm:



Копируем файл in\_out.asm в каталог с файлом lab6-1.asm с помощью функциональной клавиши F5:



- С помощью функциональной клавиши F6 создаем копию файла lab6-1.asm с именем lab6-2.asm:



- Исправляем текст программы в файле lab6-2.asm с использованием подпрограмм из внешнего файла in\_out.asm в соответствии с листингом. Создаем исполняемый файл и проверяем его работу:

```
GNU nano 6.0 /home/didrozdova/work/arch-pc/lab06/lab6-2.asm Изм
-----
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

didrozdova@fedora lab06]$ nasm -f elf lab6-2.asm
didrozdova@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
didrozdova@fedora lab06]$ ./lab6-2
введите строку:
роздова Дарья Игоревна
didrozdova@fedora lab06]$ |
```

- В файле lab6-2.asm заменяем подпрограмму sprintf на sprint, создаем исполняемый файл и проверяем его работу:



```
GNU nano 6.0 /home/didrozdova/work/arch-pc/lab06/lab6-2.asm
-----
Программа вывода сообщения на экран и ввода строки с клавиатуры
-----
#include 'in_out.asm' ; подключение внешнего файла

SECTION .data ; Секция иницированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не иницированных 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 завершения

[didrozdova@fedora lab06]$ nasm -f elf lab6-2.asm
[didrozdova@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[didrozdova@fedora lab06]$ ./lab6-2
Введите строку: Дроздова Дарья Игоревна
```

Заметим разницу между командами `sprintLF` и `sprint`: в первом случае ввод осуществляется с новой строки, а во втором - сразу после выводимого текста.

### 3. Задание для самостоятельной работы

- Делаем копию файла `lab6-1.asm`, вносим изменения в программу, без использования команд из файла `in_out.asm`, чтобы программа работала по следующему алгоритму:
- вывести приглашение типа “Введите строку:”;
- ввести строку с клавиатуры;
- вывести введенную строку на экран.

```

;----- 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 ;Descriptor 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

```

^G Справка    ^O Записать    ^M Поиск    ^K Вырезать    ^T Выполнить    ^C Позиция  
 ^X Выход    ^R ЧитФайл    ^\ Замена    ^U Вставить    ^J Выровнять    ^\_ К строке

**Создаем исполняемый файл и проверяем его работу:**

```

[didrozdova@fedora lab06]$ nasm -f elf lab6-1_copy.asm
[didrozdova@fedora lab06]$ ld -m elf_i386 -o lab6-1_copy lab6-1_copy.o
[didrozdova@fedora lab06]$ ./lab6-1_copy
Введите строку:
Дроздова Дарья Игоревна
Дроздова Дарья Игоревна
[didrozdova@fedora lab06]$ |

[didrozdova@fedora lab06]$ ./lab6-1_copy
Введите строку:
Дроздова
Дроздова
[didrozdova@fedora lab06]$ |

```

- Создаем копию файла lab6-2.asm и исправляем текст программы 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 введенную строку на экран.

```

;-----
; 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
    mov eax, buf1
    call sprint
    call quit ; вызов подпрограммы завершения

```

Создаем исполняемый файл и проверяем его работу:

```

[drozdova@fedora lab06]$ nasm -f elf lab6-2_copy.asm
[drozdova@fedora lab06]$ ld -m elf_i386 -o lab6-2_copy lab6-2_copy.o
[drozdova@fedora lab06]$ ./lab6-2_copy
Введите строку: Дроздова
Дроздова
[drozdova@fedora lab06]$

```

## 3 Выводы

В ходе выполнения практических заданий “Лабораторной работы №5” я приобрела практические навыки работы в Midnight Commander и освоила инструкции языка ассемблера `mov` и `int`.