

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 5

дисциплина: Архитектура компьютера

Газизянов Владислав Альбертович

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Порядок выполнения лабораторной работы	6
3.2	Задание для самостоятельной работы	11
4	Выводы	15

Список иллюстраций

3.1	ms	6
3.2	Переходим в каталог	6
3.3	Создаем каталог	7
3.4	touch	7
3.5	Открываем файл, заполняем	8
3.6	Открываем файл и убеждаемся, что файл содержит текст программы	8
3.7	Проверка	9
3.8	Скачиваем файл	9
3.9	Копируем файл	9
3.10	Создаем копию файла	10
3.11	Проверяем скопировался ли файл	10
3.12	Открываем и заполняем файл	10
3.13	Редактируем файл	11
3.14	Смотрим, как работает программа и сравниваем с прошлой . . .	11
3.15	Создаем копию файла lab5-1.asm	11
3.16	Редактируем файл	12
3.17	Проверяем правильность	12
3.18	Создаем копию файла lab5-2.asm	13
3.19	Редактируем файл	13
3.20	Проверяем правильность программы	14

1 Цель работы

Освоить инструкции языка ассемблера mov. Приобрести знания использования Midnight Commander.

2 Задание

Написать 2 программы по примеру и изменить их структуру по условию.

3 Выполнение лабораторной работы

3.1 Порядок выполнения лабораторной работы

Открываем Mid. Commander

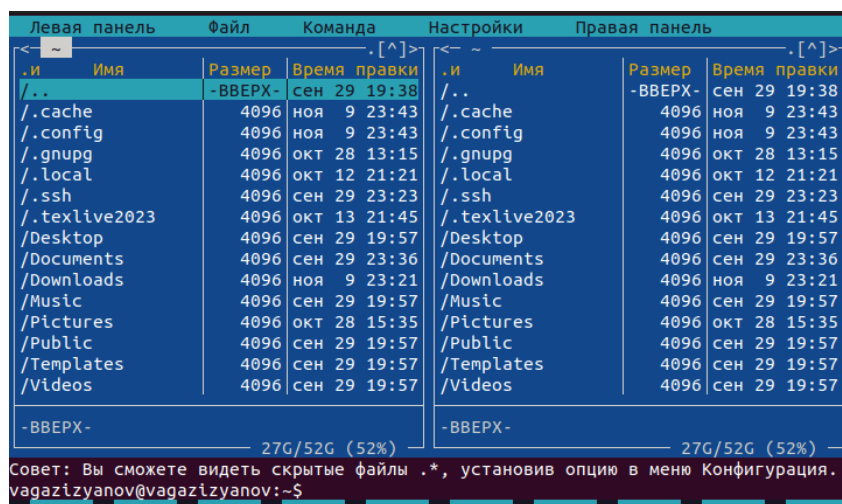


Рис. 3.1: mc

Переходим в каталог, созданный при выполнении 4 ЛБ

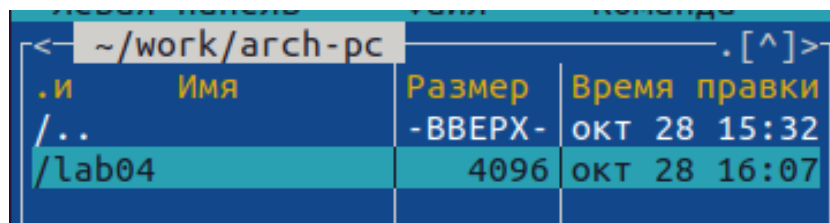
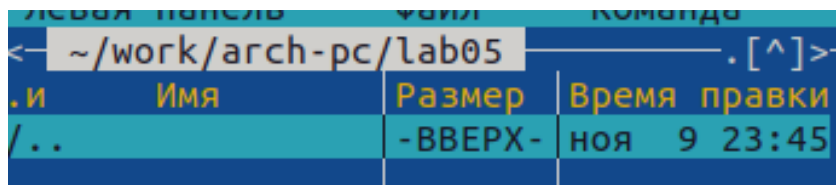


Рис. 3.2: Переходим в каталог

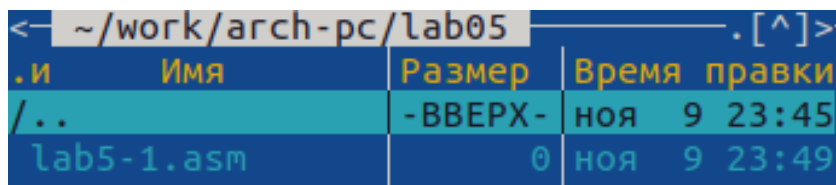
Создаем каталог lab05



<- ~/work/arch-pc/lab05 .[^]>-		
.и	Имя	Размер
/..	-ВВЕРХ-	Время правки
		ноя 9 23:45

Рис. 3.3: Создаем каталог

Создаем файл lab5-1.asm



<- ~/work/arch-pc/lab05 .[^]>-		
.и	Имя	Размер
/..	-ВВЕРХ-	Время правки
lab5-1.asm	0	ноя 9 23:49

Рис. 3.4: touch

Открываем файл для редактирования и заполняем его по листингу

```

GNU nano 6.2 /home/vagazizyanov/work/arch-pc/1
;-----
; 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 ;Descriptor 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. 3.5: Открываем файл, заполняем

Открываем файл и просматриваем

```

/home/vagazizyanov/work/arch-pc/1: cat 1.txt
;-----
; 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

```

Рис. 3.6: Открываем файл и убеждаемся, что файл содержит текст программы

Транслируем текст программы и запускаем файл

```
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
vagazizyanov
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$
```

Рис. 3.7: Проверка

Скачиваем файл

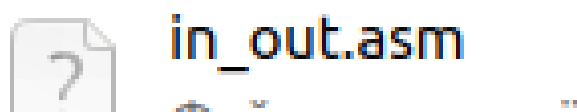


Рис. 3.8: Скачиваем файл

Копируем файл в нужную директорию

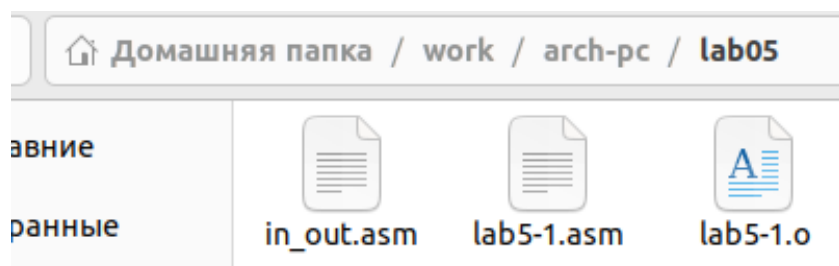


Рис. 3.9: Копируем файл

Создаем копию файла

~/work/arch-pc/lab05					.[^]>	
.и	Имя	Размер	Время	правки		
/..		-ВВЕРХ-	ноя 9	23:45		/.
	in_out.asm	3942	ноя 9	23:58		
	lab5-1.asm	2432	ноя 9	23:53		*T
	lab5-1.o	752	ноя 9	23:55		T
	*lab5-2.asm	8744	ноя 9	23:56		T

Рис. 3.10: Создаем копию файла

Проверяем созданный файл

```
GNU nano 6.2 /home/vagazizyanov/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 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 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
```

Рис. 3.11: Проверяем скопировался ли файл

Открываем новый файл и заполняем его

```
1 ;-----
2 ; Программа вывода сообщения на экран и ввода строки с клавиатуры
3 ;-----
4 #include 'in_out.asm' ; подключение внешнего файла
5 SECTION .data ; Секция иницированных данных
6 msg: DB 'Введите строку: ',0h ; сообщение
7 SECTION .bss ; Секция не иницированных данных
8 buf1: RESB 80 ; Буфер размером 80 байт
9 SECTION .text ; Код программы
0 GLOBAL _start ; Начало программы
1 _start: ; Точка входа в программу
2     mov eax, msg ; запись адреса выводимого сообщения в `EAX`
3     call sprintf ; вызов подпрограммы печати сообщения
4     mov ecx, buf1 ; запись адреса переменной в `EAX`
5     mov edx, 80 ; запись длины вводимого сообщения в `EBX`
6     call sread ; вызов подпрограммы ввода сообщения
7     call quit ; вызов подпрограммы завершения
```

Рис. 3.12: Открываем и заполняем файл

Открываем файл для редактирования и меняем `sprintLF` на `sprint`

```
call sprint ; вызов подпрограммы печати сообщения
```

Рис. 3.13: Редактируем файл

Транслируем и запускаем файл

```
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
vagazizyanov
```

Рис. 3.14: Смотрим, как работает программа и сравниваем с прошлой

3.2 Задание для самостоятельной работы

Создаем копию файла `lab5-1.asm`

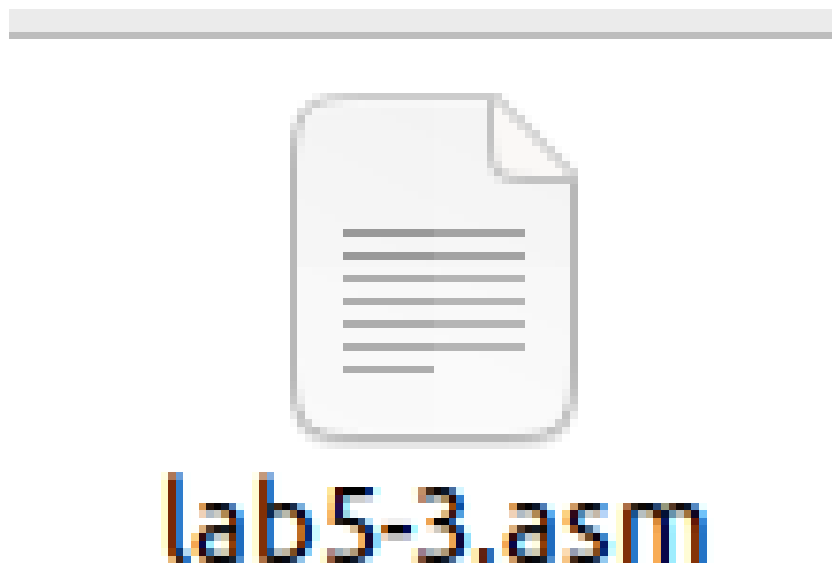


Рис. 3.15: Создаем копию файла `lab5-1.asm`

Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль

```

;----- 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
mov eax, 4 ; Системный вызов для записи (sys_write)
mov ebx, 1 ; Описатель файла 1 - стандартный вывод
mov ecx, buf1 ; Адрес строки 'msg' в 'ecx'
int 80h ; Вызов ядра

```

Рис. 3.16: Редактируем файл

Транслируем файл и запускаем программу

```

vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ nasm -f elf lab5-3.asm
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-3 lab5-3.o
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ ./lab5-3
Введите строку:
vagazizyanov
vagazizyanov

```

Рис. 3.17: Проверяем правильность

Создаем копию файла lab5-2.asm



Рис. 3.18: Создаем копию файла lab5-2.asm

Редактируем файл, чтобы введенный текст с клавиатуры выводился в консоль

```
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
```

Рис. 3.19: Редактируем файл

Транслируем файл и запускаем

```
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ nasm -f elf lab5-4.asm
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-4 lab5-4.o
vagazizyanov@vagazizyanov:~/work/arch-pc/lab05$ ./lab5-4
Введите строку: vagazizyanov
vagazizyanov
```

Рис. 3.20: Проверяем правильность программы

4 Выводы

Мы приобрели навыки работы с Midnight Commander и освоили инструкции mov.