Лабораторная работа №6. Основы работы с Midnight Commander (mc). Структура программы на языке ассемблера NASM.

Дисциплина: Архитектура ЭВИ

Осокин Георгий Иванович НММбд-02-22

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение

2 Выполнение лабораторной работы

2.1 Откроем через терминал MidnightCommander

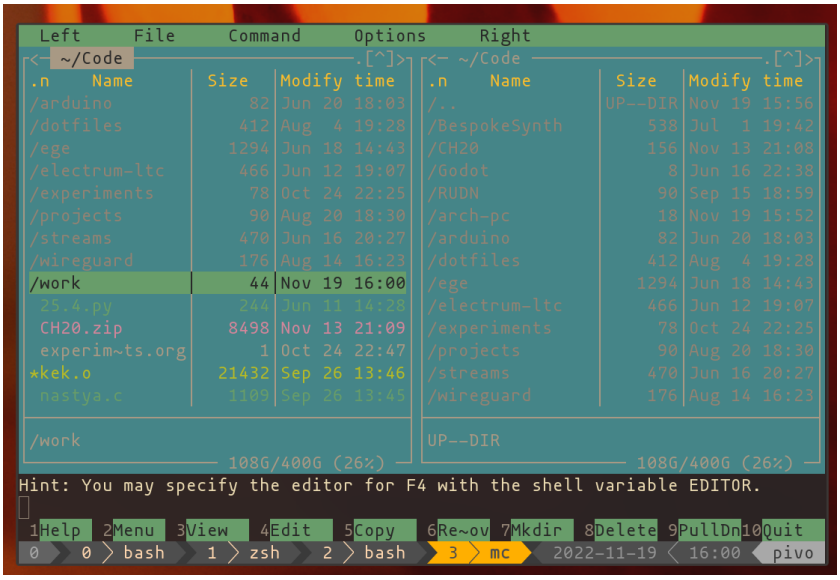

Рис. 2.1: Открытие Midnight Commander

Перейдем в каталог `./work/arch-pc` и создадим папку `./lab06`

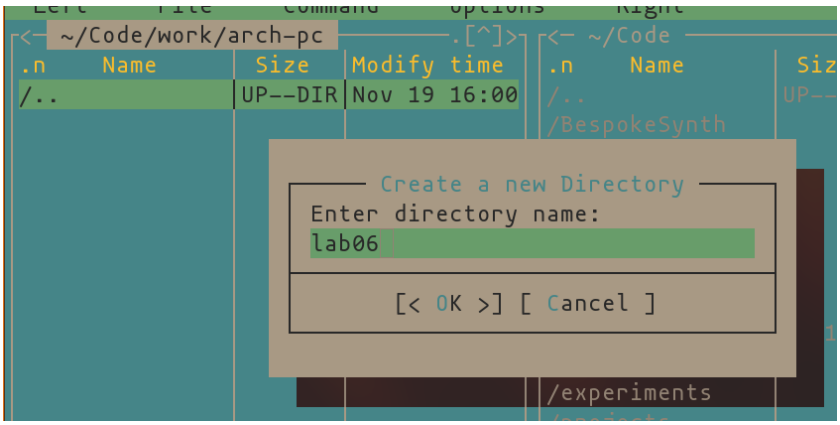

Рис. 2.2: Создание папки

Наберем в строке ввода `touch lab6.asm`

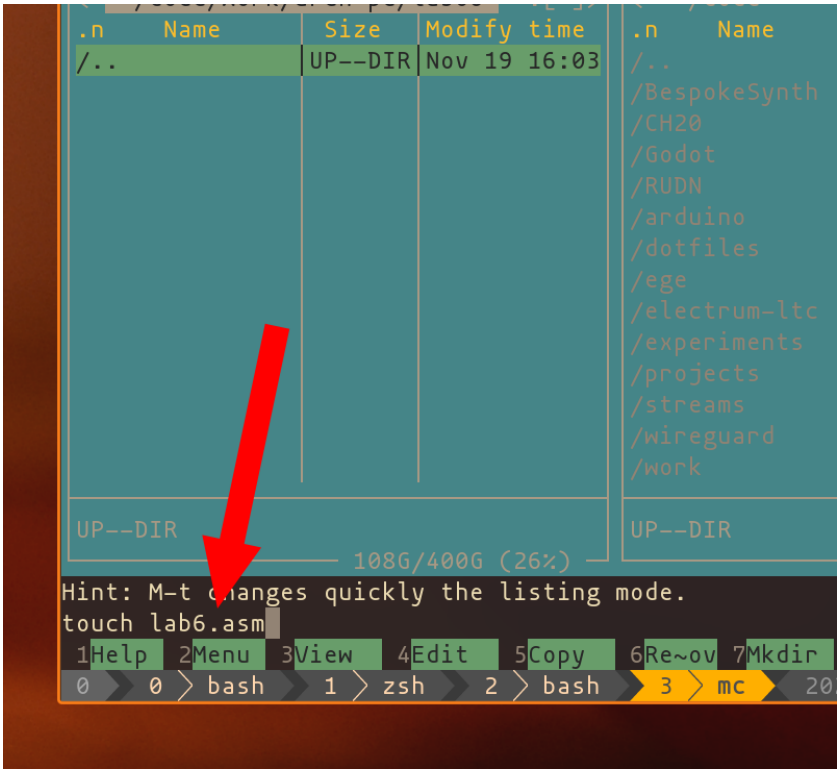

Рис. 2.3: Создание файла

2.2 Для своего удобства я изменю текстовый редактор по умолчанию на VIM

Зайдем в настройки, нажав F9, o, c

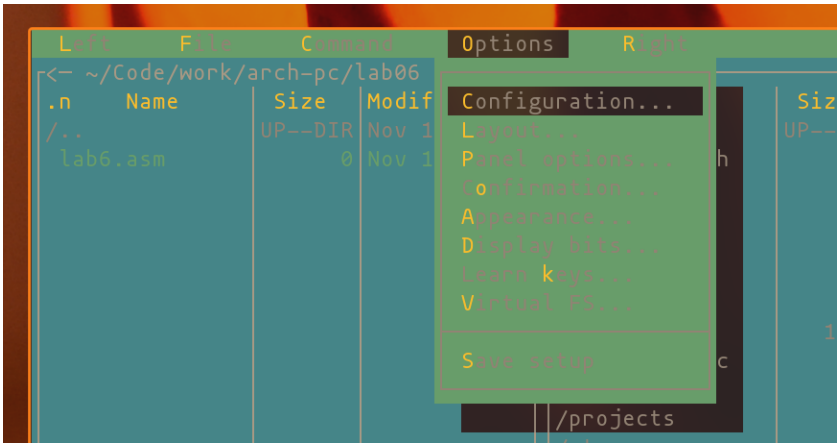

Рис. 2.4: Открытие настроек MC

Нажмем t что бы снять галочку с “Use internal edit”

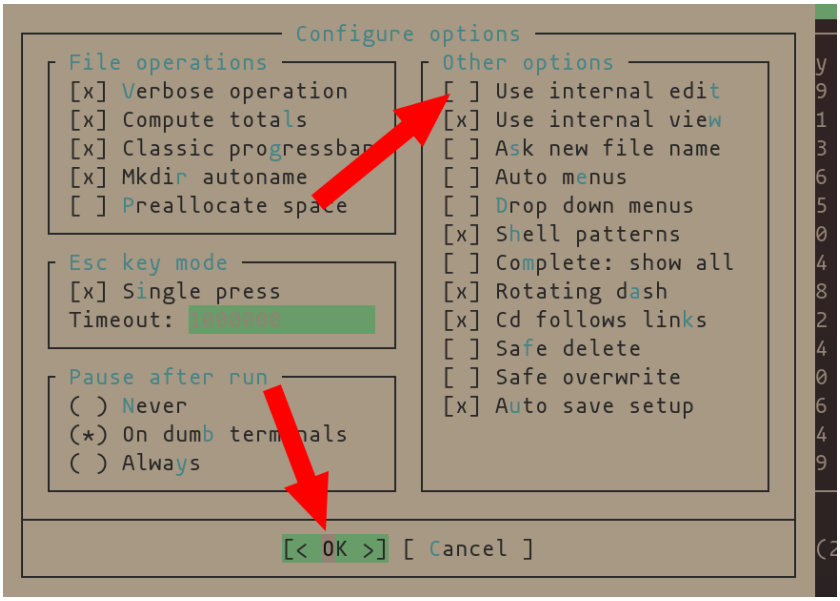

Рис. 2.5: Изменение конфигурации MC

Теперь, когда мы нажимаем F4 в MC, открывается текстовый редактор VIM

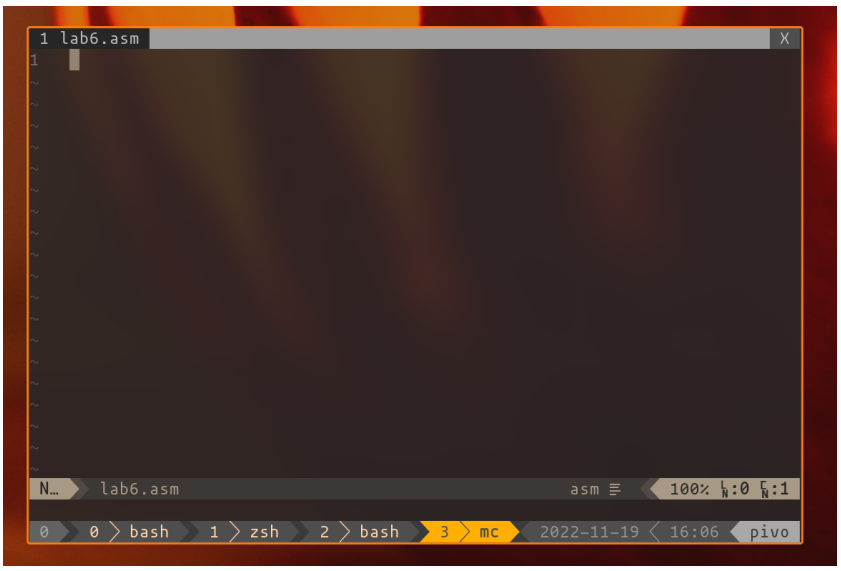

Рис. 2.6: Открытие текстового редактора через MC

2.3 Введем текст программы из листинга

Рис. 2.7: Введенный текст на языке ассемблера

Просмотрим содержимое измененного и сохраненного файла через MC, нажав F3

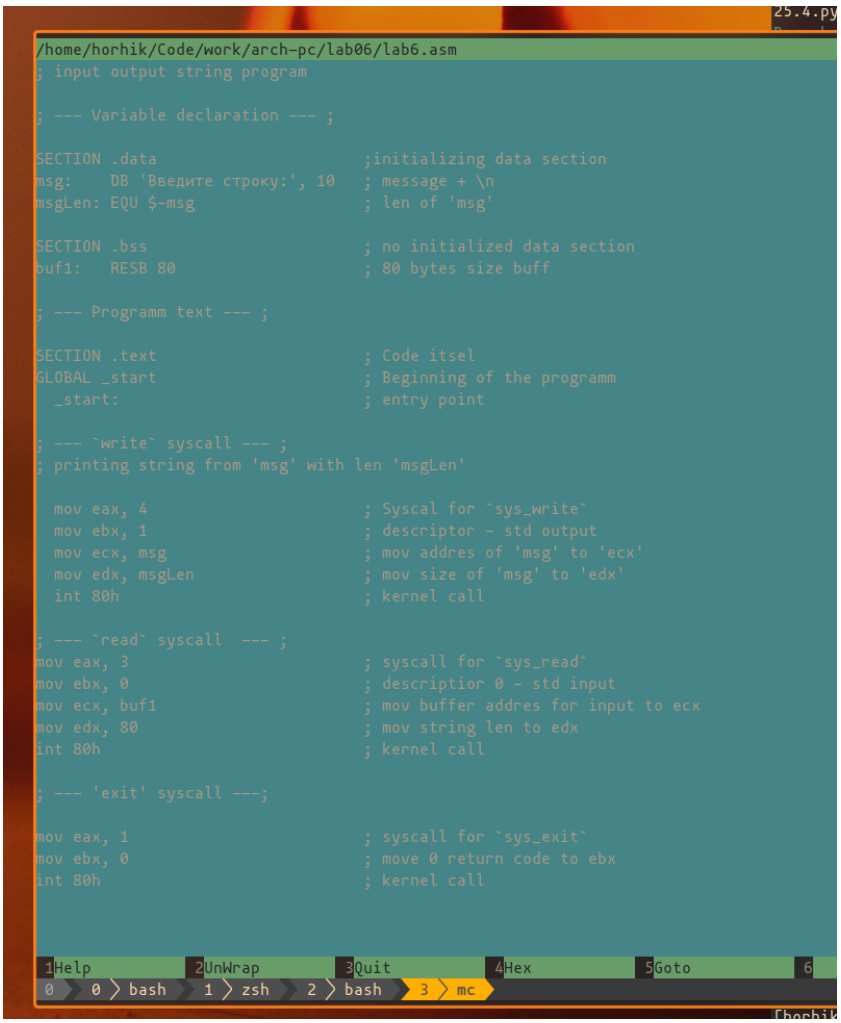

Рис. 2.8: Просмотор содержимого через MC

Оттранслируем, слинкуем файл `lab6-1.asm` (переместив lab6 в lab6-1). Запустим исполнимый файл.

```
[giosokin:lab06]$ mv lab6.asm  lab6-1.asm
[giosokin:lab06]$ nasm -f elf lab6-1.asm
[giosokin:lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[giosokin:lab06]$ ./lab6-1
Введите строку:
giosokin
[giosokin:lab06]$
```

Рис. 2.9: Трансляция линковка и запуск lab6-1.asm

2.4 Подключим внешний файл

Откроем MC

Рис. 2.10: Открытие MC

Найдем файл `in_out.asm` в `~/Downloads`

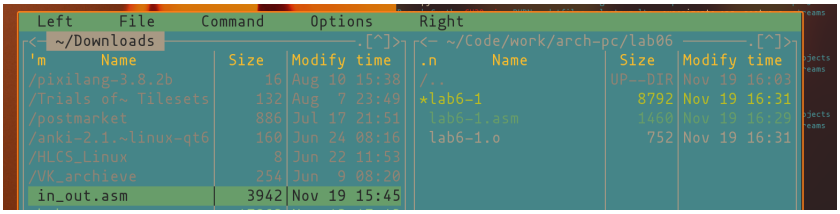

Рис. 2.11: Просмотр папки Downloads

Скопируем файл в lab5, нажав `F5`

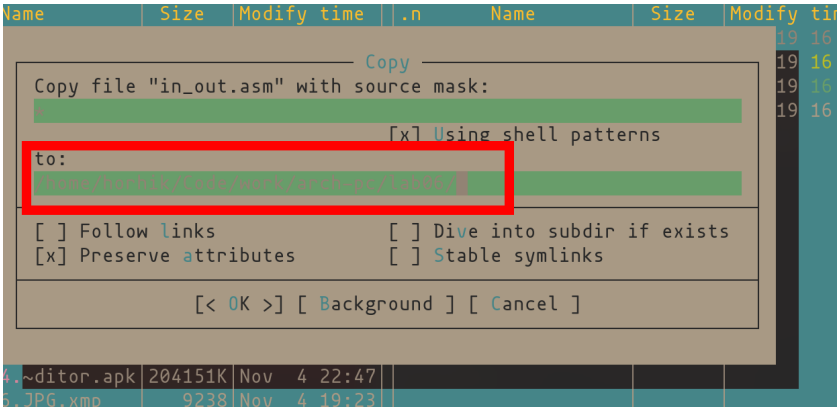

Рис. 2.12: Копирование через MC

Как видим, файл появился в директории `lab06`

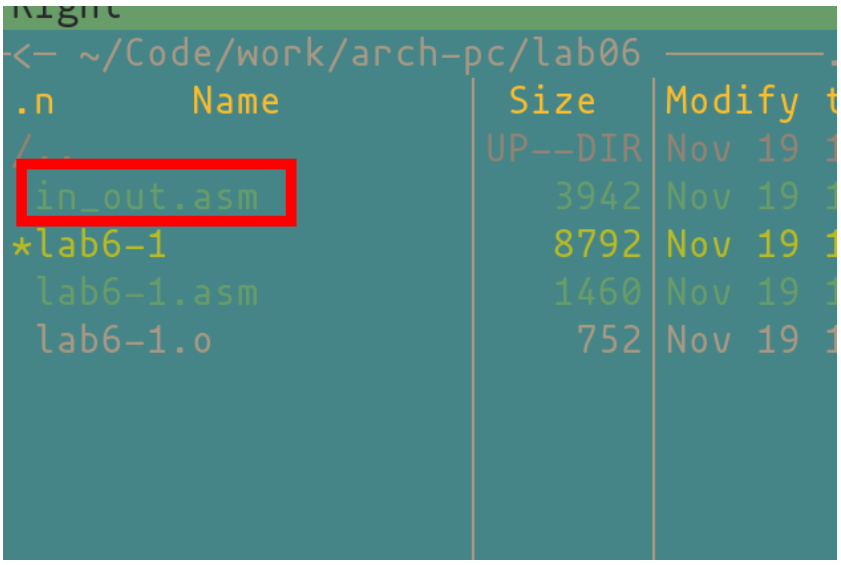

Рис. 2.13: Скопированный файл

Скопируем файл в `lab6-2.asm`

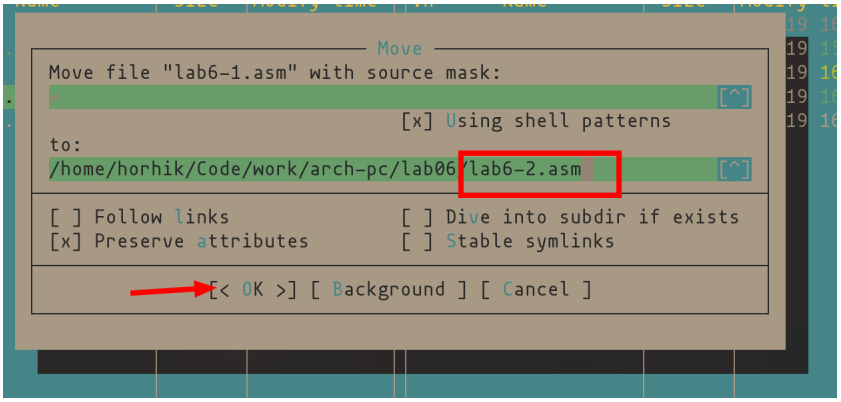

Рис. 2.14: Копирование в lab6-2.asm

Как видим, файл скопировался

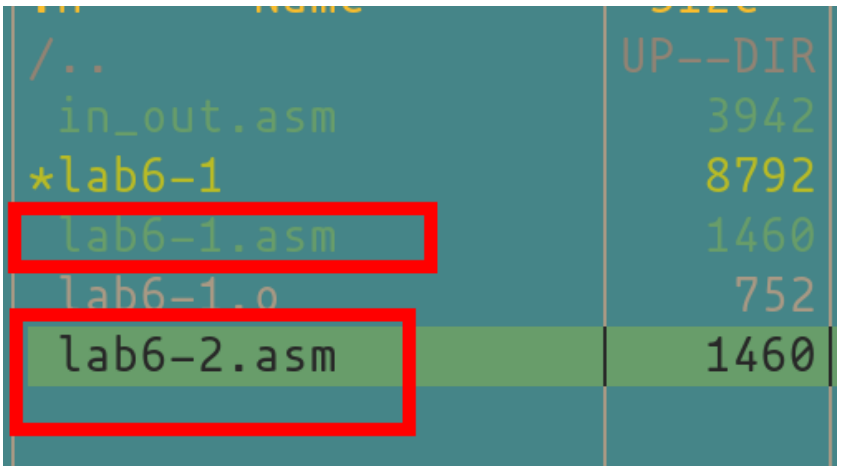

Рис. 2.15: Просмотр скопированного файла

2.5 Изменим содержимое lab6-2.asm, используя подпрограммы из `in_out.asm`

```
lab6-2.asm
; input output string program

%include 'in_out.asm'              ; including external file

; --- Variable declaration --- ;

SECTION .data                      ;initializing data section
msg:    DB 'Введите строку:', 10   ; message + \n

SECTION .bss                       ; no initialized data section
buf1:   RESB 80                    ; 80 bytes size buff

; --- Programm text --- ;

SECTION .text                      ; Code itsel
GLOBAL _start                      ; Beginning of the programm
  _start:                          ; entry point

; --- `write` syscall --- ;
; printing string from 'msg' with len 'msgLen'

  mov   eax, msg                     ; moving 'msg' addres to eax
  call  sprintLF                     ; call subrogramm for printing

  mov ecx, buf1                      ; moving 'buf1' addres to ecx
  mov edx, 80                        ; mov string len to edx
  call sread                         ; call subprogramm for output


  call quit                         ; call subprogram for quit
```

Рис. 2.16: Отредактированный lab6-2.asm

Создадим копию lab6-2.1.asm

Рис. 2.17: Новая копия

Заменим `sprintLF` на `sprint`

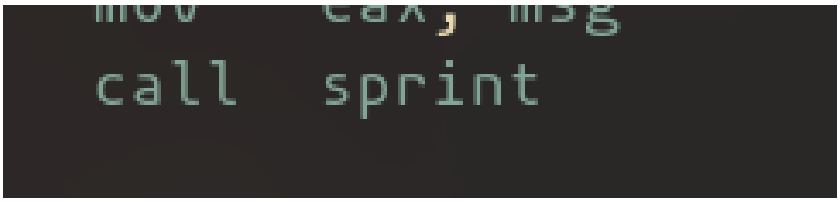

Рис. 2.18: Замена подпрограммы

Транслируем, слинкуем и запустим две эти программы

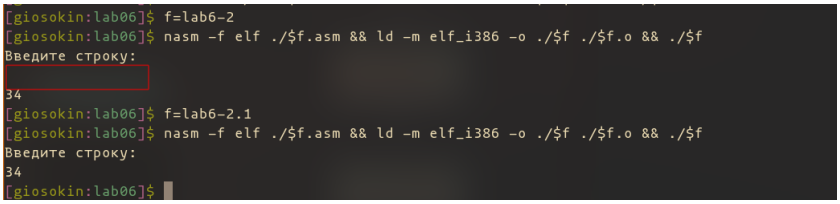

Рис. 2.19: Запуск двух программ

Как видим, у первой есть перенос строки, у второй он отсутствует

3 Задания для самостоятельной работы

3.1 Создадим копию `lab-1.asm` с выводом ввода

```
[giosokin:lab06]$ cp lab6-1.asm lab6-1-copy.asm
[giosokin:lab06]$ vim lab6-1-copy.asm
```

Рис. 3.1: Создание копии

Откроем новый файл и добавим в него блок с выводом строки из `buf1`

```
8 ; --- `write` syscall --- ;
7 ; printing inputed string from 'msg' with len 'msgLen'
6
5   mov eax, 4                      ; Syscal for `sys_write`
4   mov ebx, 1                      ; descriptor - std output
3   mov ecx, buf1
2   mov edx, 80                     ; mov size of 'msg' to 'edx'
1   int 80h                         ; kernel call
2
1
```

Рис. 3.2: Новый блок кода

Скомпилируем и запустим

```
[giosokin:lab06]$ nasm -f elf ./$f.asm && ld -m elf_i386 -o ./$f ./$f.o && ./$f
Введите строку:
Осокин Георгий
Осокин Георгий
[giosokin:lab06]$
```

Рис. 3.3: Запуск lab-1

Заметим, что если заменить `buf1` на `msg+msgLen` то резульат будет таким же. Так как `buf1` инициализируется в памяти сразу после `msg` и размер `msg` это `msgLen`. Поэтому, аддрес `msg+msgLen` в точности аддрес `buf1`

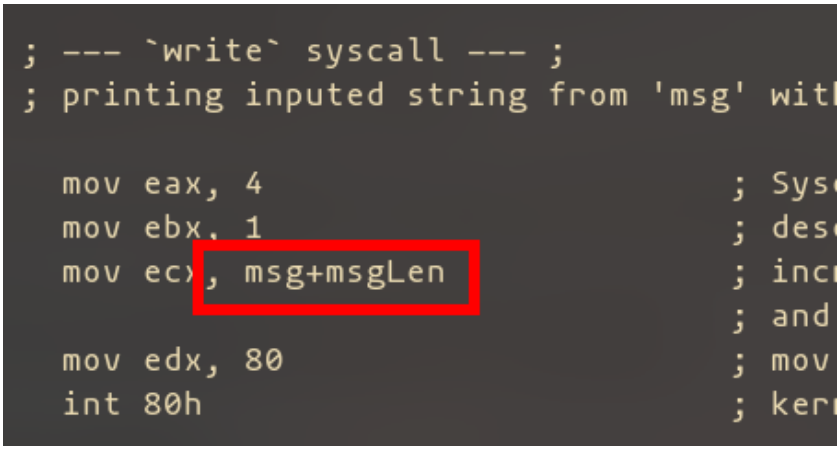

Рис. 3.4: Замена buf на msg+msgLen

Скомпилируя этот файл мы получим такой же результат

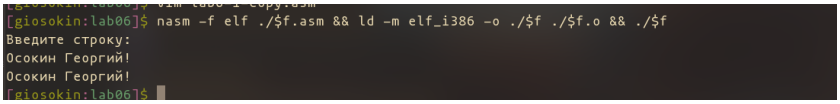

Рис. 3.5: Запуск программы

Отнимем 1 и посмотрим что получится

Рис. 3.6: Отнятие единицы

Исполним

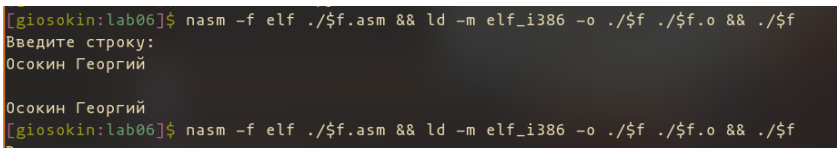

Рис. 3.7: Исполнение измененной программы

Мы захватили с собой перенос строки, которым заканчивался текст `msg`

3.2 Создадим копию `lab-2.asm` с выводом ввода

```
[giosokin:lab06]$ cp lab6-2.asm lab6-2-copy.asm
[giosokin:lab06]$ vim lab6-2-copy.asm
```

Рис. 3.8: Создание копии

Добавим две строчки, первая будет перемещать значение из буфера в eax Вторая исполнит подпрограмму вывода

```
6
5   mov   eax, buf1                    ; m
4   call  sprintLF                     ; c
3
2
```

Рис. 3.9: Добавление новых строчек

Скомпилируем и запустим

```
[giosokin:lab06]$ f=lab6-2-copy
[giosokin:lab06]$ nasm -f elf ./$f.asm && ld -m elf_i386 -o ./$f ./$f.o && ./$f
Введите строку:

Осокин Георгий
Осокин Георгий

[giosokin:lab06]$
```

Рис. 3.10: Запуск lab-2-copy

4 Выводы

В ходе данной лабораторной работы мы научились пользоваться Midnight Commander и освоили базовые инструкции языка ассемблера mov и int.

Список литературы