

Лабораторная работа 7

Сафин Андрей Алексеевич

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
5	Выводы	22

Список иллюстраций

4.1	man mc	8
4.2	mc	9
4.3	Копирование	10
4.4	Быстрый просмотр файла	11
4.5	Информация о файле	11
4.6	Дерево	12
4.7	Просмотр файла	13
4.8	Редактирование	13
4.9	Создание каталога 001	14
4.10	Копирование файла в каталог	14
4.11	Поиск файлов с условиями	15
4.12	Выполнение команды из истории	16
4.13	Возвращение в домашний каталог	16
4.14	Файл расширений	17
4.15	Изменение интерфейса mc	17
4.16	Текст, скопированный в text.txt	18
4.17	Выделение синтаксиса	19
4.18	Удаление строки	19
4.19	Копирование фрагмента	20
4.20	Перенос фрагмента на другую строку	20
4.21	Отмена изменений	21
4.22	Изменение файла в начале и в конце	21

Список таблиц

1 Цель работы

Освоение основных возможностей командной оболочки Midnight Commander. Приобретение навыков практической работы по просмотру каталогов и файлов; манипуляций с ними.

2 Задание

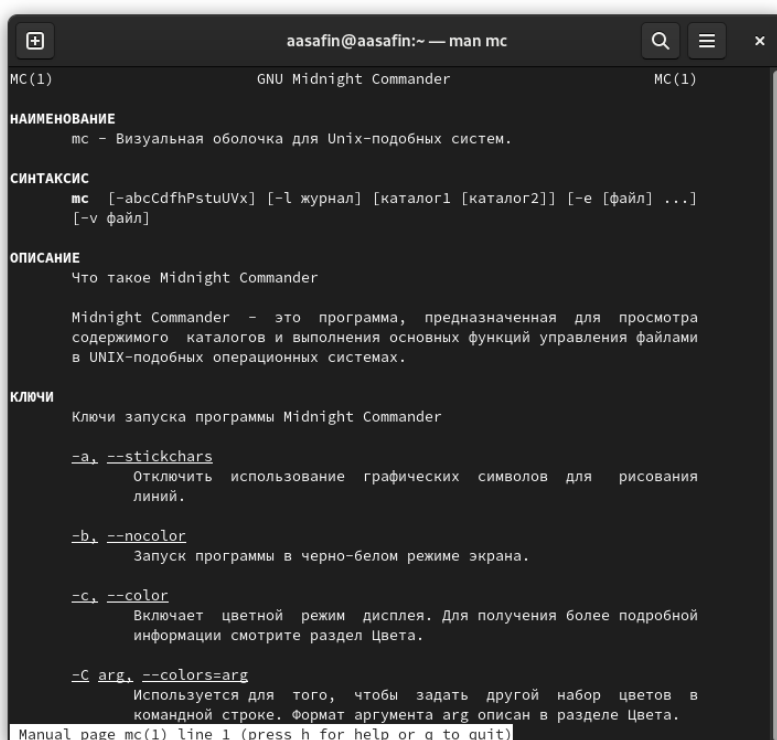
Выполнить описанные в работе действия с оболочкой тс и её встроенным текстовым редактором.

3 Теоретическое введение

Командная оболочка — интерфейс взаимодействия пользователя с операционной системой и программным обеспечением посредством команд. Midnight Commander (или mc) — псевдографическая командная оболочка для UNIX/Linux систем.

4 Выполнение лабораторной работы

Вызвана справка по mc (рис. 4.1). Открыт mc (рис. 4.2). Осуществлены несколько операций, в их числе копирование (рис. 4.3). Выполнены основные команды меню левой панели (рис. 4.4-4.6).



```
aasafin@aasafin:~ — man mc
MC(1)                                GNU Midnight Commander                                MC(1)

НАИМЕНОВАНИЕ
mc - Визуальная оболочка для Unix-подобных систем.

СИНТАКСИС
mc [-abcCdfhPstuUVx] [-l журнал] [каталог1 [каталог2]] [-e [файл] ...]
[-v файл]

ОПИСАНИЕ
Что такое Midnight Commander

Midnight Commander - это программа, предназначенная для просмотра
содержимого каталогов и выполнения основных функций управления файлами
в UNIX-подобных операционных системах.

КЛЮЧИ
Ключи запуска программы Midnight Commander

-a, --stickchars
Отключить использование графических символов для рисования
линий.

-b, --nocolor
Запуск программы в черно-белом режиме экрана.

-c, --color
Включает цветной режим дисплея. Для получения более подробной
информации смотрите раздел Цвета.

-C arg, --colors=arg
Используется для того, чтобы задать другой набор цветов в
командной строке. Формат аргумента arg описан в разделе Цвета.

Manual page mc(1) line 1 (press h for help or q to quit)
```

Рис. 4.1: man mc

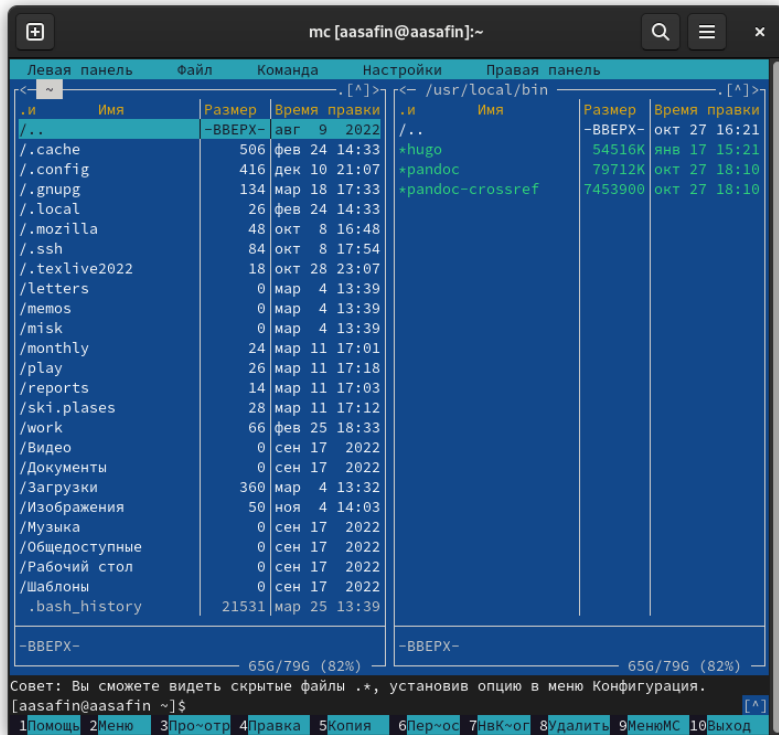


Рис. 4.2: mc

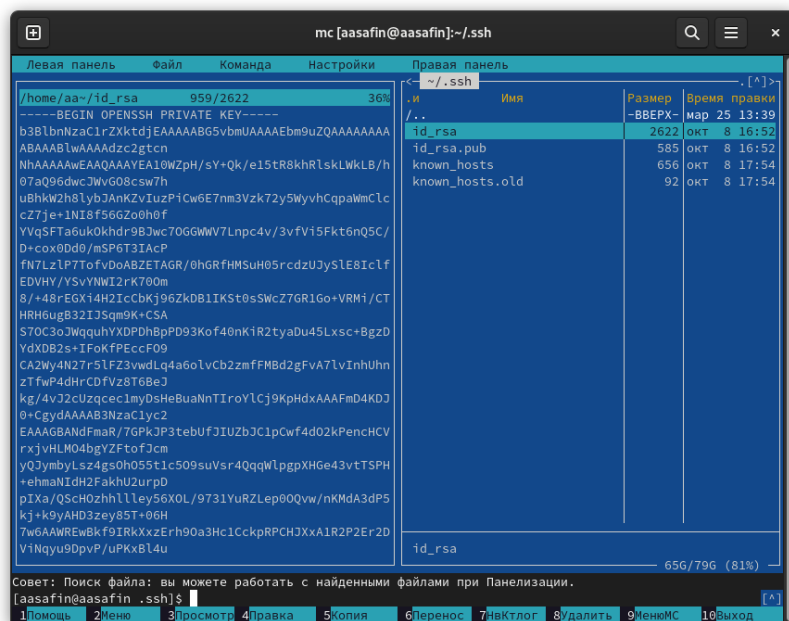


Рис. 4.4: Быстрый просмотр файла

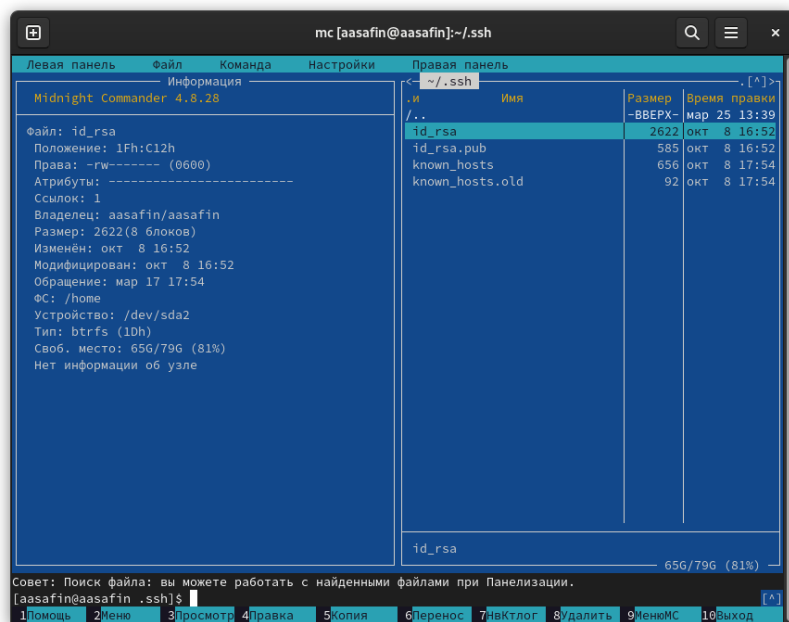


Рис. 4.5: Информация о файле

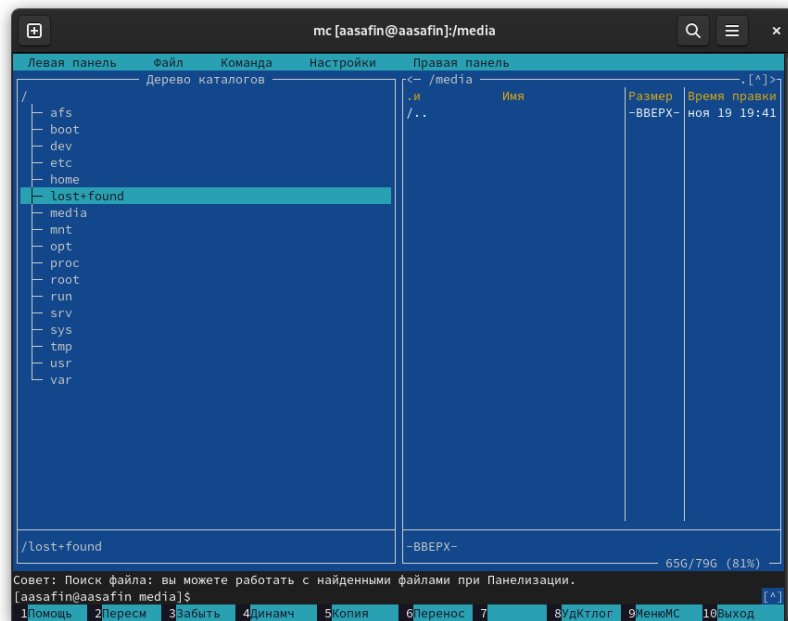


Рис. 4.6: Дерево

С помощью подменю файл: просмотрено и отредактировано содержимое hello.asm (рис. 4.7, 4.8); создан каталог 001 (рис. 4.9); скопирован lab5 в 001 (рис. 4.10).

```
mc [aasafin@aasafin]:~/work/arch-pc/lab05
/home/aasafin/work/arch-pc/lab05/hello.asm 812/812 100%
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
1Помощь 2Разверн 3Выход 4Нех 5Перейти 6 7Поиск 8Исходный 9Формат 10Выход
```

Рис. 4.7: Просмотр файла

```
mc [aasafin@aasafin]:~/work/arch-pc/lab05
hello.asm [-M--] 0 L: [ 1+ 2 3/ 17] *(67 / 759b) 0010 0x00A [*] [X]
; hello.asm
SECTION .data ; Начало секции данных
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перейти 7Поиск 8Удалить 9МенюМС 10Выход
```

Рис. 4.8: Редактирование

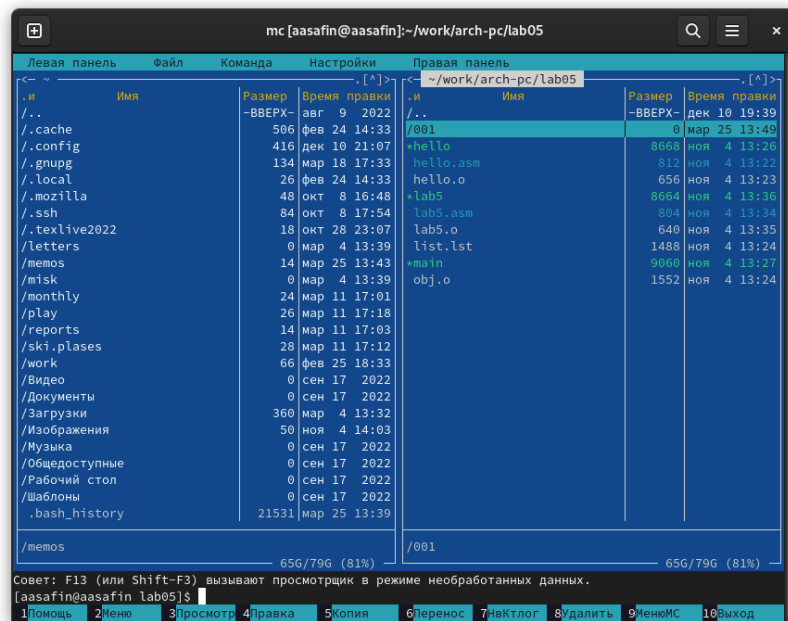


Рис. 4.9: Создание каталога 001

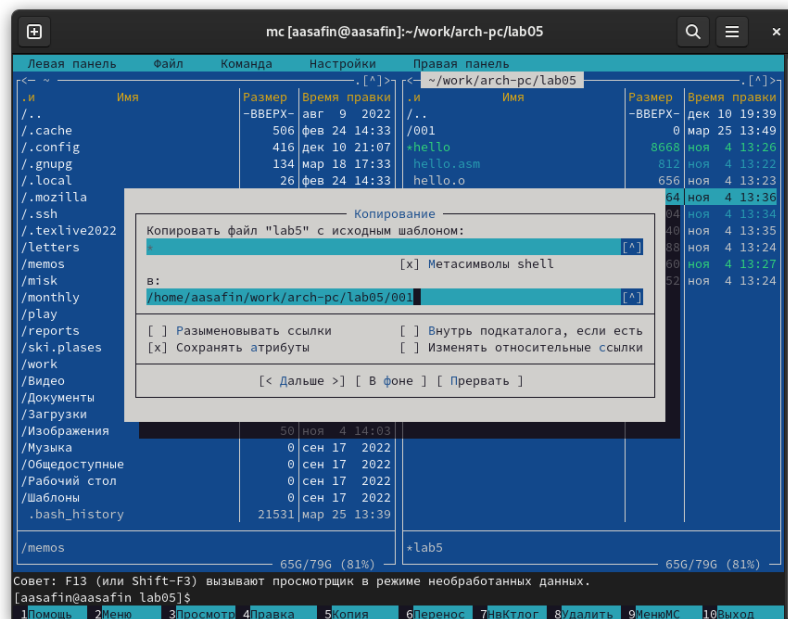


Рис. 4.10: Копирование файла в каталог

С помощью подменю команда выполнены: поиск в файловой системе с усло-

виями (рис. 4.11); повторение одной из предыдущих команд (рис. 4.12); переход в домашний каталог (рис. 4.13); просмотр файла расширений (рис. 4.14).

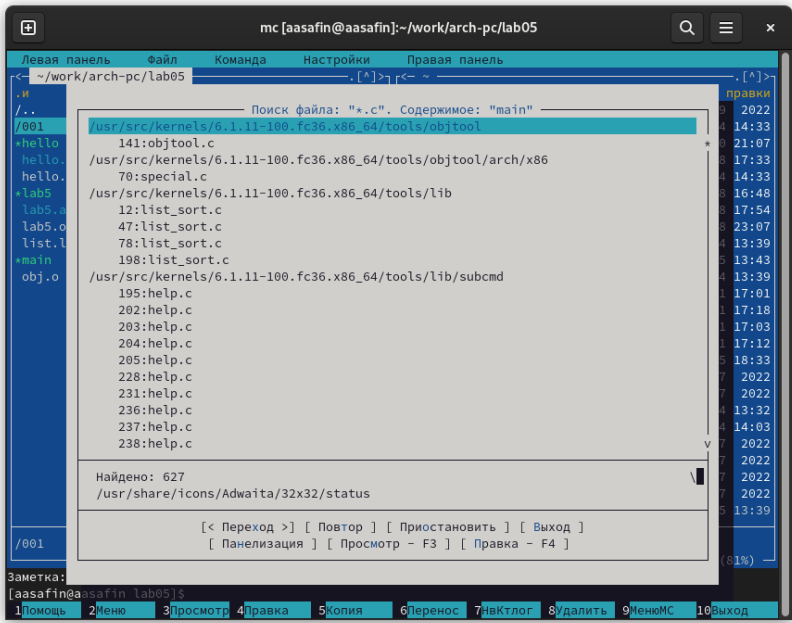


Рис. 4.11: Поиск файлов с условиями

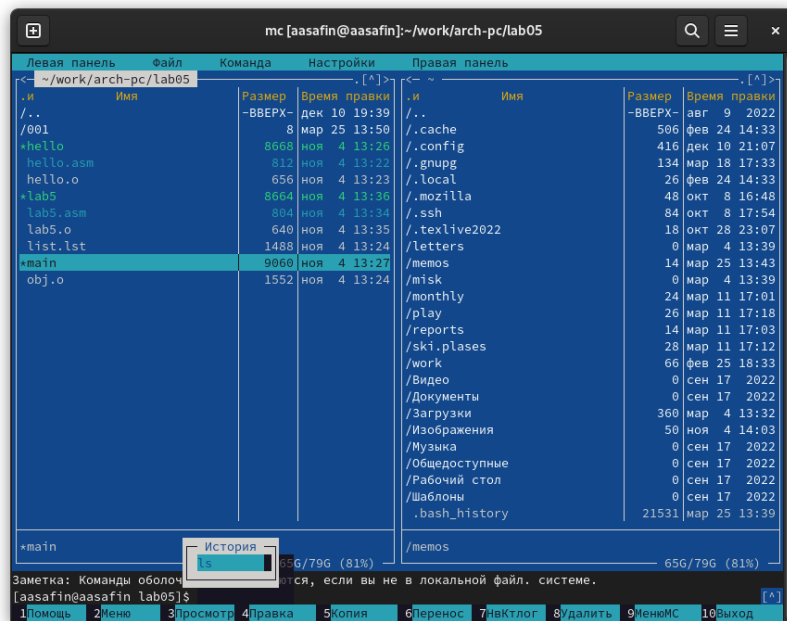


Рис. 4.12: Выполнение команды из истории

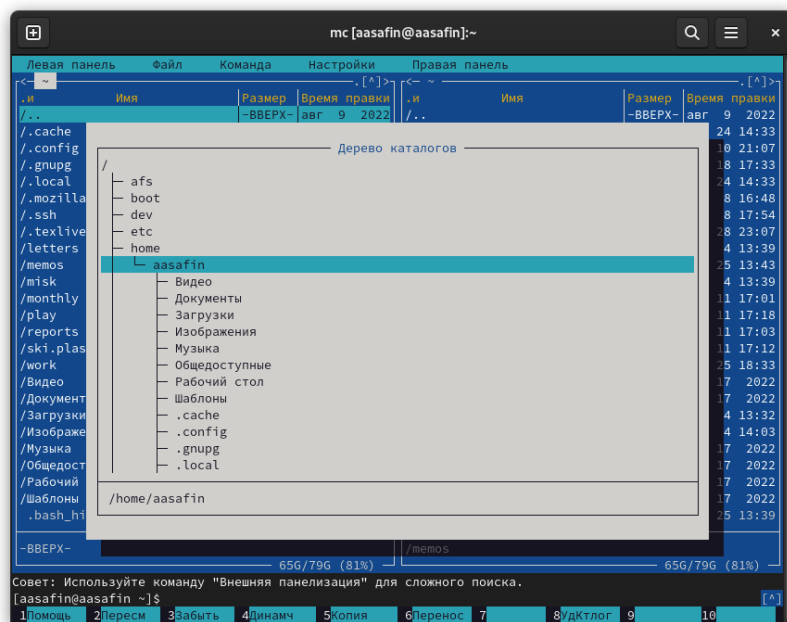


Рис. 4.13: Возвращение в домашний каталог

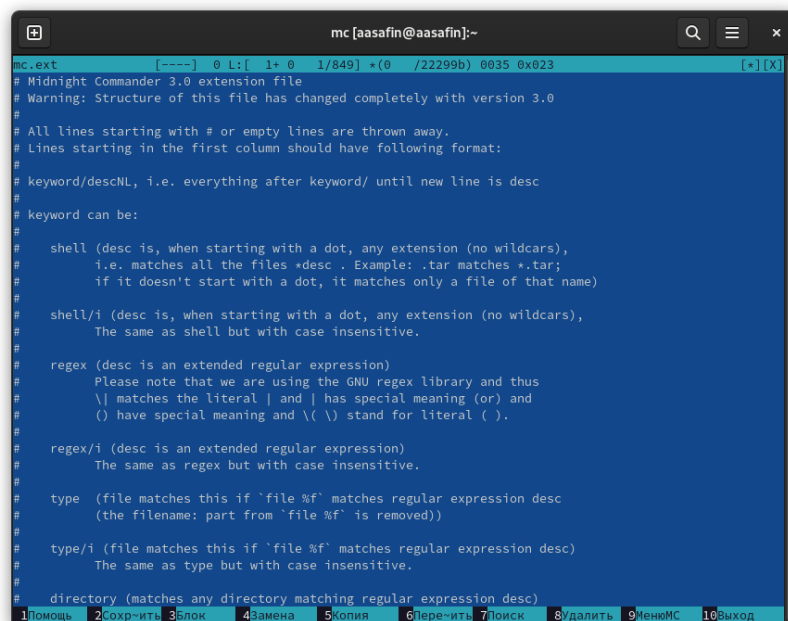


Рис. 4.14: Файл расширений

С помощью подменю настройки изменен интерфейс (рис. 4.15).

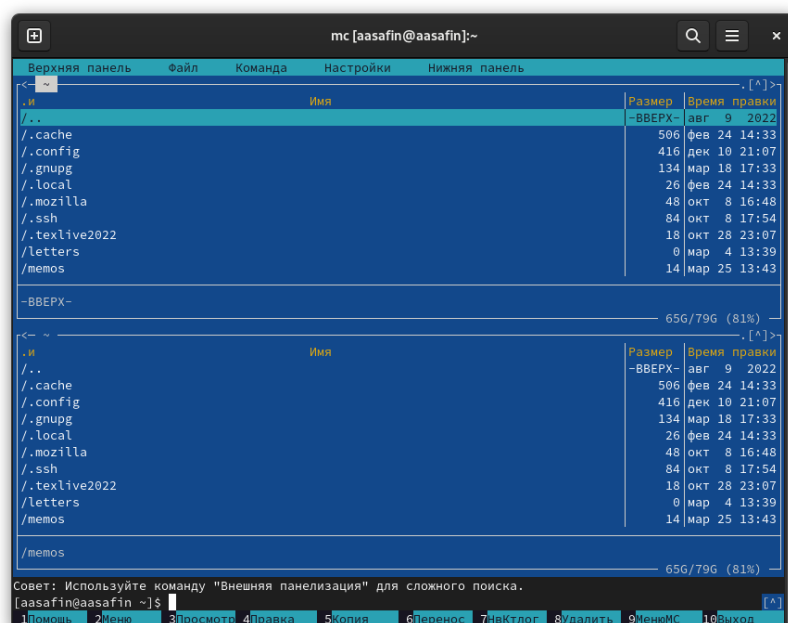
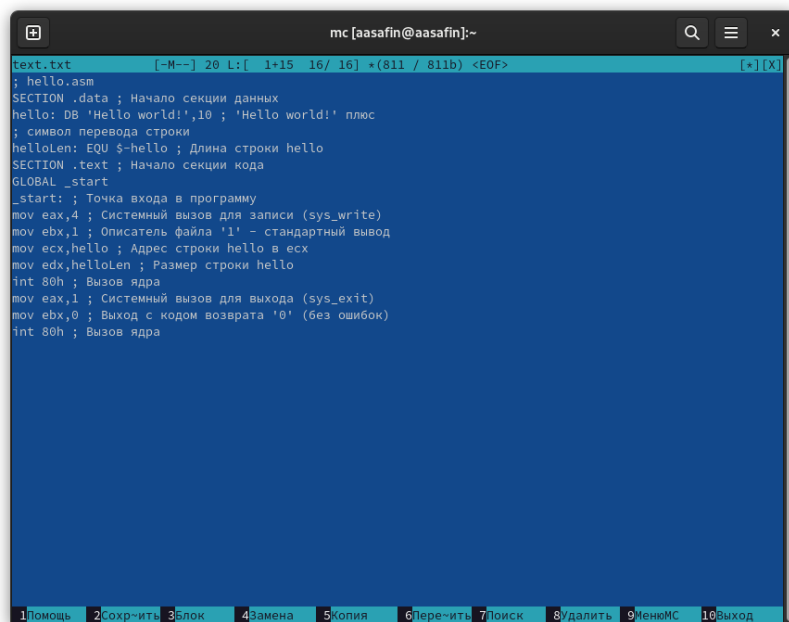


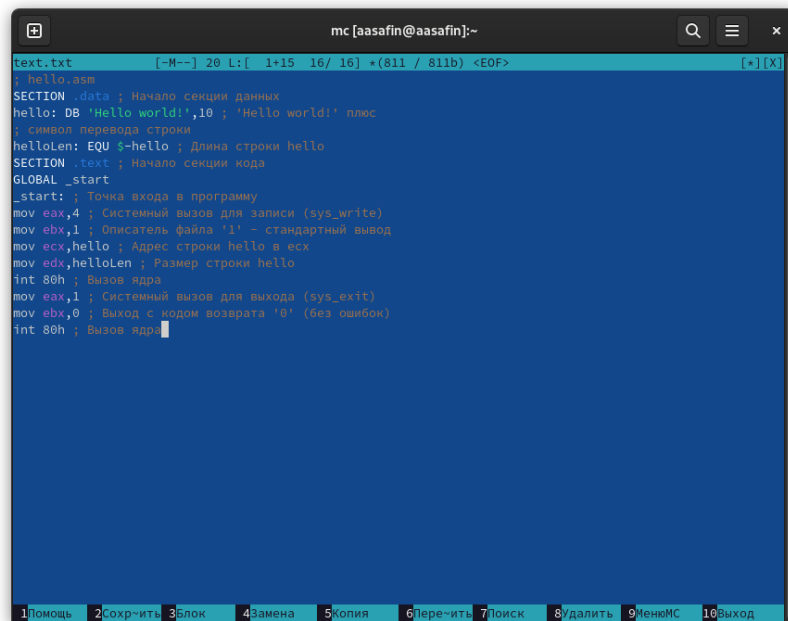
Рис. 4.15: Изменение интерфейса mc

Создан файл text.txt, в него скопирован текст hello.asm (рис. 4.16). Установлено выделение синтаксиса (рис. 4.17). Удалена строка (рис. 4.18). Скопирован фрагмент (рис. 4.19). Фрагмент перенесен (рис. 4.20). Изменение отменено (рис. 4.21). Файл изменен в конце и в начале (рис. 4.22).



```
text.txt [-M--] 20 L: [ 1+15 16/ 16] *(811 / 811b) <EOF> [*][X]
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
```

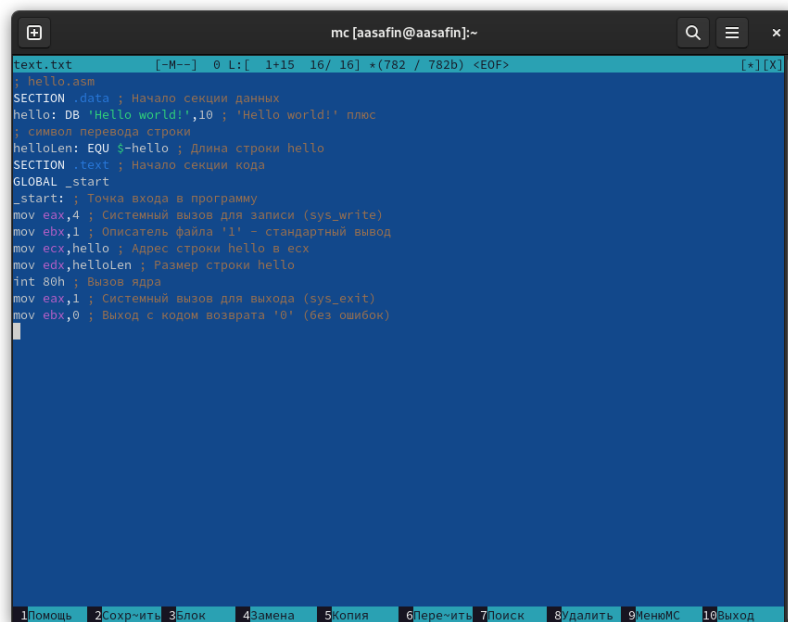
Рис. 4.16: Текст, скопированный в text.txt



The screenshot shows a text editor window titled 'mc [aasafin@aasafin]:~'. The file 'text.txt' is open, displaying assembly code for a 'hello.asm' program. The code includes sections for data and text, and a global start section. The text section contains instructions for writing to stdout and exiting. The code is color-coded: comments are grey, directives like SECTION and GLOBAL are blue, and instructions like mov and int are green. The status bar at the bottom shows a menu with options: 1Помощь, 2Сохранить, 3Блок, 4Замена, 5Копия, 6Пере-ить, 7Поиск, 8Удалить, 9Исменить, 10Выход.

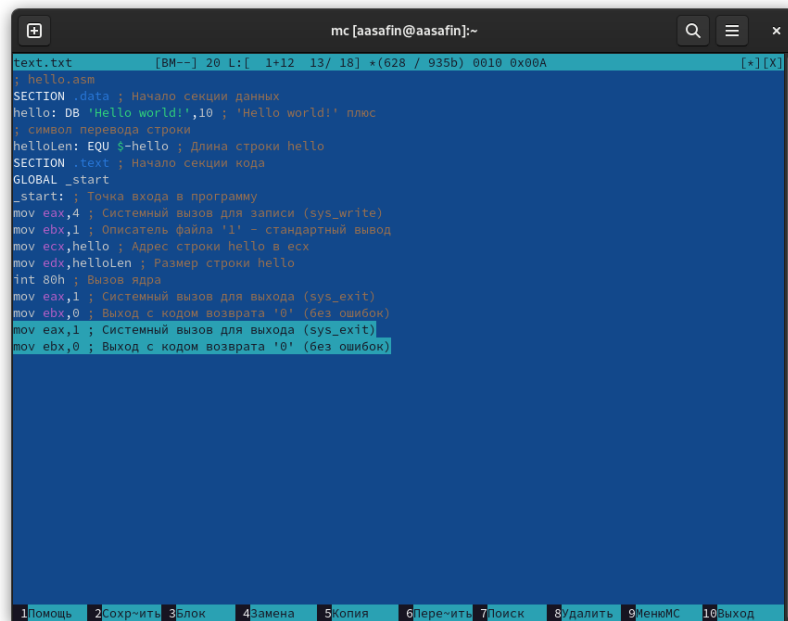
```
text.txt [-M--] 20 L: [ 1+15 16/ 16] *(811 / 811b) <EOF>
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
int 80h ; Вызов ядра
```

Рис. 4.17: Выделение синтаксиса



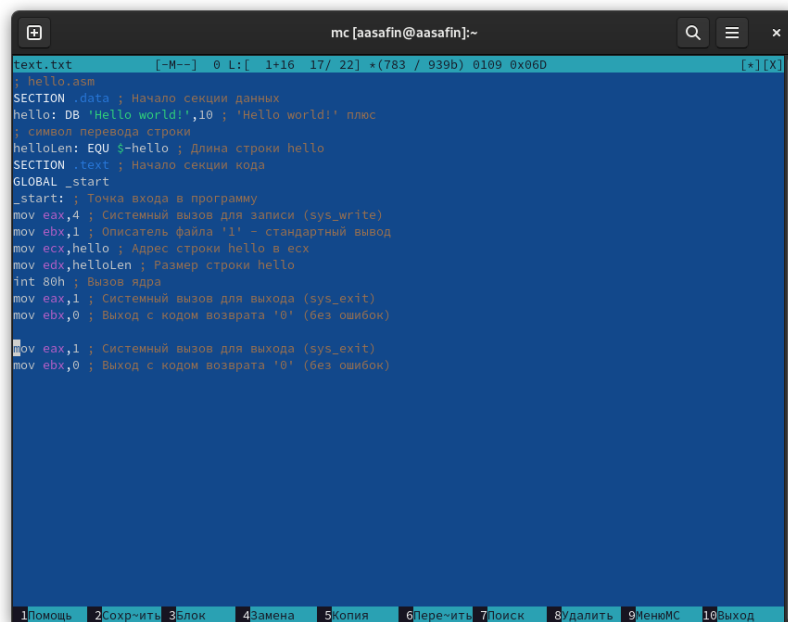
This screenshot is identical to the one above, showing the same assembly code in the same text editor with syntax highlighting. The status bar at the bottom shows the same menu options: 1Помощь, 2Сохранить, 3Блок, 4Замена, 5Копия, 6Пере-ить, 7Поиск, 8Удалить, 9Исменить, 10Выход.

Рис. 4.18: Удаление строки



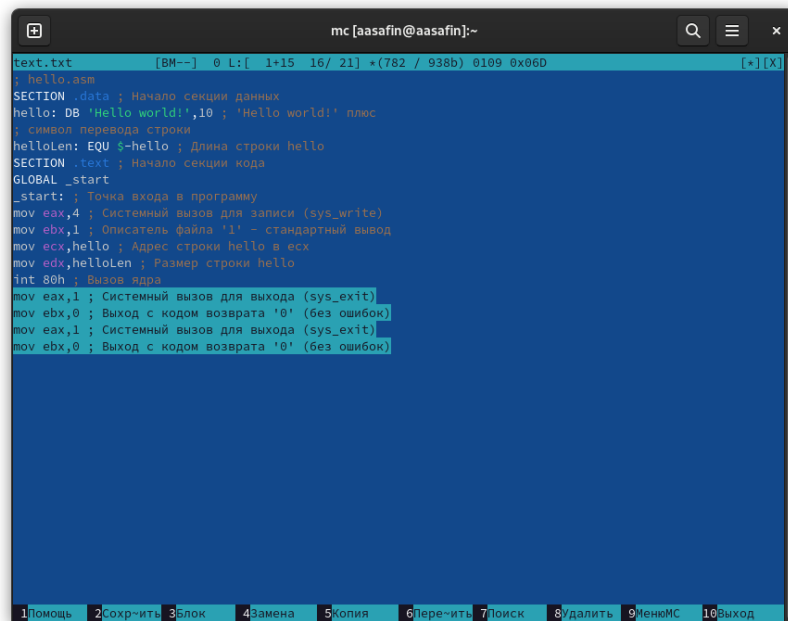
```
text.txt [BM--] 20 L: [ 1+12 13/ 18] *(628 / 935b) 0010 0x00A [*] [X]
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
```

Рис. 4.19: Копирование фрагмента



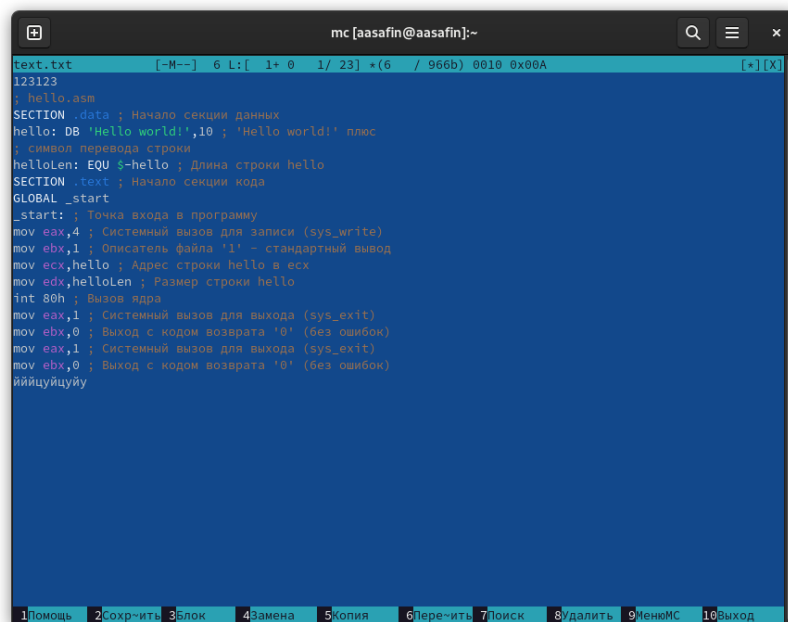
```
text.txt [-M--] 0 L: [ 1+16 17/ 22] *(783 / 939b) 0109 0x06D [*] [X]
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
```

Рис. 4.20: Перенос фрагмента на другую строку



```
text.txt [BM--] 0 L: [ 1+15 16/ 21] *(782 / 938b) 0109 0x06D [*][X]
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
```

Рис. 4.21: Отмена изменений



```
text.txt [-M--] 6 L: [ 1+ 0 1/ 23] *(6 / 966b) 0010 0x00A [*][X]
123123
; hello.asm
SECTION .data ; Начало секции данных
hello: DB 'Hello world!',10 ; 'Hello world!' плюс
; символ перевода строки
helloLen: EQU $-hello ; Длина строки hello
SECTION .text ; Начало секции кода
GLOBAL _start
_start: ; Точка входа в программу
mov eax,4 ; Системный вызов для записи (sys_write)
mov ebx,1 ; Описатель файла '1' - стандартный вывод
mov ecx,hello ; Адрес строки hello в ecx
mov edx,helloLen ; Размер строки hello
int 80h ; Вызов ядра
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата '0' (без ошибок)
иййцйцйу
```

Рис. 4.22: Изменение файла в начале и в конце

5 Выводы

Описанные действия с тс произведены, навык работы получен.