

# **Отчёт по лабораторной работе №9**

**Командная оболочка Midnight Commander**

Маныев Ресулбег

# Содержание

1	Цель работы	4
2	Выполнение лабораторной работы	5
3	Вывод	22
4	Контрольные вопросы	23

# List of Figures

2.1	Запуск ms . . . . .	5
2.2	Выделение . . . . .	6
2.3	Отмена . . . . .	6
2.4	Копирование . . . . .	7
2.5	Перемещение . . . . .	7
2.6	Информация . . . . .	7
2.7	Быстрый просмотр . . . . .	8
2.8	Информация . . . . .	8
2.9	Дерево каталогов . . . . .	9
2.10	Просмотр содержимого текстового файла . . . . .	9
2.11	Отредактируем содержимое текстового файла без сохранения ре- зультатов . . . . .	10
2.12	Создание каталога . . . . .	10
2.13	Копирование в файлов в созданный каталог . . . . .	10
2.14	Поиск файлов . . . . .	11
2.15	История команд . . . . .	11
2.16	Переход в домашний каталог . . . . .	11
2.17	Просмотр файла расширений . . . . .	12
2.18	Просмотр файла меню . . . . .	12
2.19	Конфигурация . . . . .	13
2.20	Внешний вид . . . . .	13
2.21	Настройки панелей . . . . .	14
2.22	Подтверждение . . . . .	14
2.23	Оформление . . . . .	14
2.24	Кодировка символов . . . . .	15
2.25	Распознавание клавиш . . . . .	15
2.26	Файл с текстом . . . . .	16
2.27	Файл с текстом . . . . .	16
2.28	Копирование фрагмента . . . . .	17
2.29	Сохранение . . . . .	17
2.30	Отмена . . . . .	18
2.31	Переход в конец файла . . . . .	19
2.32	Переход в начало файла . . . . .	19
2.33	Файл с программой . . . . .	20
2.34	Цветовыделение синтаксиса . . . . .	21

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

1 Изучим информацию о `mc` при помощи справки `man`. Воспользуемся справкой и узнаем что для того чтобы войти в командную оболочку мы должны ввести в командной строке `mc`.

2 Запустим из командной строки `mc`.

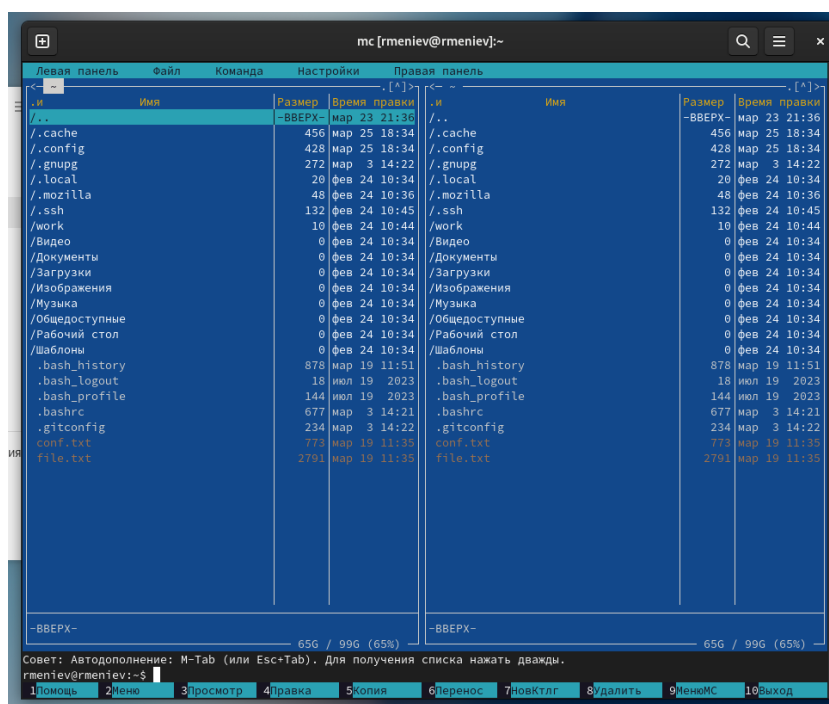


Figure 2.1: Запуск mc

3 Выполните несколько операций в `mc`, используя управляющие клавиши

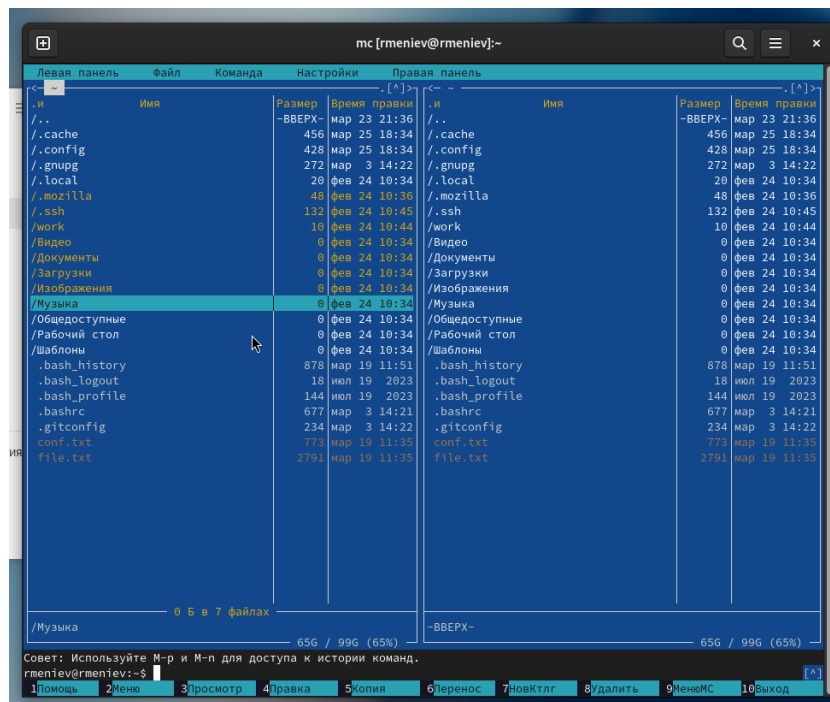


Figure 2.2: Выделение

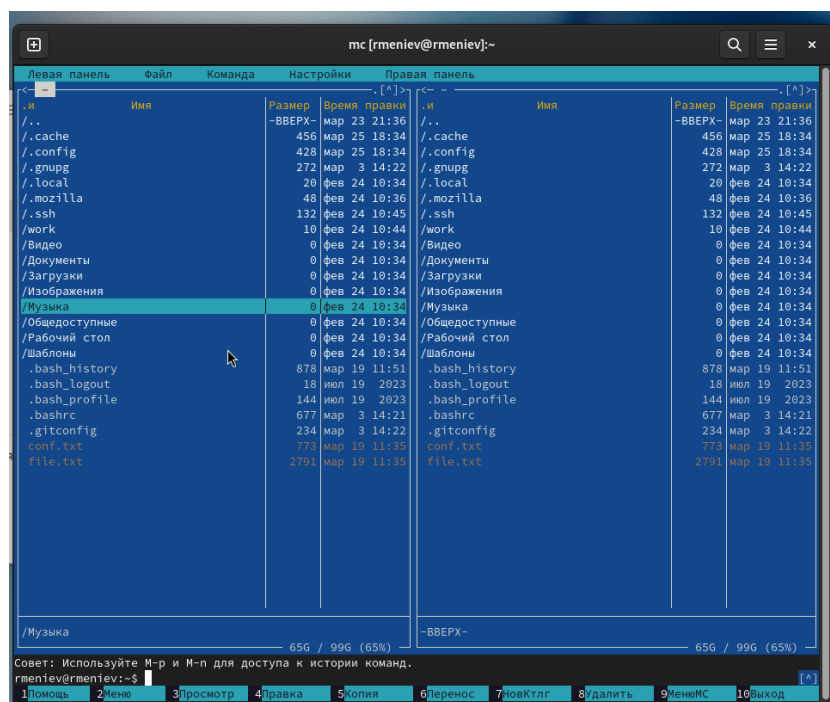


Figure 2.3: Отмена

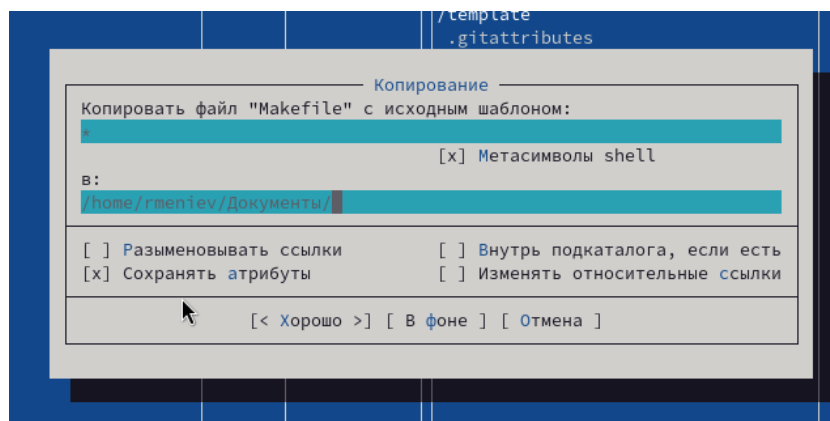


Figure 2.4: Копирование

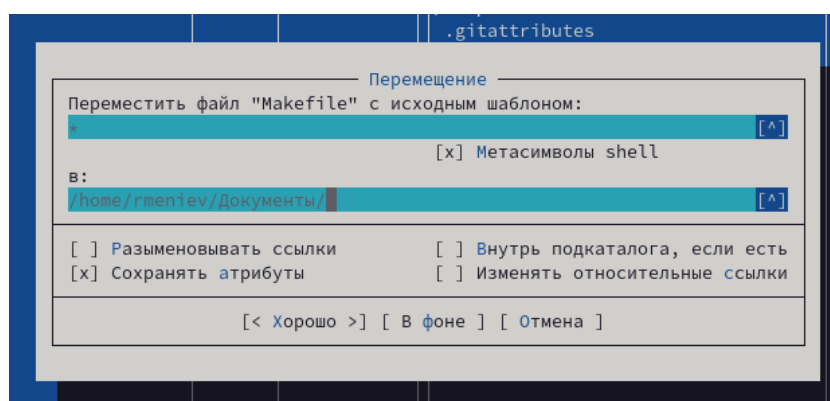


Figure 2.5: Перемещение

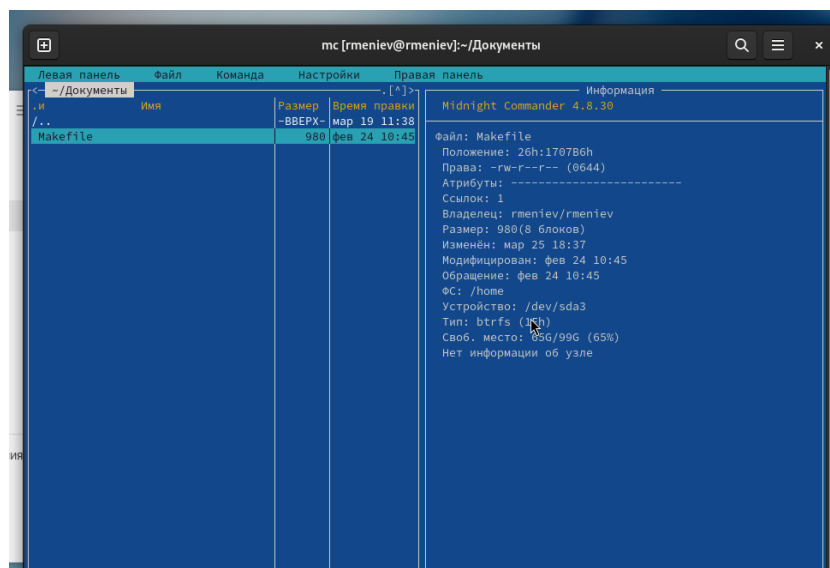


Figure 2.6: Информация

4 Выполните основные команды меню левой панели.

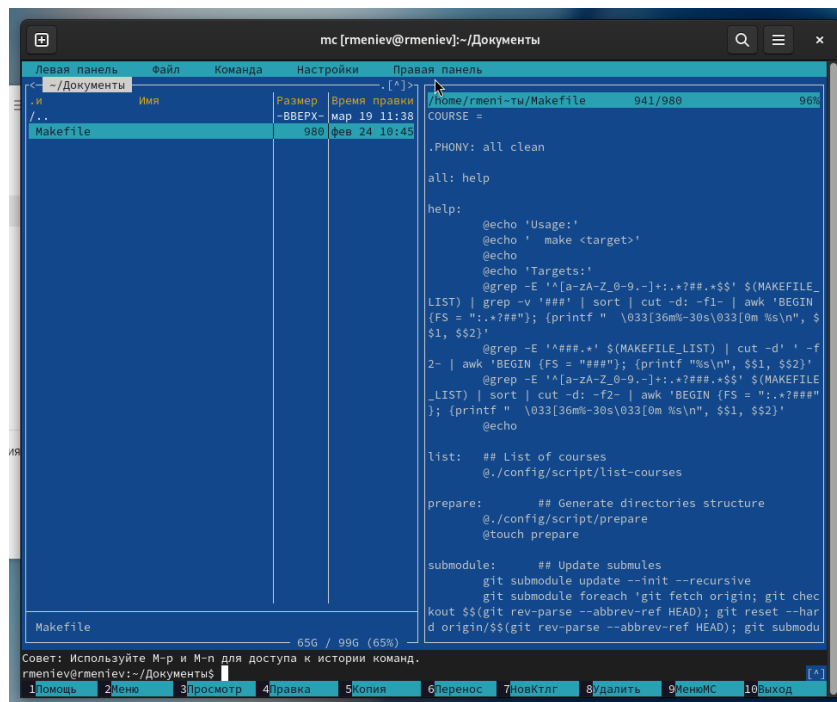


Figure 2.7: Быстрый просмотр

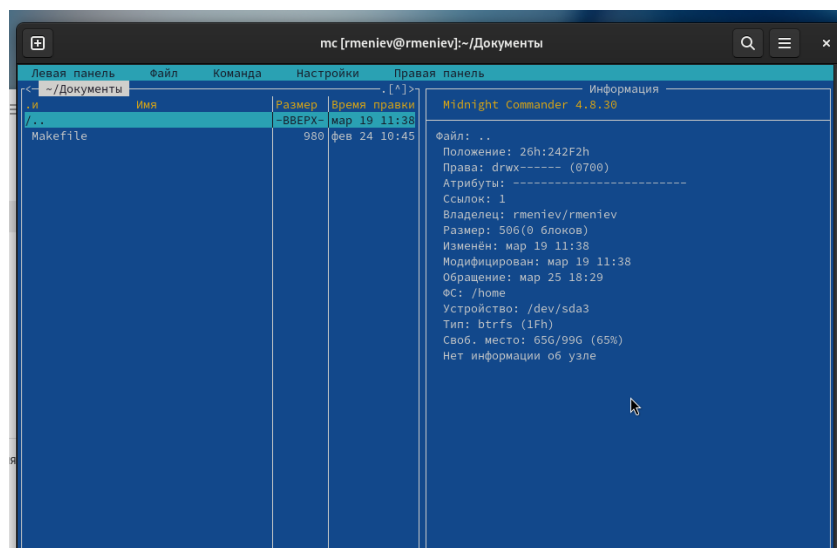


Figure 2.8: Информация



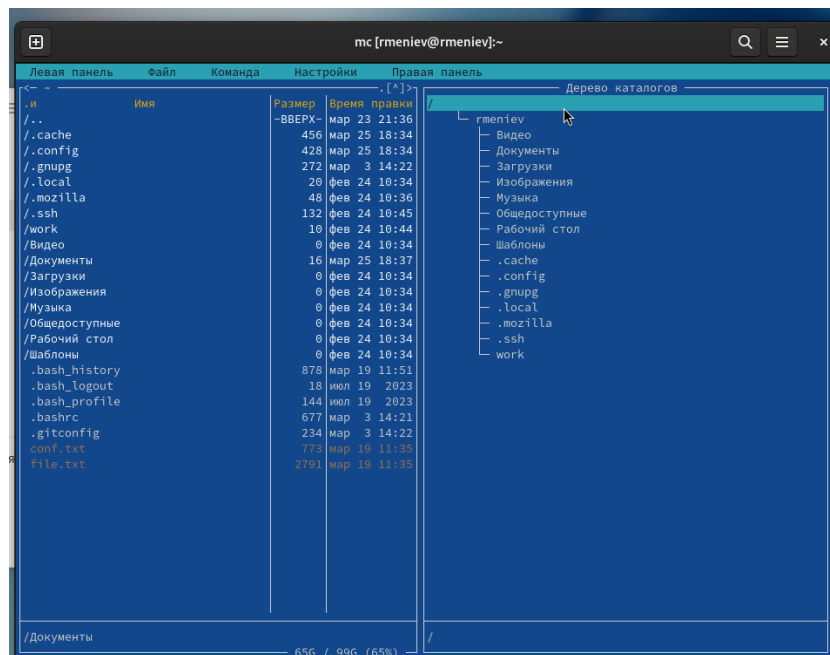


Figure 2.9: Дерево каталогов

5 Используя возможности подменю Файл , выполним:

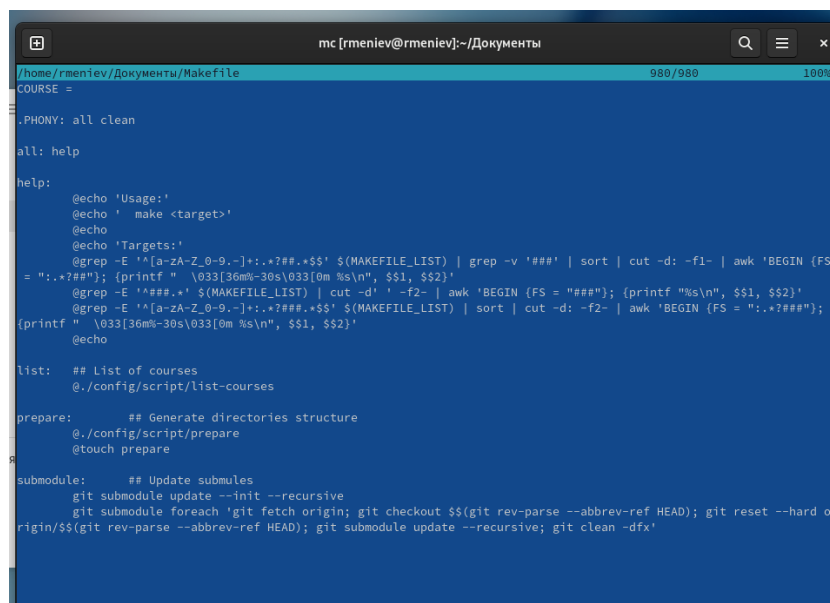
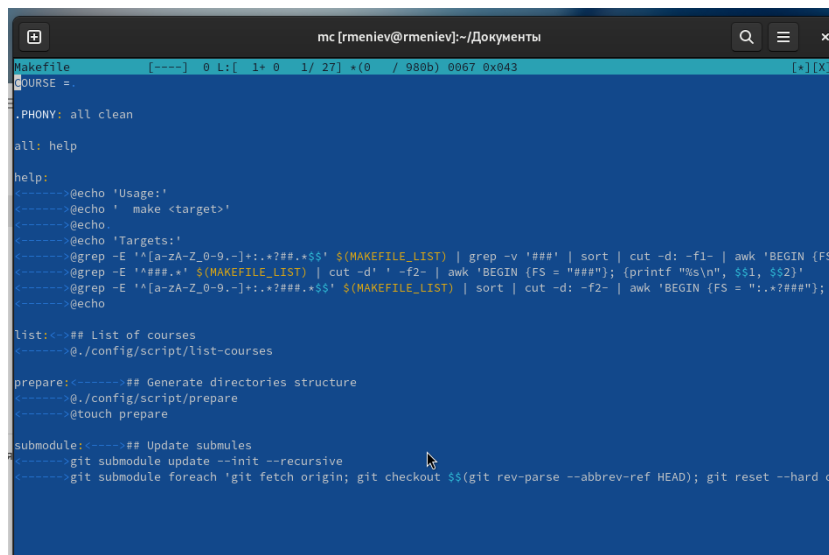


Figure 2.10: Просмотр содержимого текстового файла



```
mc [rmeniev@rmeniev]~/Документы
Makefile  [----]  0 L:[ 1+ 0 1/ 27] *(0 / 980b) 0067 0x043  [X]
@COURSE = .
.PHONY: all clean
all: help
help:
<----->@echo 'Usage:'
<----->@echo ' make <target>'
<----->@echo .
<----->@echo 'Targets:'
<----->@grep -E '[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | grep -v '###' | sort | cut -d: -f1- | awk 'BEGIN {FS
<----->@grep -E '[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | cut -d: -f2- | awk 'BEGIN {FS = "###"; {printf "%s\n", $1, $2}'
<----->@grep -E '[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | sort | cut -d: -f2- | awk 'BEGIN {FS = "###"; {printf "%s\n", $1, $2}'
<----->@echo
list:<-->## List of courses
<----->@./config/script/list-courses
prepare:<----->## Generate directories structure
<----->@./config/script/prepare
<----->@touch prepare
submodule:<----->## Update submodules
<----->git submodule update --init --recursive
<----->git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
```

Figure 2.11: Отредактируем содержимое текстового файла без сохранения результатов

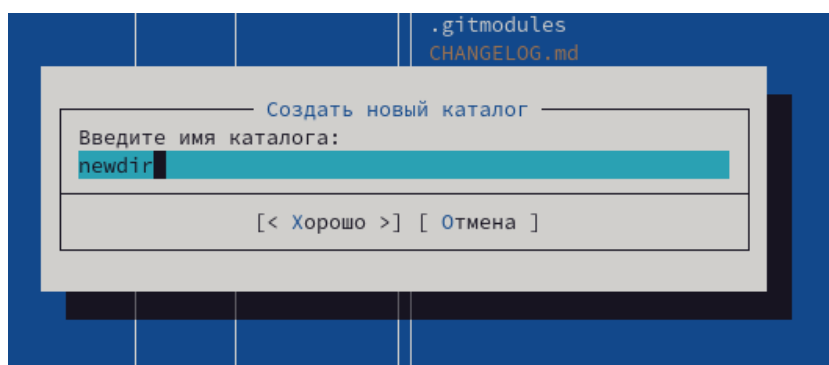


Figure 2.12: Создание каталога

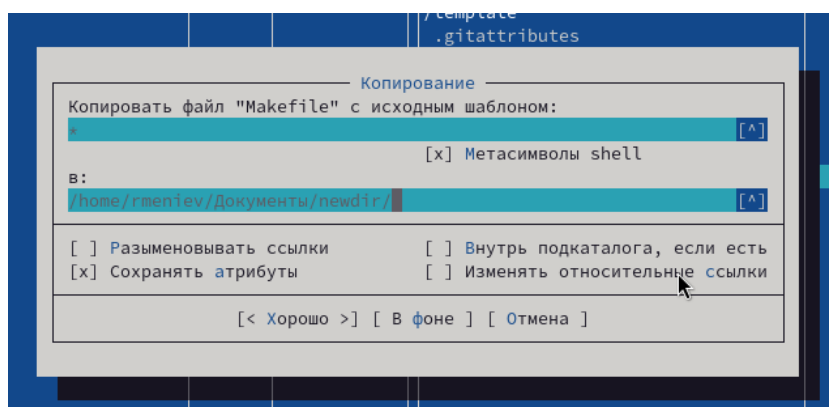


Figure 2.13: Копирование в файлов в созданный каталог

6. С помощью соответствующих средств подменю Команда осуществите:

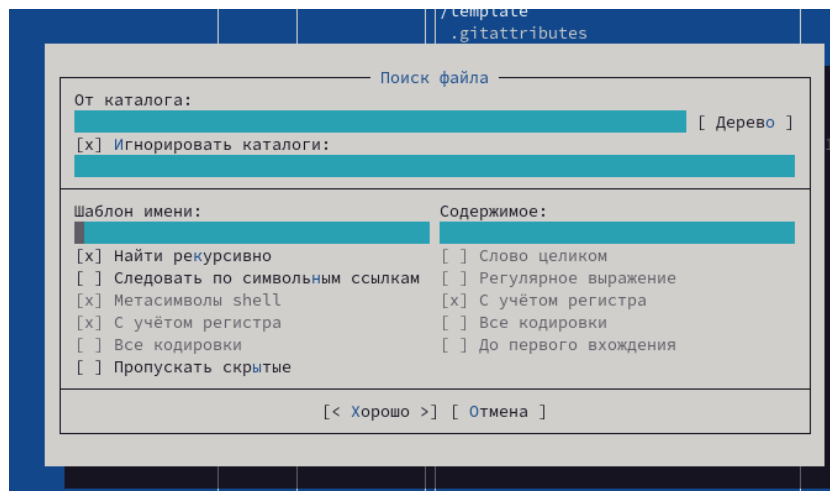


Figure 2.14: Поиск файлов

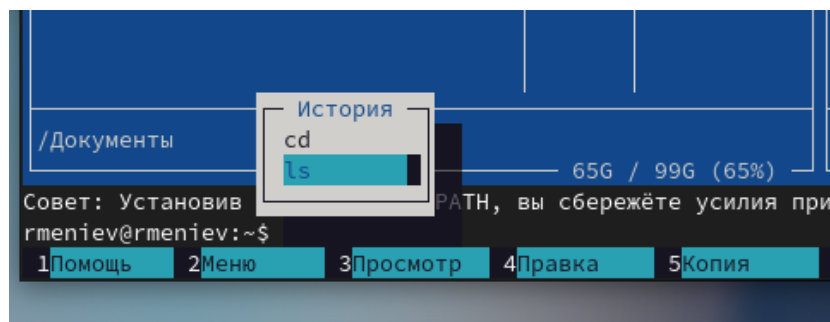


Figure 2.15: История команд

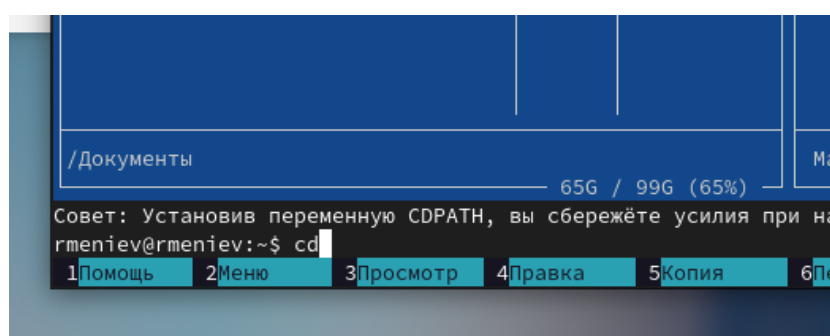


Figure 2.16: Переход в домашний каталог

```
mc.ext.ini [-----] 0 L: [ 1+ 0 1/1136] * (0 /26909b) 0035 0x023 [X]
Midnight Commander 4.8 extension file

# Warning: The structure of this file has been completely changed with the version 4.0!
# All lines starting with # or empty lines are ignored.
# IMPORTANT: mc scans this file only upon first use or after editing it using the
# mc "Edit extension file" command (F9-c-e). If you edit this file in any other way
# while mc is running, you will need to press F9-c-e and exit the editor for your
# changes to take effect, or exit mc and start it again.
#
# Section name can be anything with following exceptions:
#   there are two reserved section names:
#       mc.ext.ini
#       Default
#   special name pattern:
#       Include/xxxx
# See below for more details.
#
# Section [mc.ext.ini] is mandatory. It contains file metadata.
# "Version" parameter is mandatory. It contains the file format version.
#
# Section [Default] is optional. It is applied only if no other match was found.
#
# Sections like [Include/xxxx] can be referenced as "Include-xxxx" from other sections.
# Section [Include/xxxx] can be located as before as after sections that point to it.
#
# Sections are processed from top to bottom, thus the order is important.
# If there are more than one sections with the same name in this file, the first
# section will be used.
#
# [Default] should be a catch-all action and come last.
#
```

Figure 2.17: Просмотр файла расширений

```
menu [-----] 1 L: [ 1+16 17/370] * (732 /11821b) 0010 0x00A [X]
shell_patterns=0

#####
# % The % character
# %f The current file (if non-local vfs, file will be copied locally and
# %f will be full path to it)
# %p The current file
# %d The current working directory
# %s "Selected files"; the tagged files if any, otherwise the current file
# %t Tagged files
# %u Tagged files (and they are untagged on return from expand_format)
# %view Runs the commands and pipes standard output to the view command
# If %view is immediately followed by '(', recognize keywords
# ascii, hex, nroff and uniform
#
# If the format letter is in uppercase, it refers to the other panel
#
# With a number followed the % character you can turn quoting on (default)
# and off. For example:
# %f quote expanded macro
# %lf ditto
# %0f don't quote expanded macro
#####
+ ! t t
@ Do something on the current file
CMD=%(Enter command)
$CMD %f

+ t t
@ Do something on the tagged files
CMD=%(Enter command)
for i in %t ; do

1Помощь 2Сохранить 3Слук 4Замена 5Копия 6Перем-тить 7Поиск 8Удалить 9НенжМС 10Выход
```

Figure 2.18: Просмотр файла меню

## 7. Вызовем подменю Настройки. Изучим опции

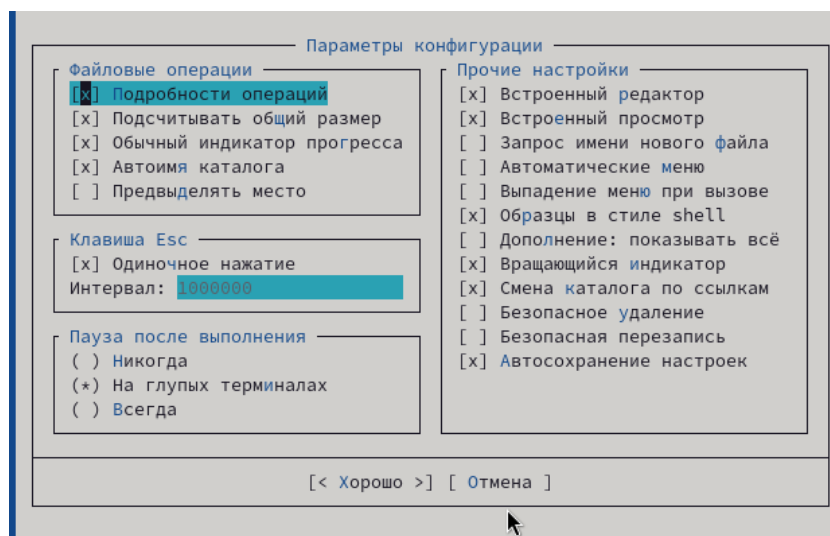


Figure 2.19: Конфигурация

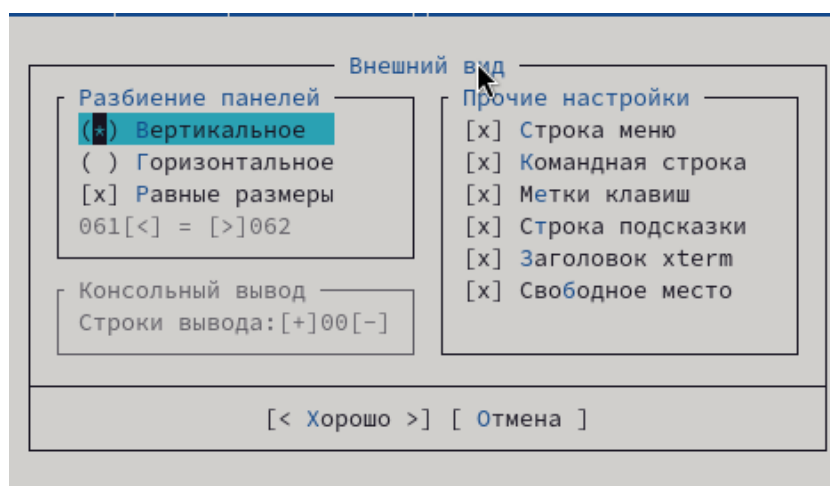


Figure 2.20: Внешний вид

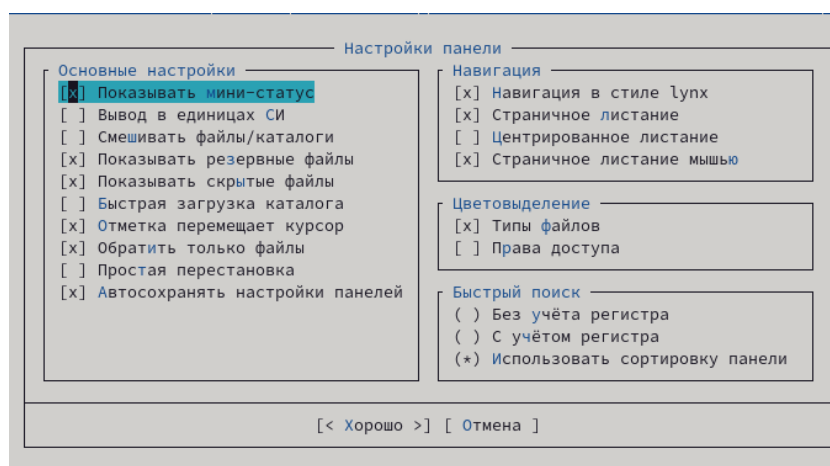


Figure 2.21: Настройки панелей

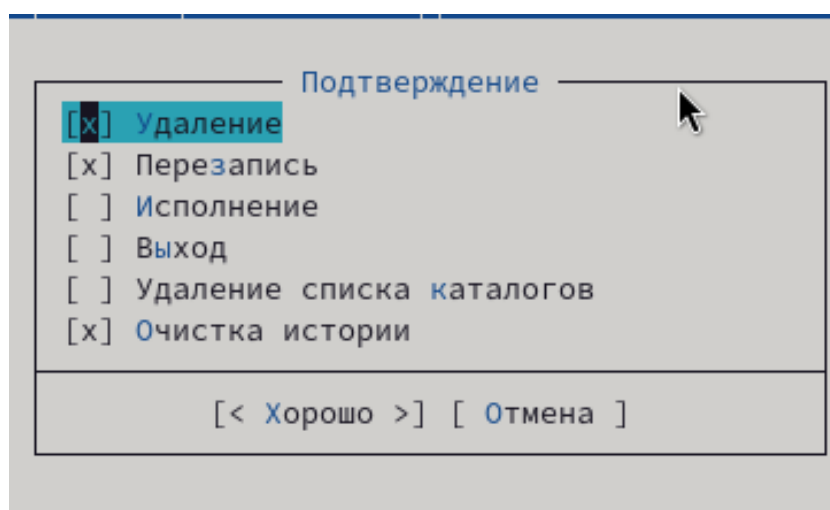


Figure 2.22: Подтверждение

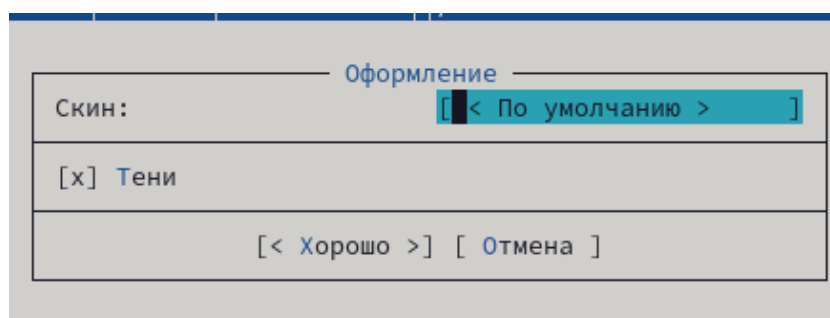


Figure 2.23: Оформление

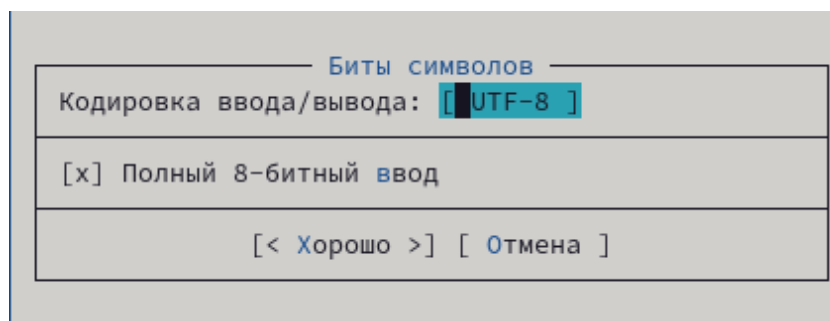


Figure 2.24: Кодировка символов

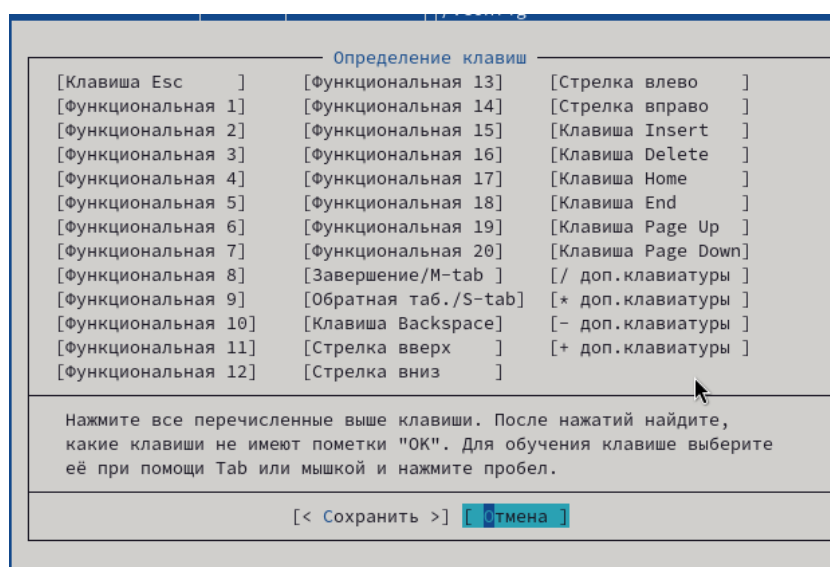
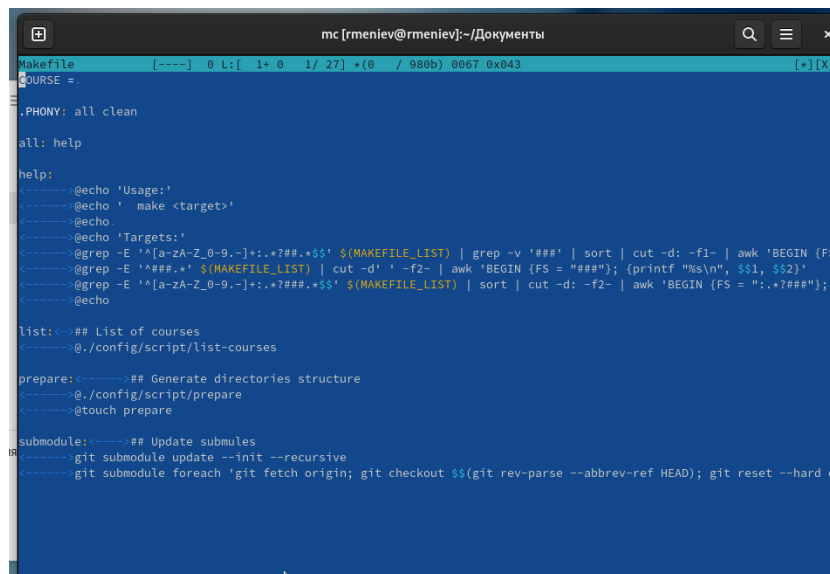


Figure 2.25: Распознавание клавиш

8 Создадим текстовый файл text.txt.

9 Откроем этот файл с помощью встроенного в тс редактора, и вставим в открытый файл небольшой фрагмент текста, скопированный из любого другого файла или Интернета.



```
mc [rmeniev@rmeniev]:~/Документы
Makefile  [----]  0 L: [ 1+ 0 1/ 27] *(0 / 980b) 0067 0x043  [*][X]
COURSE = .
.PHONY: all clean
all: help
help:
-----@echo 'Usage:'
-----@echo ' make <target>'
-----@echo
-----@echo 'Targets:'
-----@grep -E '[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | grep -v '###' | sort | cut -d: -f1- | awk 'BEGIN {FS
-----@grep -E '^###.*' $(MAKEFILE_LIST) | cut -d' ' -f2- | awk 'BEGIN {FS = "###"; {printf "%s\n", $$1, $$2}'
-----@grep -E '[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | sort | cut -d: -f2- | awk 'BEGIN {FS = ".*?##"}';
-----@echo

list:<>## List of courses
-----@./config/script/list-courses

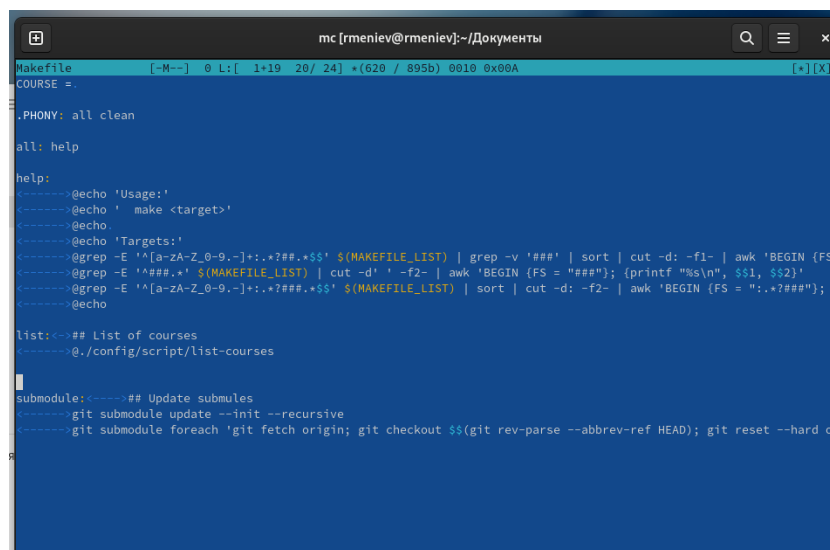
prepare:<-----## Generate directories structure
-----@./config/script/prepare
-----@touch prepare

submodule:<-----## Update submules
-----@git submodule update --init --recursive
-----@git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
```

Figure 2.26: Файл с текстом

10 Прделаем с текстом следующие манипуляции, используя горячие клавиши:

Удалим строку текста. - F8



```
mc [rmeniev@rmeniev]:~/Документы
Makefile  [----]  0 L: [ 1+19 20/ 24] *(620 / 895b) 0010 0x00A  [*][X]
COURSE = .
.PHONY: all clean
all: help
help:
-----@echo 'Usage:'
-----@echo ' make <target>'
-----@echo
-----@echo 'Targets:'
-----@grep -E '[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | grep -v '###' | sort | cut -d: -f1- | awk 'BEGIN {FS
-----@grep -E '^###.*' $(MAKEFILE_LIST) | cut -d' ' -f2- | awk 'BEGIN {FS = "###"; {printf "%s\n", $$1, $$2}'
-----@grep -E '[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | sort | cut -d: -f2- | awk 'BEGIN {FS = ".*?##"}';
-----@echo

submodule:<-----## Update submules
-----@git submodule update --init --recursive
-----@git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
```

Figure 2.27: Файл с текстом

Выделим фрагмент текста и скопируйте его на новую строку. - F5



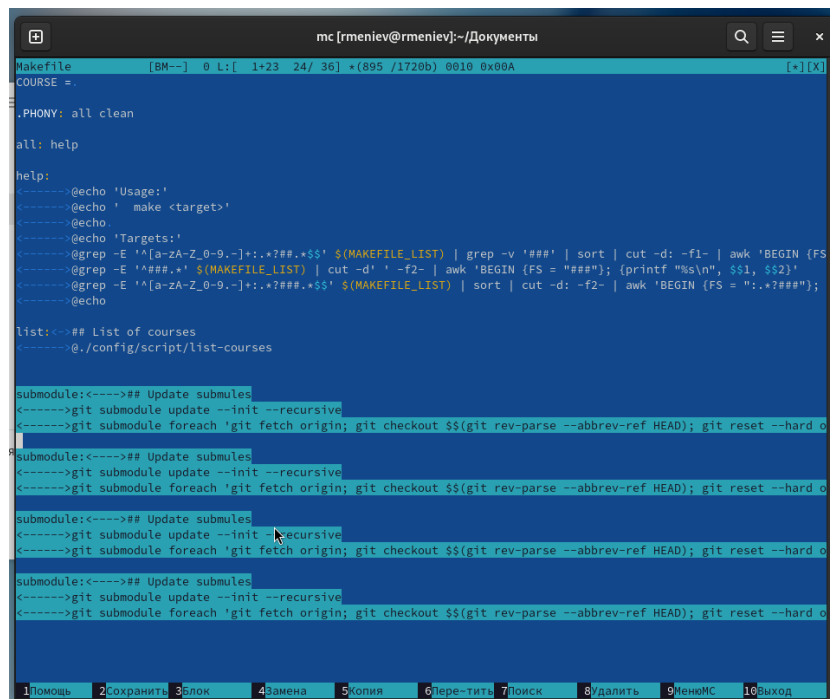


Figure 2.28: Копирование фрагмента

Сохраним файл. - F2

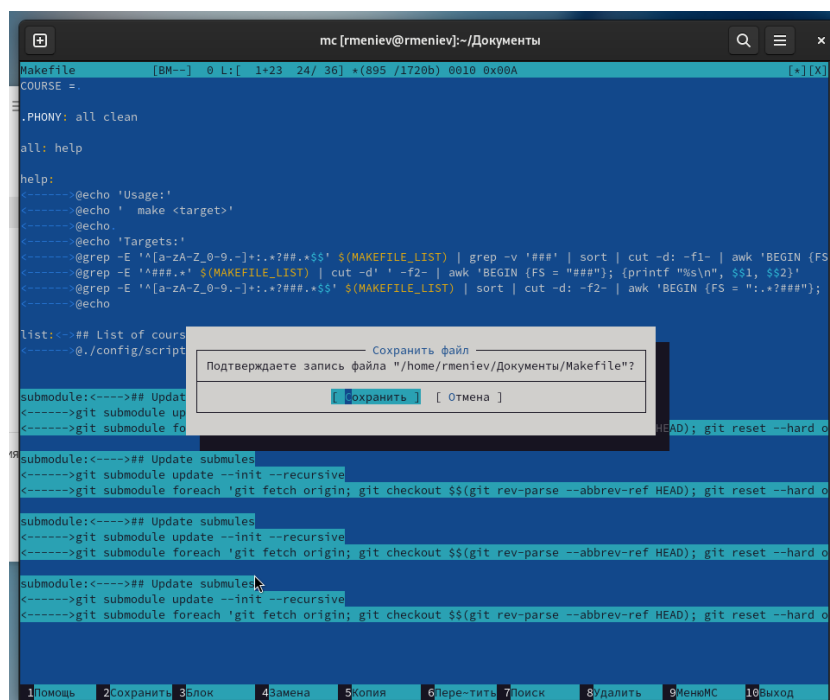
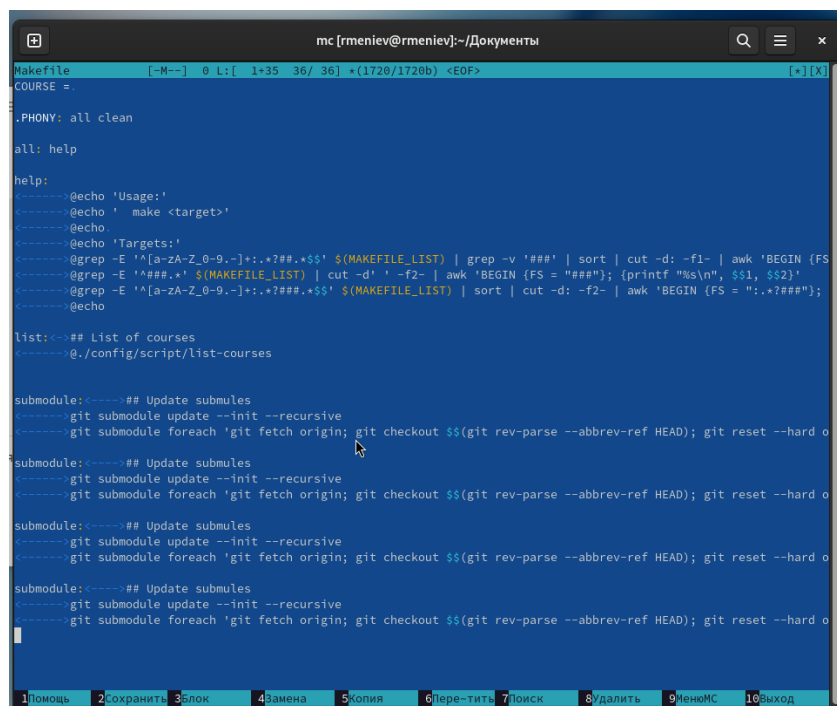


Figure 2.29: Сохранение

Отменим последнее действие. - Ctrl+U



```
mc [rmeniev@rmeniev:~/Документы]
Makefile [-M--] 0 L: 1+35 36/ 36) *(1720/1720b) <EOF> [*] [X]
COURSE =
.PHONY: all clean
all: help
help:
-----@echo 'Usage:'
-----@echo ' make <target>'
-----@echo
-----@echo 'Targets:'
-----@grep -E '^[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | grep -v '###' | sort | cut -d: -f1- | awk 'BEGIN {FS
-----@grep -E '^###.*' $(MAKEFILE_LIST) | cut -d' ' -f2- | awk 'BEGIN {FS = "###"; (printf "%s\n", $1, $2)'
-----@grep -E '^[a-zA-Z_0-9-]+:.*?##.*$$' $(MAKEFILE_LIST) | sort | cut -d: -f2- | awk 'BEGIN {FS = "###";
-----@echo

list:<>## List of courses
-----@./config/script/list-courses

submodule:<---->## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $$ (git rev-parse --abbrev-ref HEAD); git reset --hard o

submodule:<---->## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $$ (git rev-parse --abbrev-ref HEAD); git reset --hard o

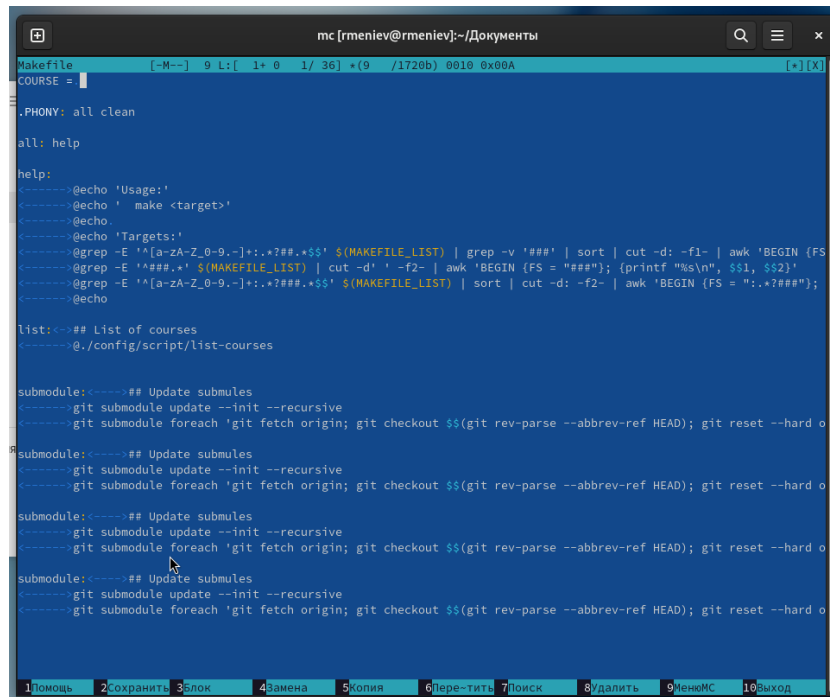
submodule:<---->## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $$ (git rev-parse --abbrev-ref HEAD); git reset --hard o

submodule:<---->## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $$ (git rev-parse --abbrev-ref HEAD); git reset --hard o

1Помощь 2Сохранить 3Злокт 4Замена 5Копия 6Пере-тить 7Поиск 8Удалить 9МенюМС 10Выход
```

Figure 2.30: Отмена

Перейдем в конец файла - PageDown или Ctrl+X



```
mc [rmeniev@rmeniev]:~/Документы
Makefile [-M--] 9 L: [ 1+ 0 1/ 36] *(9 /1720b) 0010 0x00A [*][X]
COURSE =
.PHONY: all clean
all: help
help:
-----@echo 'Usage:'
-----@echo ' make <target>'
-----@echo
-----@echo 'Targets:'
-----@grep -E '^[a-zA-Z_0-9-]*:.*?##.*$$' $(MAKEFILE_LIST) | grep -v '###' | sort | cut -d: -f1- | awk 'BEGIN {FS
-----@grep -E '###.*' $(MAKEFILE_LIST) | cut -d' ' -f2- | awk 'BEGIN {FS = "###"; (printf "%s\n", $$1, $$2)'
-----@grep -E '^[a-zA-Z_0-9-]*:.*?##.*$$' $(MAKEFILE_LIST) | sort | cut -d: -f2- | awk 'BEGIN {FS = ".*?##"}';
-----@echo

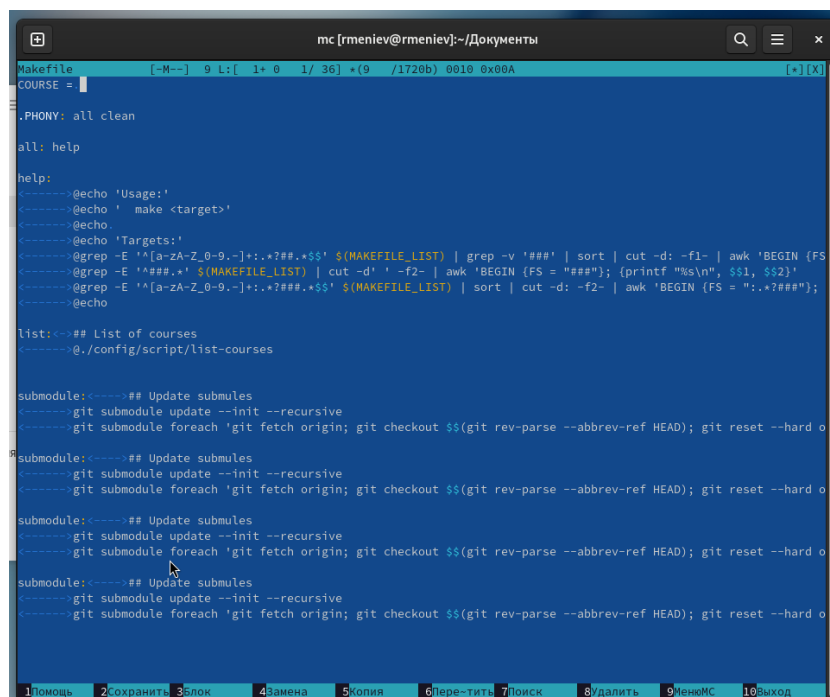
list:<>## List of courses
-----@./config/script/list-courses

submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Пере-тите 7Поиск 8/далить 9ленюМС 10Выход
```

Figure 2.31: Переход в конец файла

Перейдем в начало файла - PageUp или Ctrl+Z



```
mc [rmeniev@rmeniev]:~/Документы
Makefile [-M--] 9 L: [ 1+ 0 1/ 36] *(9 /1720b) 0010 0x00A [*][X]
COURSE =
.PHONY: all clean
all: help
help:
-----@echo 'Usage:'
-----@echo ' make <target>'
-----@echo
-----@echo 'Targets:'
-----@grep -E '^[a-zA-Z_0-9-]*:.*?##.*$$' $(MAKEFILE_LIST) | grep -v '###' | sort | cut -d: -f1- | awk 'BEGIN {FS
-----@grep -E '###.*' $(MAKEFILE_LIST) | cut -d' ' -f2- | awk 'BEGIN {FS = "###"; (printf "%s\n", $$1, $$2)'
-----@grep -E '^[a-zA-Z_0-9-]*:.*?##.*$$' $(MAKEFILE_LIST) | sort | cut -d: -f2- | awk 'BEGIN {FS = ".*?##"}';
-----@echo

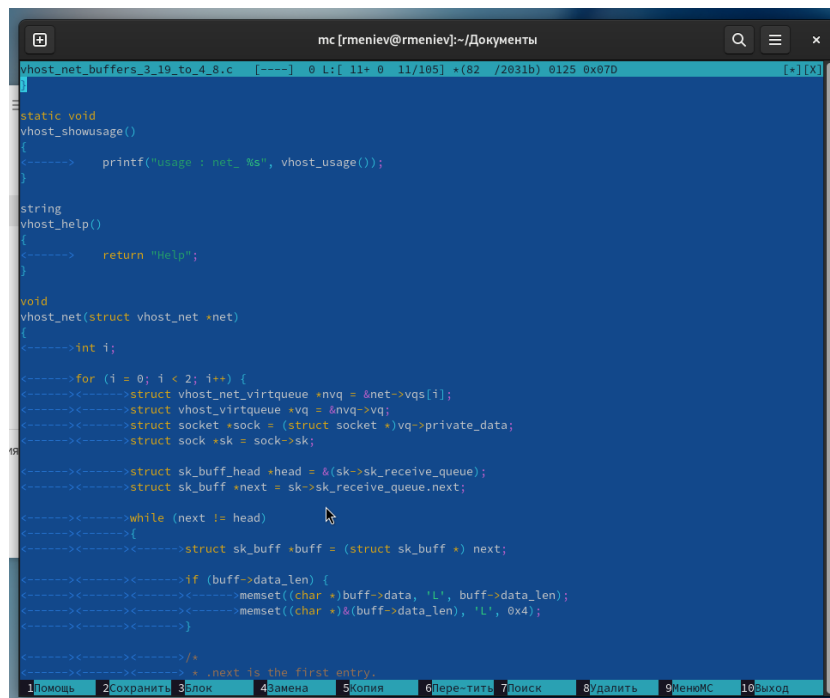
list:<>## List of courses
-----@./config/script/list-courses

submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o
submodule:-----## Update submules
-----git submodule update --init --recursive
-----git submodule foreach 'git fetch origin; git checkout $(git rev-parse --abbrev-ref HEAD); git reset --hard o

1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Пере-тите 7Поиск 8/далить 9ленюМС 10Выход
```

Figure 2.32: Переход в начало файла

11 Откроем файл с исходным текстом на языке программирования C



The screenshot shows a text editor window titled 'mc [rmeniev@rmeniev:~/Документы]'. The file being edited is 'vhost\_net\_buffers\_3\_19\_to\_4\_8.c'. The code is written in C and includes several functions: 'vhost\_showusage()', 'vhost\_help()', and 'vhost\_net()'. The 'vhost\_net()' function is a void function that takes a 'struct vhost\_net \*net' as an argument. It contains a loop for 'i' from 0 to 2, where it initializes 'struct vhost\_net\_virtqueue \*nvq', 'struct vhost\_virtqueue \*vq', 'struct socket \*sock', and 'struct sock \*sk'. It also initializes 'struct sk\_buff\_head \*head' and 'struct sk\_buff \*next'. A 'while' loop is used to traverse the 'sk\_buff' list. The code uses various macros and constants, such as 'L', 'L', '0x4', and '0x07D'. The editor has a menu bar at the bottom with options: '1.Помощь', '2.Сохранить', '3.Блок', '4.Замена', '5.Копия', '6.Пере-тит', '7.Поиск', '8.Удалить', '9.ЛениМС', and '10.Выход'.

```
mc [rmeniev@rmeniev:~/Документы]
vhost_net_buffers_3_19_to_4_8.c [----] 0 L: [ 11+ 0 11/105] +(82 /2831b) 0125 0x07D [*)(X]

static void
vhost_showusage()
{
    printf("usage : net_ %s", vhost_usage());
}

string
vhost_help()
{
    return "Help";
}

void
vhost_net(struct vhost_net *net)
{
    int i;

    for (i = 0; i < 2; i++) {
        struct vhost_net_virtqueue *nvq = &net->vqs[i];
        struct vhost_virtqueue *vq = &nvq->vq;
        struct socket *sock = (struct socket *)vq->private_data;
        struct sock *sk = sock->sk;

        struct sk_buff_head *head = &(sk->sk_receive_queue);
        struct sk_buff *next = sk->sk_receive_queue.next;

        while (next != head)
        {
            struct sk_buff *buff = (struct sk_buff *) next;

            if (buff->data_len) {
                memset((char *)buff->data, 'L', buff->data_len);
                memset((char *)&(buff->data_len), 'L', 0x4);
            }

            /*
             * next is the first entry.
            */
        }
    }
}
```

Figure 2.33: Файл с программой

12 Используя меню редактора, выключим подсветку синтаксиса.

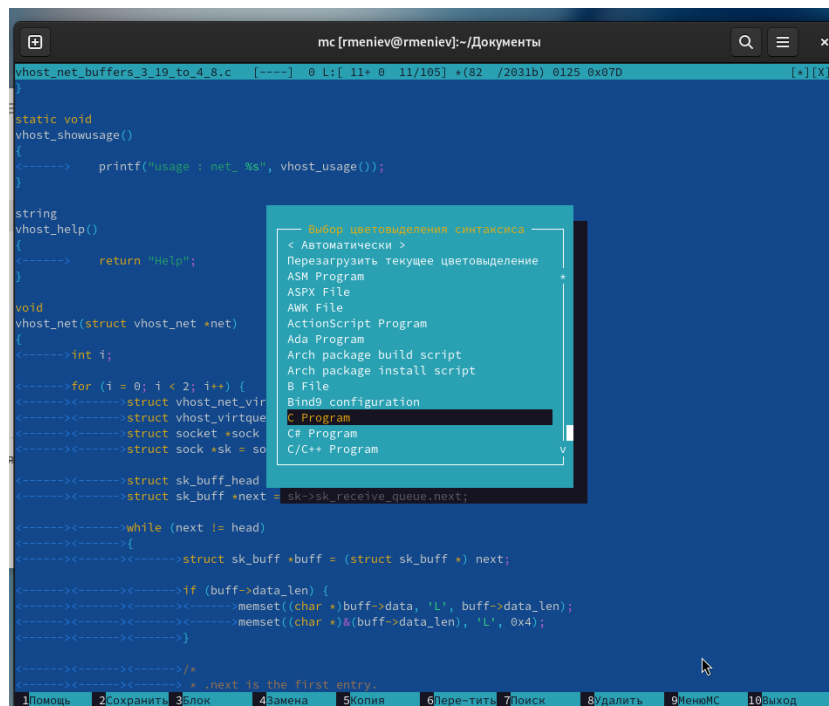


Figure 2.34: Цветовыделение синтаксиса

## **3 Вывод**

В данной работе мы ознакомились с инструментами командной оболочки Midnight Commander. Приобрели навыки практической работы по просмотру каталогов и файлов

## 4 Контрольные вопросы

1. Какие режимы работы есть в tc? Охарактеризуйте их. Ответ: В командной оболочке tc есть два режима Информация и Дерево. В режиме Информация на панель выводятся сведения о файле и текущей файловой системе, расположенных на активной панели. В режиме Дерево на одной из панелей выводится структура дерева каталогов. Управлять режимами отображения панелей можно через пункты меню tc
2. Какие операции с файлами можно выполнить как с помощью команд shell, так и с помощью меню tc? Привести несколько примеров. Ответ: Командные интерпретатор Shell и оболочка Midnight Commander имеют похожую структуру и многие одинаковые команды можно выполнить в обоих оболочках вот некоторые из них
  - a) Системная информация
  - b) Поиск
  - c) Копирование
3. Опишите структуру меню левой панели tc, дайте характеристику командам. Ответ: Меню левой панели tc представляет собой следующую конструкцию:

Где подпункты меню а) Список файлов показывает файлы в домашнем каталоге. б) Быстрый просмотр позволяет выполнить быстрый просмотр содержимого панели.

с) Информация позволяет посмотреть информацию о файле или каталоге d) Командная оболочка Midnight Commander В меню каждой (левой или правой) панели можно выбрать Формат списка: стандартный, ускоренный, расширенный и определённый пользователем. е) Порядок сортировки позволяет задать критерии сортировки при выводе списка файлов и каталогов: без сортировки, по имени, расширенный, время правки, время доступа, время изменения атрибута, размер, узел.

4. Опишите структура меню Файл mc и дайте характеристику командам. Ответ: Меню Фаил mc представляет собой следующую конструкцию:

Где подпункты меню а) Просмотр ( F3 ) позволяет посмотреть содержимое текущего файла без возможности редактирования. b) – Просмотр вывода команд ( M + ! ) функция запроса команды с параметрами. с) Правка ( F4 ) открывает текущий (или выделенный) файл для его редактирования. d) Копирование ( F5 ) осуществляет копирование одного или нескольких файлов или каталогов в указанное пользователем во всплывающем окне место. е) Права доступа ( Ctrl-x c ) позволяет изменить права доступа к одному или нескольким файлам или каталогам. f) Права доступа на файлы и каталоги g) Жёсткая ссылка ( Ctrl-x l ) позволяет создать жёсткую ссылку к текущему (или выделенному) файлу1 . h) Символическая ссылка ( Ctrl-x s ) — позволяет создать символическую ссылку к текущему файлу . i) Владелец группы ( Ctrl-x o ) позволяет задать владельца и имя группы для одного или нескольких файлов или каталогов. j) Права (расширенные) позволяет изменить права доступа и владения для одного или нескольких файлов или каталогов. k) Переименование ( F6 ) позволяет переименовать один или несколько файлов или каталогов. l) Создание каталога ( F7 ) позволяет создать каталог. m) Удалить ( F8 ) позволяет удалить один или несколько файлов или каталогов. n) Выход ( F10 ) завершает работу mc.

5 Опишите структура меню Команда mc, дайте характеристику командам Ответ: Ответ: Меню Команда mc представляет собой следующую конструкцию:



Где подпункты меню а) Дерево каталогов отображает структуру каталогов системы. б) Поиск файла выполняет поиск файлов по заданным параметрам. с) Переставить панели меняет местами левую и правую панели. d) Сравнить каталоги ( Ctrl-x d ) сравнивает содержимое двух каталогов. е) Размеры каталогов отображает размер и время изменения каталога (по умолчанию в ms размер каталога корректно не отображается). f) История командной строки выводит на экран список ранее выполненных в оболочке команд. g) Каталоги быстрого доступа ( Ctrl- ) при вызове выполняется быстрая смена текущего каталога на один из заданного списка. h) Восстановление файлов позволяет восстановить файлы на файловых системах ext2 и ext3. i) Редактировать файл расширений позволяет задать с помощью определённого синтаксиса действия при запуске файлов с определённым расширением (например, какое программное обеспечение запускать для открытия или редактирования файлов с расширением .с или .сpp). j) Редактировать файл меню позволяет отредактировать контекстное меню пользователя, вызываемое по клавише F2 . k) Редактировать файл расцветки имён позволяет подобрать оптимальную для пользователя расцветку имён файлов в зависимости от их типа.

6. Опишите структура меню Настройки ms, дайте характеристику командам

Ответ: Меню Настройки ms представляет собой следующую конструкцию:

Где подпункты меню а) Конфигурация позволяет скорректировать настройки работы с панелями. б) Внешний вид и Настройки панелей определяет элементы, отображаемые при вызове ms, а также цветовое выделение. с) Биты символов задаёт формат обработки информации локальным терминалом. d) Подтверждение позволяет установить или убрать вывод окна с запросом подтверждения действий при операциях удаления и перезаписи файлов, а также при выходе из программы. е) Распознавание клавиш диалоговое окно используется для тестирования функциональных клавиш, клавиш управления курсором и прочее. f) Виртуальные ФС настройки виртуальной файловой системы: тайм-аут, пароль и прочее.

7. Назовите и дайте характеристику встроенным командам тс. Ответ: В командную оболочку тс встроены стандартные команды. Вот некоторые из них.

- a) F1 Вызов контекстно-зависимой подсказки.
- b) F2 Вызов пользовательского меню с возможностью создания and/or.
- c) F3 Просмотр содержимого файла, на который указывает подсветка в активной панели.
- d) F4 Вызов встроенного в тс редактора для изменения содержания файла, на который указывает подсветка в активной панели.
- e) F5 Копирование одного или нескольких файлов, отмеченных в первой (активной) панели, в каталог, отображаемый на второй панели.
- f) F6 Перенос одного или нескольких файлов, отмеченных в первой панели, в каталог, отображаемый на второй панели.
- g) F7 Создание подкаталога в каталоге, отображаемом в активной панели.
- h) F8 Удаление одного или нескольких файлов, отмеченных в первой панели файлов.
- i) Вызов меню тс.
- j) F10 Выход из тс.

8 Назовите и дайте характеристику командам встроенного редактора тс. Ответ: В редактор тс встроено немало команд. Вот некоторые из них. a) Ctrl+y удалить строку. b) Ctrl+u отмена последней операции. c) Ins вставка/замена. d)F7 поиск. d)Shift+F7 повтор последней операции поиска. e) F4 замена файла. f) F3 первое нажатие начало выделения, второе это окончание выделения. g) F5 копировать выделенный фрагмент F6 переместить выделенный фрагмент. h) F8 удалить выделенный фрагмент. i) F2 записать изменения в файл. j) F10 выйти из редактора.

9. Дайте характеристику средствам тс, которые позволяют создавать меню, определяемые пользователем. Ответ: Один из четырех форматов списка

в Midnight Commander -Пользовательский определённый самим пользователем позволяет ему редактировать меню любого из двух списков. А меню пользователя – это меню, состоящее из команд, определенных пользователем. При вызове меню используется файл ~/.mc.menu. Если такого файла нет, то по умолчанию используется системный файл меню /usr/lib/mc/mc.menu. Все строки в этих файлах , начинающиеся с пробела или табуляции, являются командами, которые выполняются при выборе записи.

10. Дайте характеристику средствам mc, которые позволяют выполнять действия, определяемые пользователем, над текущим файлом
- Ответ: Когда мы выделяем файл не являющегося исполняемым, Midnight Commander сравнивает расширение выбранного файла с расширениями, прописанными в «файле расширений» ~/.mc.ext. Если в файле расширений найдется подраздел, задающий процедуры обработки файлов с данным расширением, то обработка файла производится в соответствии с заданными в этом подразделе командами и файлами:
- a) файл помощи для MC. /usr/lib/mc.hlp
  - b) файл расширений, используемый по умолчанию. /usr/lib/mc/mc.ext
  - c) файл расширений, конфигурации редактора. \$HOME/.mc.ext
  - d) системный инициализационный файл. /usr/lib/mc/mc.ini
  - e) файл который содержит основные установки. /usr/lib/mc/mc.lib
  - f) инициализационный файл пользователя. Если он существует, то системный файл mc.ini игнорируется. \$HOME/.mc.ini
  - g) этот файл содержит подсказки, отображаемые в нижней части экрана. /usr/lib/mc/mc.hint
  - h) системный файл меню MC, используемый по умолчанию. /usr/lib/mc/mc.menu
  - i) файл меню пользователя. Если он существует, то системный файл меню игнорируется. \$HOME/.mc.menu

- j) инициализационный файл пользователя. Если он существует, то системный файл `mc.ini` игнорируется. `$HOME/.mc.tree`