1 Команды операционной системы

Команды операционной системы состоят из *имени* команды или программы, *опций* (или ключей) и *аргументов*. Некоторые опции могут требовать наличия *значения*. Например, в команде

```
gcc -g -o myprog file1.c file2.c
```

gcc – это имя команды, g и o – опции, myprog – значение опции o, file1.c и file2.c — аргументы команды.

Все введённые ранее команды запоминаются. Перейти к предыдущей команде можно путём нажатия клавиши «стрелка вверх». Старые команды можно редактировать. Команда выполняется после нажатия «Enter». При вводе команды нажатие <Tab> вызывает *автоматическое дополнение* имён файлов.

1.1 Работа с файлами

операция	команда	пример
посмотреть список файлов	ls	`ls` или `ls -l` или `ls dirname`
редактировать файл	vi	`vi zadacha.c`
копировать файл	cp	`cp from.c to.c`
копировать файл в директорию	cp	`cp file.c dirname/`
переименовать	mv	`mv old.c new.c`
удалить файл	rm	`rm file.c`
удалить директорию	rm	`rm -rf dirname`
вывести файл на экран	cat	`cat output.txt`
просмотреть с прокруткой	less	`less output.txt` (выход – q)
создать директорию	mkdir	`mkdir mydir`
перейти в директорию	cd	`cd mydir`
перейти в вышестоящую директорию	cd	`cd ..`
перейти в домашнюю директорию	cd	cd или `cd ~`
узнать текущую директорию	pwd	`pwd`

В качестве имени файла может использоваться либо имя (это будет файл в текущей директории), либо *путь*, например `cp dir1/file.c dir2/dir3/` скопирует файл из одной директории в другую (поддиректорию). Специальные символы, которые можно использовать как часть пути: . (точка) – текущая директория, .. (две точки без пробела) – родительская директория, ~ (тильда) – домашняя директория.

Внимание: Удалённые файлы восстановит нельзя. При копировании в существующий файл старая версия файла удаляется.

1.2 Как копировать с/на флэшку

Зайти в систему в графическом режиме. Вставить флэшку в разъём. Должно открыться окно со списком файлов. Если ничего не происходит, то можно попробовать переставить флэшку в другой разъём Копирование файлов и отключение флэшки производится аналогично Windows.

1.3 Компиляция и запуск программ

Для компиляции программ используется команда gcc. Простейшие формы:

`gcc file1.c file2.c`	откомпилировать два файла и создать выполняемый файл a.out
`gcc file1.c -lm`	откомпилировать файл с функциями из `<math.h>`
`gcc -o prog file.c`	имя выполняемого файла будет prog, а не a.out
`gcc -g -o myprog file.c`	включить отладочную информацию

Внимание: заголовочные файлы (.h-файлы) компилировать не надо!

Для запуска своей программы нужно ввести команду `./a.out` (или ./myprog, если имя myprog было указано при компиляции в качестве значения опции `-o`). Если программа работает слишком долго («зациклилась»), то её можно принудительно остановить нажатием «`CTRL+C`» или «`CTRL+\`».

Моя программа заканчивается с ошибкой!! Если запуск программы приводит к аварийному завершению (аварийный останов, ошибка сегментирования, арифметическая ошибка), то можно выяснить место возникновения ошибки. Для этого:

1. компилируем с включением отладочной информации
   `gcc -g file1.c file2.c`
2. запускаем отладчик
   `gdb -q ./a.out`
3. запускаем программу в отладчике
   `run`
4. если требуется, вводим необходимые для работы программы данные
5. при возникновении ошибки печатаем и читаем историю вызова функций
   `where`
6. при необходимости «поднимаемся» на нужный уровень командой `up` и печатаем значения переменных командой `print` (например, `print i`, где i – имя переменной, объявленной в функции)
7. заканчиваем работу отладчика
   `quit`

1.4 Прочие команды

Получение справки: `man <команда>` или `man 3 <имя функции библиотеки C>` Например, команда `man 3 printf` выдаёт справку о стандартной функции printf, а команда `man cp` выдает справку о команде копирования файлов и директорий.
Завершение работы: `logout`

2 Редактор vim

Запуск редактора производится командой

`vim <имя файла>`

например, `vim zadacha1.c`

Редактор vim имеет два режима: режим редактирования и режим команд. При запуске редактор находится в режиме команд. В режиме редактирования нажатие на клавиши как правило приводит к вставки соответствующего символа в текущую позицию. Нажатие на «стрелки» меняет положение курсора. Переход в режим команд производится нажатием клавиши <Esc>.

Все команды выполняются в режиме команд. Перечисленные ниже команды можно всегда начинать с нажатия <Esc> для гарантированного перехода в режим команд. Для многих команд существует полная и краткая форма. Необязательная часть команды ниже приводится в квадратных скобках. Минимально необходимые знания: команды вставки текста, выход из редактора и ошибки при запуске.

Выход из редактора

:q[uit]	Выход без сохранения. Если файл был изменён – ошибка.
:q[uit]!	Выход без сохранения.
:wq	Выход с сохранением (записать и выйти)
ZZ	Выход с сохранением (без двоеточия)

Перемещение курсора

nG	перейти на строку с номером n, например, 12G
G	перейти в конец файла
gg	перейти в начало файла
0	(ноль) начало строки
^	первый непробельный символ строки
$	последний символ строки
w	перейти на начало следующего слова
e	перейти на конец слова
W	перейти на начало расширенного слова (не пробелы)
E	перейти на конец расширенного слова
%	если курсор стоит на скобке, то перейти на соответствующую открывающую или закрывающую скобку
}	начало следующего абзаца (разделитель – пустая строка)
{	начало следующего абзаца (разделитель – пустая строка)

Команды вставки текста

i	начать ввод перед курсором (перейти в режим редактирования)
I	начать ввод перед первым непробельным символом строки
a	начать ввод после курсора
A	начать ввод после последнего символа строки
o	(строчная буква o) вставить строку ниже курсора и перейти в режим редактирования
O	(заглавная буква O) вставить строку выше курсора и перейти в ражим редактирования

Отмена и повторное выполнение

:u[ndo]	отмена последнего действия
u	отмена последнего действия
nu	n – число; отменить последние n действий
:red[o]	повторно выполнить последнее отменённое действие
CTRL+R	повторно выполнить отменённое действие
.	повторить выполнение последней операции

Вставка, удаление и копирование строк

dd	удалить текущую строку и скопировать её в буфер
$n$dd	n – число; удалить n строк
yy	скопировать текущую строку в буфер обмена
$n$yy	n – число; скопировать n строк в буфер
p	вставить содержимое буфера после курсора (строки ниже)
P	вставить содержимое буфера перед курсором (строки выше)

Режим визуального выделения

v	начать выделение блока («стрелками») по символам
V	начать выделение блока по строчкам
<Esc>	выйти из режима визуального выделения

Выделенный блок можно скопировать (y), отформатировать (=), удалить (d).

Поиск и замена

`/шаблон/`	найти в файле указанный шаблон (например, слово)
#	найти все вхождения слова под курсором
n	перейти к следующему вхождению (повторить поиск)
N	повторить поиск в обратном направлении (предыдущее вхождение)
`:%s/old/new/g`	заменить в файле все вхождения old на new
:noh[lsearch]	убрать подсветку найденных фрагментов

Автоматическое форматирование исходного кода

Команда `gg=G` расставит отступа в вашем файле.

Работа с несколькими файлами

В редакторе vim можно одновременно открыть несколько файлов при запуске (например, `vim file1.c file2.c`). Переключение между открытыми файлами производится командами `:bn[ext]` и `:bp[rev]`.

В одном файле можно скопировать часть текста, например, скопировать в буфер 10 строк командой `10yy`, и вставить их в другой файл командой `p`.

Ошибки при запуске vim

Если при выполнении команды `vim file.c` вместо Вашего файла выводится сообщение `Found a swap file by the name ".file.c.swp"` и много других строк, то вероятнее всего Вы выключили компьютер без выхода из редактора или случайно нажали «`CTRL-Z`» в процессе редактирования этого файла.

Решение: нажимаем `q` (вернулись в оболочку операционной системы), вводим команду `fg` и, если на экране не появился редактор (печатается сообщение «`no such job`»), то вводим команду «`rm .file.c.swp`», где file.c нужно заменить на имя Вашего файла.