Справочный материал по ОС UNIX
для выполнения лабораторных работ по курсу
«CS221. Архитектура компьютера и операционные системы»

ОСНОВНЫЕ КОМАНДЫ UNIX-ПОДОБНЫХ ОПЕРАЦИОННЫХ СИСТЕМ

Интерфейс командной строки

Будем для краткости называть UNIX-подобные операционные системы UNIX-системами или просто UNIX.

В UNIX базовый уровень общения с пользователем заключается во вводе с клавиатуры команд и просмотре выводимой текстовой информации на дисплее (такой способ общения часто называют «интерфейсом командной строки»). Устройства ввода и вывода текста вместе называют терминалом.

Регистрация

Во всех UNIX-подобных ОС, установленных на компьютере, имеется некоторая база данных пользователей, имеющих право использования ресурсов этого компьютера. Пользователей, не включенных в этот список, система к работе не допустит. База данных ведется администратором компьютера и содержит для каждого пользователя следующую информацию:

- регистрационное имя;
- зашифрованный пароль;
- идентификатор пользователя (User ID – UID);
- список групп, в которые включен пользователь;
- путь к командной оболочке;
- путь к домашнему каталогу;
- другую дополнительную информацию.

Регистрационное имя и пароль необходимы для процедуры регистрации пользователя. Идентификатор UID используется внутренними функциями системы, самим пользователем он используется редко. Механизм групп позволяет объединять пользователей по определенным полномочиям на доступ к файлам и программам. Путь к командной оболочке нужен для ее запуска после процедуры регистрации. И, наконец, домашний каталог – это обычно место в файловой системе, целиком принадлежащее данному пользователю, но доступное, конечно, и администратору.

При работе в компьютерных классах мехмата (после прохождения авторизации в UNIX) для запуска терминала следует выбрать приложение LXTerminal, расположенное на Рабочем столе.

После авторизации пользователя с указанным именем, система сделает домашний каталог пользователя текущим. При запуске терминала также обычно выполняется некоторый начальный набор команд, который может вывести приветственное сообщение (все эти действия зависят от особенностей настройки конкретной ОС). И, наконец, на экране появится приглашение к вводу команды:

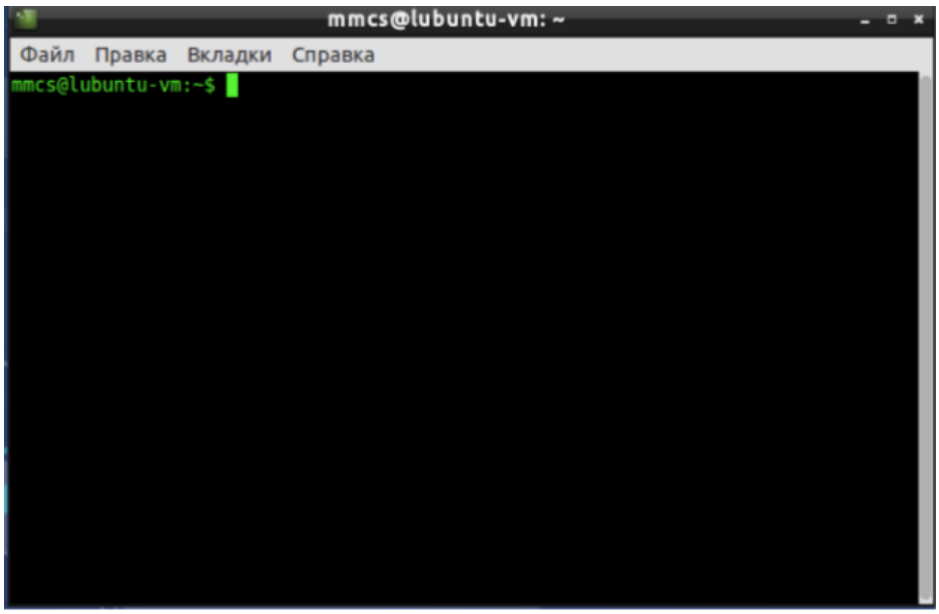

Рис. 1 Вид окна программы LXTerminal.

Приглашение не обязательно имеет такой вид, как показано выше. Оно зависит от конкретной командной оболочки и ее конфигурации. Далее в примерах, приведённых в тексте материалов будем считать, что приглашение к вводу команды имеет более простой и короткий вид:

:~$

В компьютерных классах мехмата и в той конфигурации Linux, что рекомендована к скачиванию на странице курса в Moodle, вызов терминала назначен на нажатие «горячих» клавиш Ctrl+Alt+T.

Командные оболочки UNIX

В UNIX используются различные командные оболочки (command shells), называемые также командными процессорами или интерпретаторами команд. По умолчанию в компьютерных классах мехмата установлен в качестве оболочки bash.

Основными функциями командных оболочек являются:

- организация диалога с пользователем (ввод команд);
- выполнение внутренних команд;
- запуск внешних программ;
- исполнение командных файлов.

Команды UNIX и запуск программ

Общий синтаксис команд в UNIX выглядит следующим образом:

```
имя_команды [ключи ...] [параметры ...]
```

Первый элемент обозначает конкретную команду, аргументы (ключи и параметры) могут сообщать дополнительную информацию. Ключи обычно начинаются со знака «минус». Например, команда

```
:~$ ls -l -a /home
```

состоит из:

- имени команды «ls», выводящей список файлов в заданном каталоге;
- ключа (модификатора) «l», указывающего, что нужно вывести подробный листинг;
- ключа «a», указывающего, что нужно выводить все файлы, включая служебные;
- параметра «/home», задающего путь к каталогу.

В командах ОС UNIX, их ключах и параметрах регистр букв (строчные или заглавные) различается. Для большей части команд характерна запись строчными буквами. Ключи во многих случаях могут объединяться в одну группу. Например, команда

```
:~$ ls -la /home
```

полностью эквивалентна рассмотренной выше.

Команда echo

Команда echo - это очень простая и в то же время часто используемая встроенная команда оболочки bash. Она имеет только одно назначение - выводить строку текста в терминал, но применяется очень часто в различных скриптах, программах, и даже для редактирования конфигурационных файлов. Команда echo - это не системная утилита, у нее нет исполняемого файла. Она существует только внутри интерпретатора bash. Синтаксис команды echo:

```
:~$ echo [ключи … ] строка
```

Ключей всего несколько:

- -n - не выводить перевод строки;
- -e - включить поддержку вывода Escape последовательностей;
- -E - отключить интерпретацию Escape последовательностей.

Если включена опция -e, то вы можете использовать такие Escape последовательности для вставки специальных символов:

- /c - удалить перевод строки;
- /t - горизонтальная табуляция;
- /v - вертикальная табуляция;
- /b - удалить предыдущий символ;
- /n - перевод строки;
- /r - символ возврата каретки в начало строки.

Получение справочной информации

По ходу изучения операционной системы UNIX вам часто будет требоваться информация о том, что делает та или иная команда или системный вызов, какие у них параметры и опции, для чего предназначены некоторые системные файлы, каков их формат и т.д. Самая полная информация содержится в UNIXManual – руководству по

операционной системе, которая доступна в интерактивном режиме с помощью утилиты man.

Пользоваться утилитой man достаточно просто – наберите команду

```
man имя
```

где имя – это имя интересующей вас команды, утилиты, системного вызова, библиотечной функции или файла.

Чтобы пролистать страницу полученного описания, если оно не поместилось на экране полностью, следует нажать клавишу <пробел>. Для прокрутки одной строки воспользуйтесь клавишей <Enter>. Вернуться на страницу назад позволит одновременное нажатие клавиш <Ctrl> и <b>. Выйти из режима просмотра информации можно с помощью клавиши <q>.

Простейшие команды для работы с файловой системой

Команда изменения текущего каталога:

```
cd [имя каталога]
```

Если команда cd вызвана без аргументов, текущим каталогом станет домашний каталог пользователя. Чтобы вывести на экран полное имя текущего каталога, нужно использовать команду pwd без аргументов.

Команда

```
ls [имя_каталога]
```

позволяет получить листинг указанного каталога. Если имя каталога не указано, то будет выведен листинг текущего каталога. У команды ls есть несколько полезных ключей

- l – вывести полную информацию о каждом файле;
- a – вывести листинг всех файлов, включая такие, имена которых начинаются с символа точки.

Команды mkdir и rmdir позволяют, соответственно, создать или удалить указанный каталог:

```
mkdir [имя каталога]
rmdir [имя каталога]
```

Команда просмотра файлов less позволяет просматривать файлы произвольного размера и перемещаться по их содержимому с помощью клавиш управления курсором (для выхода используется клавиша «q»):

```
less [имя файла]
```

Команда копирования файлов:

```
cp [источник] [приемник]
```

Команда перемещения или переименования файлов:

```
mv [источник] [приемник]
```

Команда удаления файлов:

```
rm [имя файла]
```

С командами cp и rm может использоваться ключ «r», позволяющий копировать, перемещать или удалять каталоги со всем их содержимым рекурсивно.

Для полной информации о перечисленных командах, их аргументах и вариантах их использования можно обратиться к страницам руководства пользователя (команда man).

Стандартные потоки ввода-вывода

С каждой программой, запускаемой из командной строки UNIX, связаны три стандартных потока данных:

- стандартный поток ввода (stdin);
- стандартный поток вывода (stdout);
- стандартный поток ошибок (stderr).

Программы, требующие входных данных, обычно читают информацию из стандартного потока ввода. Стандартный поток вывода – это поток, куда программы записывают выходные данные. Большинство неинтерактивных команд (включая echo, pwd, ls и другие) работают со стандартными потоками ввода и вывода. Но эти потоки можно переопределить (перенаправить). Для связывания стандартного потока вывода с файлом используется операция «*>*», например:

```
:~$ ls > filelist.txt
```

В этом примере команда ls, вместо того, чтобы вывести список файлов на экран, записала его в файл с именем «filelist.txt». При этом, если файл с таким именем не существовал, он будет создан, в противном случае его старое содержимое будет потеряно.

Стандартный поток ввода перенаправляется при помощи операции «<».

Существует и другая возможность перенаправления вывода, когда новые выходные данные будут дописаны в конец существующего файла. Для этого используется операция (метасимвол) «>>». В следующем примере текущие дата и время будут дописаны в конец файла с именем «dates.txt»:

```
:~$ date >> dates.txt
```

Наряду с только что описанным способом дописывания информации в файл в UNIX есть отдельные команды, выполняющие подобные действия. В частности, команда cat (это сокращение от concatenate). На самом деле это некая утилита, которая позволяет вам сцеплять, связывать файлы и т.д. Она может выводить содержимое файла на стандартный вывод и многое другое в зависимости от командной строки, в которой используется.

Общий вид команды таков:

```
cat [filename1] [filename2... ]
```

Приведённая команда выводит содержимое перечисленных файлов (filename1, и т.д.) на стандартный поток вывода. Если ни одно имя не указано, то cat получает данные со стандартного потока ввода и выводит их в стандартный поток вывода. Благодаря использованию метасимволов перенаправления потоков можно объединить несколько файлов в один и дописать все это в конец 3-го файла. При этом команду cat следует записать в следующем виде:

```
:~$ cat filename1 filename2 >> filename3
```

Сообщения об ошибках выводятся в стандартный поток ошибок. Например, пусть выполняется попытка получить список файлов в каталоге без соответствующих прав доступа:

```
:~$ ls -l /home/ftp/bin/
ls: cannot access '/home/ftp/bin': No such file or directory
```

В данном случае команда ls вывела сообщение в поток стандартной ошибки. Чтобы перенаправить его в указанный файл, можно использовать операции «2>» и «2>>» (по аналогии с «>» и «>>», только цифра 2 говорит о том, что нужно перенаправить поток ошибок), например:

```
:~$ ls -l /home/ftp/bin/ 2> last-error.txt
```

Операции перенаправления ввода-вывода можно комбинировать, например:

```
:~$ wc < /etc/passwd 2>> errors.txt > result.txt
```

Существует другой полезный способ перенаправления ввода-вывода – конвейеры команд. Операция «|» (знак вертикальной черты) позволяет перенаправить стандартный поток вывода одной команды на стандартный входной поток другой команды:

```
:~$ ls -l /etc | less
```

В этом примере команда ls выводит длинный список файлов в каталоге /etc, эти данные попадают на вход программы less, которая позволяет пролистывать текст с помощью клавиш управления курсором. Так осуществляется «объединение» двух независимых команд в один «конвейер».

Простейшие команды для работы с содержимым файла

Команда grep сопоставляет строки исходных указанных файлов с шаблоном “строка”, заданным ограниченным регулярным выражением. Если файлы не указаны,

используется стандартный поток ввода. Результат выполнения по умолчанию выдается в стандартный поток вывода. Общий вид команды.

```
grep [опции] строка [файл][файл] ...
```

Наиболее употребимые опции:

- i поиск без учета регистра
- n отображать номера строк, содержащих контекст “строка”
- v отображать строки, не содержащие контекст “строка”.

К полезным функциям работы с содержимым файла можно отнести

```
tail [опции] filename
```

просмотр конца файла. По умолчанию 10 последних строк. С помощью опций можно начать просмотр с любой позиции:

- n number просмотр с указанной строки
- r number отображение в обратном порядке
- f непрерывная выдача файла по мере его заполнения

Команда find может искать файлы по имени, размеру, дате создания или модификации и некоторым другим критериям. Общий синтаксис команды find имеет следующий вид:

```
find [список_каталогов] критерий_поиска
```

Параметр "список_каталогов" определяет, где искать нужный файл. Пример.

```
:~$ find /usr/share /usr/src -name readme.txt
/usr/share/doc/libcolumbus0-0/readme.txt
```

Завершение работы с UNIX

Каждый сеанс работы с ОС UNIX должен заканчиваться вводом команды logout или exit. Также можно использовать комбинацию клавиш Ctrl-D, которая позволяет выполнить команду завершения работы с командной оболочкой, после чего система переходит в режим ожидания регистрации следующего пользователя.

ФАЙЛОВАЯ СИСТЕМА

Особенности формирования файлового пространства

Файловое пространство UNIX–систем представляет собой иерархию файлов, которая имеет единый общий корень – так называемый корневой каталог, обозначаемый знаком косой черты «/». Чтобы однозначно идентифицировать любой файл, можно указать путь к этому файлу от корневого или текущего каталога. Все элементы пути отделяются друг от друга символом косой черты. Если первый символ строки также косая черта, то путь берет начало в корневом каталоге, в противном случае – в текущем. Путь с единственным именем обозначает файл в текущем каталоге.

Примеры:

- docs.ps – файл с именем docs.ps в текущем каталоге;
- /usr/doc/FAQ/README – файл с именем README в каталоге /usr/doc/FAQ;
- work/thesis.tex – файл thesis.tex в подкаталоге work текущего каталога.

Понятие текущего каталога несколько отличается от такового в системе MS-DOS или Windows. В UNIX у каждого процесса собственный текущий каталог. Корневой каталог файлового дерева UNIX обычно содержит следующие подкаталоги (в разных системах эта структура может отличаться)

- /bin – минимальный набор исполняемых файлов, необходимый для работоспособности системы;
- /etc – файлы конфигурации системы;
- /dev – файлы устройств;
- /home – домашние каталоги пользователей;
- /lib – основные системные библиотеки и модули;
- /root – каталог администратора системы root;
- /proc – файлы-образы выполняющихся процессов;
- /sbin – минимальный набор утилит администратора;
- /tmp – каталог для временных файлов;
- /usr – основной объем файлов системы: установленные программы, библиотеки, исходные тексты ядра, файлы данных и прочее;
- /var – каталог для изменяющейся информации (учетных данных, почтовых ящиков, очередей принтера, отформатированных страниц документации, логов и др.).

Следует отметить, что символ косой черты не является частью имен каталогов, а лишь указывает, что данные элементы находятся в корневом каталоге. В каждом каталоге также существует два особых «подкаталога» с именами «две точки» и «точка». Первый из них служит указателем на однозначно определенный родительский каталог (вышестоящий), а второй – на данный текущий каталог. Например, путь «../readme» указывает на файл «readme», который находится в родительском каталоге (на ступень выше), а путь «./readme.now» укажет на файл «readme.now», который находится в текущем каталоге.

Большая часть файлового дерева UNIX обычно сосредоточена в каталоге /usr. Как правило, там можно найти следующие подкаталоги:

/usr/bin – исполняемые файлы;

/usr/doc – документация в различных форматах;

/usr/etc – файлы конфигурации программного обеспечения, установленного дополнительно;

/usr/include – включаемые файлы для программ, например на языке C;

/usr/lib – разделяемые библиотеки;

/usr/local – локальное программное обеспечение, файлы данных и библиотеки (этот каталог в некоторых системах может не использоваться);

/usr/sbin – утилиты администратора;

/usr/share – данные, совместно используемые различными прикладными программами;

/usr/src – исходные тексты различных компонент системы, включая ядро.

Формирование имен файлов

В связи с тем, что зачастую для одного языка существует несколько кодировок (например, для русского языка существуют следующие кодировки: CP866, CP1251, KOI-8R и т. д., хотя в последнее время с распространением UTF8 ситуация постепенно улучшается), то рекомендуется, чтобы имя файла или каталога составлялось из следующих символов:

- прописные и строчные латинские буквы;
- цифры;
- символ подчеркивания;
- символ точки;
- знак минуса (не должен быть первым символом имени);
- знак плюса (использовать не рекомендуется).

В каждой конкретной ОС в именах файлов могут быть допустимы и другие символы, но их использование может привести к некорректности работы некоторых программ и, кроме того, может затруднить перенос файлов между разными ОС. Не рекомендуется использовать названия файлов из локальных таблиц кодировок (например, имена файлов на русском языке), т. к. очень часто для одного языка существует несколько кодировок. Максимальная длина имени файла варьируется в разных системах и зависит скорее от используемой файловой системы, чем от самой ОС. Обычно можно использовать достаточно длинные имена файлов (до 255 символов). Максимальный размер файла в файловой системе также зависит от ее типа. Для современных файловых систем размер файла более 4 Гбайт не является проблемой.

Как отмечалось выше, прописные и строчные буквы в системе UNIX различаются, т.е. имена «filename», «FILENAME» и «FileName» являются разными. При этом файлы, отличающиеся только регистром букв, могут находиться в одном каталоге.

В отличие от системы MS-DOS, знак точки является обычным символом, допустимым в любом месте имени файла, а такого понятия, как расширение имени файла, строго говоря, в системе UNIX нет. Тем не менее, последние части имен файлов, отделенные от остальной части имени точками, часто указывают на тип файла. В качестве примера имя файла «myphoto.tiff.gz» может означать, что файл представляет собой изображение в формате TIFF, сжатое программой сжатия gzip.

Точка, являющаяся первым символом имени файла или каталога, имеет особое значение: такие имена по умолчанию не выводятся в листинге содержимого каталогов (хотя к ним можно свободно обращаться), для получения полного списка файлов вместо ls, нужно ввести ls -a. Другими словами, чтобы сделать файл «скрытым», нужно начать его имя с точки. Этим часто пользуются для именования служебных файлов, на которые не имеет смысла обращать особое внимание.

Просмотр и интерпретация прав доступа к файлам

ОС семейства UNIX – традиционно многопользовательские системы. Как уж е отмечалось ранее, чтобы начать работать, пользователь должен «войти» в систему, введя

со свободного терминала свое регистрационное имя и пароль. Человек, зарегистрированный в учетных файлах системы и, следовательно, имеющий учетное имя, называется зарегистрированным пользователем системы. Регистрацию новых пользователей обычно выполняет администратор системы. Основными минимальными данными, требуемыми для регистрации пользователя в системе, являются:

- имя пользователя;
- название группы, к которой относится пользователь;
- пароль.

В UNIX базовые права доступа к файлам включают три составляющие:

- разрешение чтения (обозначается буквой «r», от слова Read);
- разрешение записи (буква «w», от слова Write);
- разрешение выполнения (буква «x», от слова eXecute).

Разрешение на чтение позволяет пользователю читать содержимое файлов, а в случае каталогов – просматривать перечень имен файлов в каталоге (используя, например, команду ls).

Разрешение на запись позволяет пользователю писать в файл, т. е. изменять его содержимое. Для каталогов это дает право создавать в каталоге новые файлы и каталоги или удалять файлы в этом каталоге.

Наконец, разрешение на выполнение позволяет пользователю запускать файлы на исполнение (как программы в машинном коде, так и командные файлы). Если на файле стоит атрибут выполнения, то независимо от его имени он считается программой, которую можно запустить (в отличие от DOS или Windows, в UNIX возможность исполнения файла не зависит от «расширения» имени файла, такого как .exe). Разрешение на выполнение применительно к каталогам означает возможность перехода в этот каталог (например, командой cd). Поэтому для каталогов право выполнения часто называют правом поиска. Отметим, что для каталогов биты чтения и выполнения (r и x) чаще всего используются в паре, т. е. либо присутствуют оба, либо отсутствуют.

В атрибутах доступа к файлам, перечисленные типы прав доступа могут быть предоставлены для трех классов пользователей:

- владельца (у каждого файла в UNIX есть один владелец);
- группы (с каждым файлом связана группа пользователей этого файла);
- всех остальных пользователей.

Набор прав доступа для конкретных файлов можно просмотреть с помощью команды ls -l. Например:

```
:~$ ls -l tmp/
drwxrwxr-x 10 john users 1024 Aug 30 2002 newdir
-rw-r----- 1 john users 173727 Jan 13 23:48 archive-0113.zip
```

В этом примере видно, что владельцем файлов является пользователь john, а группой владельцев является группа users. Набор букв и прочерков в левой части определяет тип файла (первый символ) и права доступа к файлу (остальные девять символов). В приведенном примере первая запись относится к каталогу (первая буква d) и демонстрирует права доступа rwxrwxr-x.

Вторая запись относится к обычному файлу (прочерк на месте первого символа) и показывает права rw-r-----. Девять символов прав доступа определяют возможность чтения (r), записи (w) и выполнения (x) для владельца файла (первые три символа), группы владельца (следующие три символа) и всех остальных (последние три символа). Прочерки означают отсутствие соответствующих прав.

Следовательно, в приведенном примере john и все пользователи группы users могут просматривать и изменять содержимое каталога newdir, а также переходить в него, а остальные пользователи могут читать и переходить в этот каталог, но не могут создавать или удалять в нем новые файлы; john может читать и изменять файл archive-0113.zip, пользователи группы users могут только читать содержимое этого файла, а все остальные не имеют к нему никаких прав доступа.

Кроме символьного представления прав доступа часто используется цифровая форма. В цифровом представлении права доступа составляются из трех восьмеричных цифр, каждая из которых определяет набор из трех битов полномочий rwx. Чтобы перевести права доступа из символьного представления в числовое, следует:

- представить набор прав в двоичном виде (например, 110100000 для набора прав rw-r-----);
- перевести полученное двоичное число в восьмеричную систему счисления (например, восьмеричным представлением двоичного числа 110100000 будет 640).

Права доступа так же можно представить в числовой форме путем суммирования восьмеричных значений отдельных битов прав доступа

400 – владелец имеет право на чтение;
200 – владелец имеет право на запись;
100 – владелец имеет право на выполнение;
040 – группа имеет право на чтение;
020 – группа имеет право на запись;
010 – группа имеет право на выполнение;
004 – остальные имеют право на чтение;
002 – остальные имеют право на запись;
001 – остальные имеют право на выполнение.
Можно заметить, что для прав доступа rw-r----- получим: 400 + 200 + 040 = 640.

Изменение прав доступа к файлам

Изменить права доступа к файлу может либо его владелец, либо привилегированный пользователь (root). Делается это командой chmod (change mode). Существует два формата использования этой команды: с использованием символьного и числового представления прав доступа. Использование числового представления позволяет одной командой изменить полный набор прав доступа, например:

```
chmod 770 newdir
```

Данная команда установит права доступа в числовое значение 770, т. е. rwxrwx---, что даст полные права владельцу и группе владельца, и никаких прав всем остальным.

Использование символьного представления прав доступа в команде chmod может показаться несколько сложнее, но позволяет манипулировать отдельными битами прав доступа. Например, чтобы снять бит записи для группы владельца каталога newdir, достаточно ввести:

```
:~$ chmod g-w newdir
```

Условный синтаксис этой команды таков

```
chmod {u,g,o,a}{+,-,=}{r,w,x} файлы ...
```

В качестве аргументов команда принимает указание классов пользователей

- «u» – владелец-пользователь (user),
- «g» – владелец-группа (group),
- «o» – остальные пользователи (others),
- «a» – все вышеперечисленные группы вместе (all).

Операцию, которую необходимо произвести с правами доступа:

- «+» – добавить,
- «-» – убрать,
- «=» – присвоить.

Права доступа («r», «w», «x») назначаемы каталогам и файлам. В chmod может использоваться ключ R, позволяющий рекурсивно обрабатывать содержимое подкаталогов.

Циклы и условия в командной строке

While

```
while  <УСЛОВИЕ>; do <КОМАНДА>; done
while true; do ls; sleep 2; done
```

For

```
for n in {1..5}; do <КОМАНДА>; done
for ((i=1; i<5; ++i)); do <КОМАНДА>; done
for i in 1 2 3; do mkdir ${i}; done
```

If

```
if [ <УСЛОВИЕ> ]; then <КОМАНДА>; fi
if [ "${asd}" = "true" ]; then echo "hellow world"; fi
```