Введение в bash. Работа с командной строкой

Потоки ввода и вывода програм. Перенаправление потоков. Конвейеры.
Команды grep, sort, uniq, cut, tr

3 октября 2025 г.

Иван Ильницкий
ФББ МГУ

Команда grep: примеры с опциями

- Команда `grep` используется для поиска строк, соответствующих заданному шаблону, в текстовых файлах или стандартном вводе. Она полезна для фильтрации данных и быстрого нахождения нужной информации.
- Поиск с использованием нескольких файлов

  ```
  grep -r fox path/to/dir
  ```

- Флаги -A и -B позволяют выводить строки до и после совпадения

  ```
  grep -A 2 -B 2 pattern example.txt
  ```

- Поиск по всему слову

  ```
  grep -w a example.txt
  ```

- Вывод только тех частей строк, которые соответствуют заданному шаблону

  ```
  grep -o fox example.txt
  ```

Команда grep

Регулярные выражения и специальные символы

- Поиск строк, начинающихся с определенного слова

```
grep '^The' example.txt
```

- Поиск строк, заканчивающихся на определенное слово

```
grep 'dog$' example.txt
```

- Поиск строк, содержащих цифры

```
grep '[0-9]' example.txt
grep '[A-Z]' example.txt
```

Команда grep (более сложные примеры)

Регулярные выражения и специальные символы

- Поиск слов, начинающихся или заканчивающихся на определенные символы

```
grep -e '\bte' example.txt
grep -e 'st\b' example.txt
```

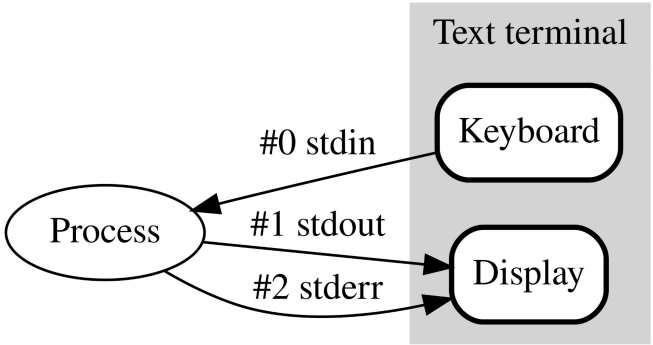
- `grep -e 'cat' -e 'dog' file.txt` найдет строки, содержащие либо "cat", либо "dog".

Потоки ввода и вывода

До сих пор мы печатали что-то на клавиатуре и видели (или не видели) результат в командной строке

Ввод и вывод распределяется между тремя стандартными потоками:

- Стандартный ввод - **stdin** - 0 - то, что вводят с клавиатуры
- Стандартный вывод - **stdout** - 1 - то, что выводится на экран
- Стандартный вывод ошибок - **stderr** - 2 - вывод ошибки на экран stdin только считывание данных



Потоки ввода и вывода

Вывод данных на экран и чтение их с клавиатуры происходит потому, что по умолчанию стандартные потоки ассоциированы с терминалом пользователя.

Это не является обязательным — потоки можно подключать к чему угодно — к файлам, программам. В командном интерпретаторе bash такая операция называется *перенаправлением*.

Потоки ввода/вывода на примере `cat`

Мы уже пользовались утилитой `cat` для просмотра содержимого файлов.

- `cat` считывает данные из стандартного потока ввода и передает в стандартный поток вывода.
- На вход можно подавать сразу несколько файлов для последующей записи их в один общий файл.

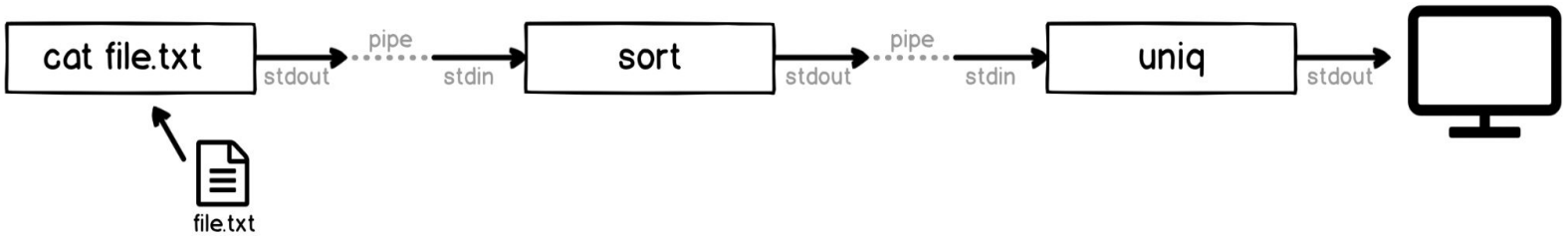
Перенаправление потоков

- Направление стандартного потока вывод в файл:
  - ✔ Перезаписывание файла:
    `> file`
  - ✔ Дописывание в конец файла:
    `>> file`
- Направление стандартного потока ошибок в файл:
  `2> file` или `2>> file`
- Использование файла как источник для стандартного потока ввода:
  `< file`
- При желании объединить потоки вывода и ошибок, т.е. чтобы вся информация была записана в один файл используйте конструкцию
  `&> file`

Перенаправляя результат выдачи любых команд в файл, вы можете создавать файлы, а затем анализировать их

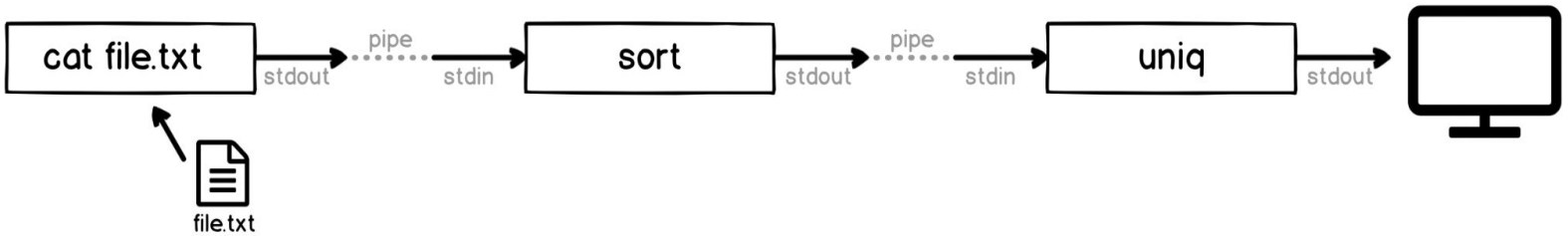
Конвейеры

- Конвейер (pipeline) — это последовательность команд, соединённых символом | , где вывод одной команды становится вводом для следующей.
- Преимущества использования конвейеров в том, что можно легко комбинировать различные программы для достижения нужного результата и в повышении эффективности, нет необходимости создавать временные файлы.



Конвейеры

- Конвейер (pipeline) — это последовательность команд, соединённых символом `|`, где вывод одной команды становится вводом для следующей.
- Преимущества использования конвейеров в том, что можно легко комбинировать различные программы для достижения нужного результата и что повышается эффективность, нет необходимости создавать временные файлы.




```
cat file.txt | sort | uniq
```

Команда echo

- Предназначена для вывода аргументов, переданных ей в командной строке. Она принимает аргументы и выводит их в терминал - на стандартный поток вывода (stdout).

- Если мы хотим напечатать текст, разделив его на 2 строки, то нужно воспользоваться символом переноса строки: \n

```
echo -e 'First line\nSecond line'
```

Команда sort

- Команда `sort` используется для сортировки строк в текстовых файлах или выводе команд. Она может сортировать данные по различным критериям, таким как числовое значение, алфавитный порядок, дата и т.д.
- Основные опции:

`-n` – Сортирует строки как числа. Полезно для сортировки по числовым значениям.

`-r` – Сортирует строки в обратном порядке (по убыванию).

`-k` – Позволяет указать ключ (колонку) для сортировки. Например, `-k2` сортирует по второй колонке.

`-t` – Устанавливает разделитель полей. Например, `-t','` для CSV файлов.

`-u` – Удаляет дубликаты строк, оставляя только уникальные.

`-V` - используется для сортировки строк в "естественном" порядке, который учитывает числовые значения в строках

Команда sort

`sort -k1`
- Сортирует строки, начиная с первого поля и до конца строки. Это означает, что сортировка будет учитывать все последующие поля после первого, если они есть.

`sort -k1,1`
- Сортирует строки только по первому полю. Здесь 1,1 указывает, что сортировка ограничивается только первым полем, и последующие поля не будут учитываться.

`sort -k1,2`
- Сортирует строки начиная с первого поля и заканчивая вторым полем. Это означает, что сортировка будет учитывать первое и второе поля вместе как единое целое.

`sort -k1,1 -k2,2n`
- Сортирует строки сначала по первому полю (-k1,1), а затем по второму полю в числовом порядке (-k2,2n). Здесь -k2,2n указывает, что второе поле будет отсортировано как число, что полезно, если второе поле содержит числовые значения.

Команда uniq

- Команда `uniq` используется для фильтрации повторяющихся строк в текстовых файлах. Она обычно применяется после команды `sort`, поскольку сам uniq сортировать не умеет и удаляет только дублирующиеся строки, идущие друг за другом.
- Основные опции:

  `-c` – Подсчитывает количество повторений каждой строки и выводит это число перед строкой.

  `-d` – Показывает только те строки, которые повторяются в файле.

  `-u` – Выводит только уникальные строки, которые не имеют повторений.

Команда cut

- Команда `cut` используется для извлечения определенных полей, столбцов или символов из строк текста в файле или потоке данных
- Основные опции:

  `-f` – Извлекает указанные поля. Поля разделяются символом, заданным с помощью опции -d. Например, `-f 1,3` извлечет первое и третье поле.

  `-d` – Определяет символ-разделитель полей. По умолчанию используется табуляция. Например, `-d ','` для CSV-файлов.

  `-c` – Извлекает указанные символы из каждой строки. Например, `-c 1-5` извлечет первые пять символов.

Команда tr

- Команда `tr` используется для преобразования или удаления символов из входного потока.
- Использование:

  `tr [опции] SET1 [SET2]`

  Без опций заменяет символы из SET1 на символы из SET2
- Основные опции:

  `-d` – удалить символы, указанные в SET1.

- Примеры:

  `tr ',' '\t'` – заменит все запятые на символ табуляции

  `tr 'ab' 'xy'` – заменит каждый a на x и каждый b на y.

Работа с набором команд

- Если нужно последовательно выполнить несколько команд, то непосредственная работа с командной строкой становится неудобной
- Можно записать все необходимые команды в командную строку, разделив их «;»
- В таком случае редактирование команд будет довольно затруднительным

Пример:

```
touch empty_file.txt; ls -a -l -h
```

Командные сценарии

- До этого мы вводили команды непосредственно в командную строку
- У bash есть неинтерактивный режим работы, в котором он исполняет команды, прочитанные из текстового файла. Такие файлы с набором команд называют *сценариями* (или, в разговорном стиле, *скриптами*). Чтобы запустить скрипт на исполнение нужно передать его команде bash в качестве аргумента.

  ```
  bash script.sh
  ```

- Все строки файла bash будет интерпретировать в качестве команд. Поэтому в файле не должно быть постороннего текста! Допускаются только пустые строки для визуального разделения команд.

Загрузка файлов из интернета: *wget*

wget:
Основная ориентированность на скачивание файлов.
Поддерживает рекурсивное скачивание (можно скачивать целые сайты с директориями).

- `wget [опции] [ссылка]`
- Опции: `-c` (продолжить загрузку), `-P` (указать директорию для скачивания), `-r` (рекурсивное скачивание), `-O` (задать имя скачанному файлу)
- Примеры:

```
wget http://example.com/file.txt
wget -r http://example.com/
wget -P directory http://example.com/
wget -O arch.zip http://example.com/archive2024.zip
```