Операционные системы

Огородников Юрий Юрьевич
yogorodnikov@gmail.com

UNIX::Использование двойных скобок

- [[...&&...]]

UNIX::Использование двойных скобок

- [[...&&...]]
- [[”abc” $>$ ”bcd”]] #сравнение строк

UNIX::Использование двойных скобок

- [[...&&...]]
- [[”abc” $>$ ”bcd”]] #сравнение строк
- [[”abc” $==$ $a*$]] #использование шаблонов

UNIX::Использование двойных скобок

- [[...&&...]]
- [["abc" > "bcd"]] #сравнение строк
- [["abc" == $a*$]] #использование шаблонов
- более подробно: man bash или help [[

UNIX::Разукрашивание вывода

- Не добавлять перевод строки в конце: ключ *-n* команды *echo*; включить поддержку вывода Escape последовательностей: ключ *-e*

UNIX::Разукрашивание вывода

- Не добавлять перевод строки в конце: ключ *-n* команды *echo*; включить поддержку вывода Escape последовательностей: ключ *-e*
- Вывод форматированного текста: echo -en "\e[МОДИФИКАТОРЫmtext\e[0m"

UNIX::Разукрашивание вывода

- Не добавлять перевод строки в конце: ключ *-n* команды *echo*; включить поддержку вывода Escape последовательностей: ключ *-e*
- Вывод форматированного текста: echo -en "\e[МОДИФИКАТОРЫmtext\e[0m"
- Что представляют из себя модификаторы:
  1-7 – тип текста (жирный, курсив, подчеркнутый, ...)
  21-27 – вернуть тип текста к типу по умолчанию
  30-37, 90-97 – цвет текста
  40-47, 100-107 – цвет фона

UNIX::Разукрашивание вывода

- Не добавлять перевод строки в конце: ключ *-n* команды *echo*; включить поддержку вывода Escape последовательностей: ключ *-e*
- Вывод форматированного текста: echo -en "$\backslash e[$МОДИФИКАТОРЫ$mtext\backslash e[0m$"
- Что представляют из себя модификаторы:
  1-7 – тип текста (жирный, курсив, подчеркнутый, ...)
  21-27 – вернуть тип текста к типу по умолчанию
  30-37, 90-97 – цвет текста
  40-47, 100-107 – цвет фона
- Комбинация модификаторов: через ;
  echo -en "$\backslash e[1;32;45mtext\backslash e[0m\backslash n$"

UNIX::Разукрашивание вывода

- Не добавлять перевод строки в конце: ключ *-n* команды *echo*; включить поддержку вывода Escape последовательностей: ключ *-e*
- Вывод форматированного текста: echo -en "\e[МОДИФИКАТОРЫmtext\e[0m"
- Что представляют из себя модификаторы:
  1-7 – тип текста (жирный, курсив, подчеркнутый, ...)
  21-27 – вернуть тип текста к типу по умолчанию
  30-37, 90-97 – цвет текста
  40-47, 100-107 – цвет фона
- Комбинация модификаторов: через ;
  echo -en "\e[1;32;45mtext\e[0m\n"
- Задание: вывести таблицу всевозможных сочетаний цветов текста и фона: каждая строка – свой цвет фона, каждый столбец – свой текст текста.

UNIX::printf

- Используется для вывода на экран строк в нужном формате:
  printf 'ФОРМАТ' аргументы

UNIX::printf

- Используется для вывода на экран строк в нужном формате: printf 'ФОРМАТ' аргументы
- printf '%s' 'hello!'

UNIX::printf

- Используется для вывода на экран строк в нужном формате:
  printf 'ФОРМАТ' аргументы
- printf '%s' 'hello!'
- printf $'%s\n'$ 'hello!'

UNIX::printf

- Используется для вывода на экран строк в нужном формате:
  printf 'ФОРМАТ' аргументы
- printf '%s' 'hello!'
- printf $'%s\n'$ 'hello!'
- Форматы:
  %b Рассматривает аргумент как строку, при этом интерпретирует все управляющие символы, содержащиеся в ней.
  %s Рассматривает аргумент как простую строку.
  %c Рассматривает аргумент как символ, при этом берется первый символ выражения или строки.
  %d Представляет аргумент в виде десятичного числа, могущего иметь знак (+ или -).
  %u Представляет аргумент в виде десятичного числа, не имеющего знака.
  %o Представляет аргумент в виде не имеющего знака восьмеричного числа.
  %x Представляет аргумент в виде не имеющего знака шестнадцатеричного числа. Буквы пишутся в нижнем регистре.
  %f Интерпретирует аргумент как число с плавающей запятой.

UNIX::printf

- Используется для вывода на экран строк в нужном формате:
  printf 'ФОРМАТ' аргументы
- printf '%s' 'hello!'
- printf $'%s\n'$ 'hello!'
- Форматы:
  %b Рассматривает аргумент как строку, при этом интерпретирует все управляющие символы, содержащиеся в ней.
  %s Рассматривает аргумент как простую строку.
  %c Рассматривает аргумент как символ, при этом берется первый символ выражения или строки.
  %d Представляет аргумент в виде десятичного числа, могущего иметь знак (+ или -).
  %u Представляет аргумент в виде десятичного числа, не имеющего знака.
  %o Представляет аргумент в виде не имеющего знака восьмеричного числа.
  %x Представляет аргумент в виде не имеющего знака шестнадцатеричного числа. Буквы пишутся в нижнем регистре.
  %f Интерпретирует аргумент как число с плавающей запятой.
- printf $'%20s\n'$ 'Stroka latinitzej'# Если строка короче 20 символов, то ее нужно дополнить слева пробелами
- printf "%.20f\n"4, 3# дополнение строки нулями

UNIX::printf

- Используется для вывода на экран строк в нужном формате:
  printf 'ФОРМАТ' аргументы
- printf '%s' 'hello!'
- printf '%s\n' 'hello!'
- Форматы:
  %b Рассматривает аргумент как строку, при этом интерпретирует все управляющие символы, содержащиеся в ней.
  %s Рассматривает аргумент как простую строку.
  %c Рассматривает аргумент как символ, при этом берется первый символ выражения или строки.
  %d Представляет аргумент в виде десятичного числа, могущего иметь знак (+ или -).
  %u Представляет аргумент в виде десятичного числа, не имеющего знака.
  %o Представляет аргумент в виде не имеющего знака восьмеричного числа.
  %x Представляет аргумент в виде не имеющего знака шестнадцатеричного числа. Буквы пишутся в нижнем регистре.
  %f Интерпретирует аргумент как число с плавающей запятой.
- printf '%20s\n' 'Stroka latinitzej'# Если строка короче 20 символов, то ее нужно дополнить слева пробелами
- printf "%.20f\n"4, 3# дополнение строки нулями
- Задание. Пусть есть список студентов и их средняя оценка. Цель: вывести этот список так, чтобы имена студентов были друг под другом, а средние оценки были всегда округлены до 1 знака после запятой и располагались друг под другом. Отобразить и заголовки столбцов.

UNIX::awk

- *awk* — команда для поиска и преобразования текста

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}' # комбинация текстовой строки и колонки из подходящих строк

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}' # комбинация текстовой строки и колонки из подходящих строк
- ls -la | awk '/os/ {print("hello, "$3)} /c#/ {print("Bye, "$0)}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}' # комбинация текстовой строки и колонки из подходящих строк
- ls -la | awk '/os/ {print("hello, "$3)} /c#/ {print("Bye, "$0)}' # по-разному обрабатываем строки, содержащие "os"и "c#"

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}' # комбинация текстовой строки и колонки из подходящих строк
- ls -la | awk '/os/ {print("hello, "$3)} /c#/ {print("Bye, "$0)}' # по-разному обрабатываем строки, содержащие "os"и "c#"
- По умолчанию колонки разделяются пробелами и знаками табуляции. Чтобы заменить разделитель на свой, есть ключ -F

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}' # комбинация текстовой строки и колонки из подходящих строк
- ls -la | awk '/os/ {print("hello, "$3)} /c#/ {print("Bye, "$0)}' # по-разному обрабатываем строки, содержащие "os"и "c#"
- По умолчанию колонки разделяются пробелами и знаками табуляции. Чтобы заменить разделитель на свой, есть ключ -F
- Специальные переменные в awk:
  NR - номер текущей строки
  NF - число полей в текущей строке

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}' # комбинация текстовой строки и колонки из подходящих строк
- ls -la | awk '/os/ {print("hello, "$3)} /c#/ {print("Bye, "$0)}' # по-разному обрабатываем строки, содержащие "os"и "c#"
- По умолчанию колонки разделяются пробелами и знаками табуляции. Чтобы заменить разделитель на свой, есть ключ -F
- Специальные переменные в awk:
  NR - номер текущей строки
  NF - число полей в текущей строке
- ls -la | awk '/os/ {print("hello, "$3"from line "NR)}'

UNIX::awk

- *awk* — команда для поиска и преобразования текста
- awk-программа состоит из правил вида:
  шаблон {действия}
  действия разделяются или переводами строк
- ls -la | awk '{print}' # выведет все строки
- ls -la | awk '/os/ {print}' #выведет все строки, содержащие os
- ls -la | awk '/os/ {print $0}' # то же самое
- ls -la | awk '/os/ {print $1}' #выведет первые "слова в строках, содержащих os
- ls -la | awk '/os/ {print "Hello!"}' #выведется Hello! столько раз, сколько строк, содержащих os
- ls -la | awk '/os/ {print "(Hello, "$3)}' # комбинация текстовой строки и колонки из подходящих строк
- ls -la | awk '/os/ {print("hello, "$3)} /c#/ {print("Bye, "$0)}' # по-разному обрабатываем строки, содержащие "os"и "c#"
- По умолчанию колонки разделяются пробелами и знаками табуляции. Чтобы заменить разделитель на свой, есть ключ -F
- Специальные переменные в awk:
  NR - номер текущей строки
  NF - число полей в текущей строке
- ls -la | awk '/os/ {print("hello, "$3"from line "NR)}'
- Программа awk может иметь следующий вид:
  BEGIN {действие}
  шаблон {действие}
  шаблон {действие}
  ...
  END {действие}

UNIX::awk

- Посчитать сумму размеров файлов:
  ls -la | awk 'BEGIN {res=0}\
  {res+=$5}\
  END {print(res)}'

UNIX::awk

- Посчитать сумму размеров файлов:
  ls -la | awk 'BEGIN {res=0}\
  {res+=$5}\
  END {print(res)}'
- Этот же пример:
  ls -la | awk '{res+=$5}\
  END {print(res)}'

UNIX::awk

- Посчитать сумму размеров файлов:
  ls -la | awk 'BEGIN {res=0}\
  {res+=$5}\
  END {print(res)}'
- Этот же пример:
  ls -la | awk '{res+=$5}\
  END {print(res)}'
- Можно работать только с избранными строками:
  ls -la | awk '/Sep/ {res+=$5}\
  END {print(res)}'

UNIX::awk

- Посчитать сумму размеров файлов:
  ls -la | awk 'BEGIN {res=0}\
  {res+=$5}\
  END {print(res)}'
- Этот же пример:
  ls -la | awk '{res+=$5}\
  END {print(res)}'
- Можно работать только с избранными строками:
  ls -la | awk '/Sep/ {res+=$5}\
  END {print(res)}'
- Допустимые операции:
  Арифметические: =, +=, -=, *=, /=, %=, +, /, %, ++, –
  Сравнения чисел/строк: <, <=, ==, !=, >=, >
  Логические операции: !, ||, &&
  А также: конкатенация

UNIX::awk

- Посчитать сумму размеров файлов:
  ls -la | awk 'BEGIN {res=0}\
  {res+=$5}\
  END {print(res)}'
- Этот же пример:
  ls -la | awk '{res+=$5}\
  END {print(res)}'
- Можно работать только с избранными строками:
  ls -la | awk '/Sep/ {res+=$5}\
  END {print(res)}'
- Допустимые операции:
  Арифметические: =, +=, -=, *=, /=, %=, +, /, %, ++, –
  Сравнения чисел/строк: <, <=, ==, !=, >=, >
  Логические операции: !, ||, &&
  А также: конкатенация
- Более сложный пример:
  ls -la | awk '$5<1000 && $6=="*Sep*"'

UNIX::awk

- Посчитать сумму размеров файлов:
  ls -la | awk 'BEGIN {res=0}\
  {res+=$5}\
  END {print(res)}'
- Этот же пример:
  ls -la | awk '{res+=$5}\
  END {print(res)}'
- Можно работать только с избранными строками:
  ls -la | awk '/Sep/ {res+=$5}\
  END {print(res)}'
- Допустимые операции:
  Арифметические: =, +=, -=, *=, /=, %=, +, /, %, ++, –
  Сравнения чисел/строк: <, <=, ==, !=, >=, >
  Логические операции: !, ||, &&
  А также: конкатенация
- Более сложный пример:
  ls -la | awk '$5<1000 && $6=="*Sep*"'
- Если команд становится много, то можно все их написать во внешнем файле
  ls -la | awk -f 'commands.awk'
  Внутри commands.awk можно использовать for, while, if:
  if (условие) операторы [else операторы]
  while (условие) операторы
  for (выражение; условие; выражение) операторы

UNIX::awk

Задания

- Вывести среднее значение 5го столбца в выводе ls -la.
- Вывести только четные строки в выводе ls -la.
- Есть файл с успеваемостью студентов в следующем формате:
  Имя1 оценка1 оценка2 оценка3
  Имя2 оценка4 оценка5 оценка6
  Цель: подсчитать среднюю оценку для каждого студента и вывести в конце каждой строки: ... посчитать средний балл по всем студентам и вывести после таблицы. Нужно не забыть, что количество оценок у студентов разное