CLOUD
УЧЕБНЫЙ ЦЕНТР

Учебный центр АйТи Клауд

СОВРЕМЕННЫЕ ТЕХНОЛОГИИ ПРОСТЫМ ЯЗЫКОМ

Преподаватель _____

АйТи Клауд –

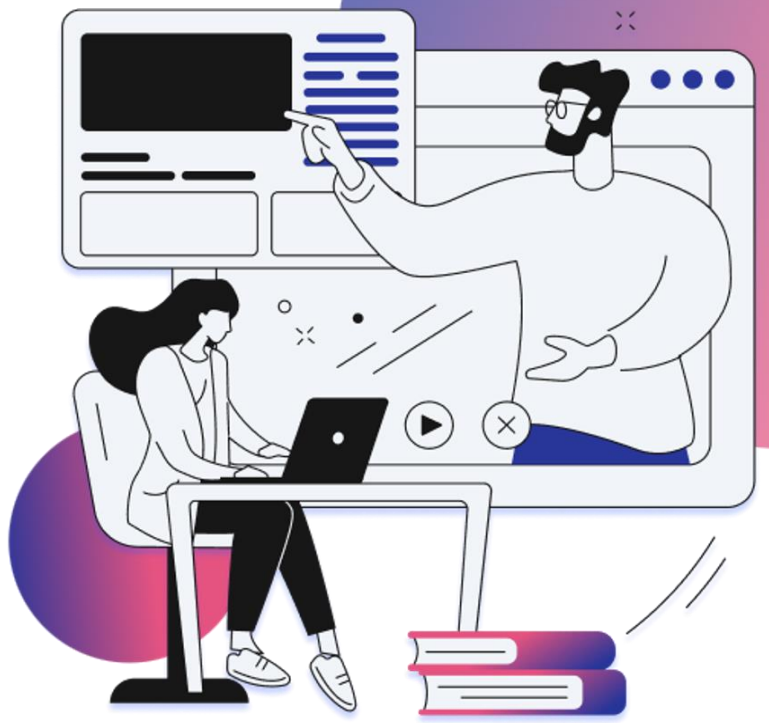
один из ведущих Учебных центров России в сфере ИТ-образования с 1996 года

преподаватель

Груздев Павел

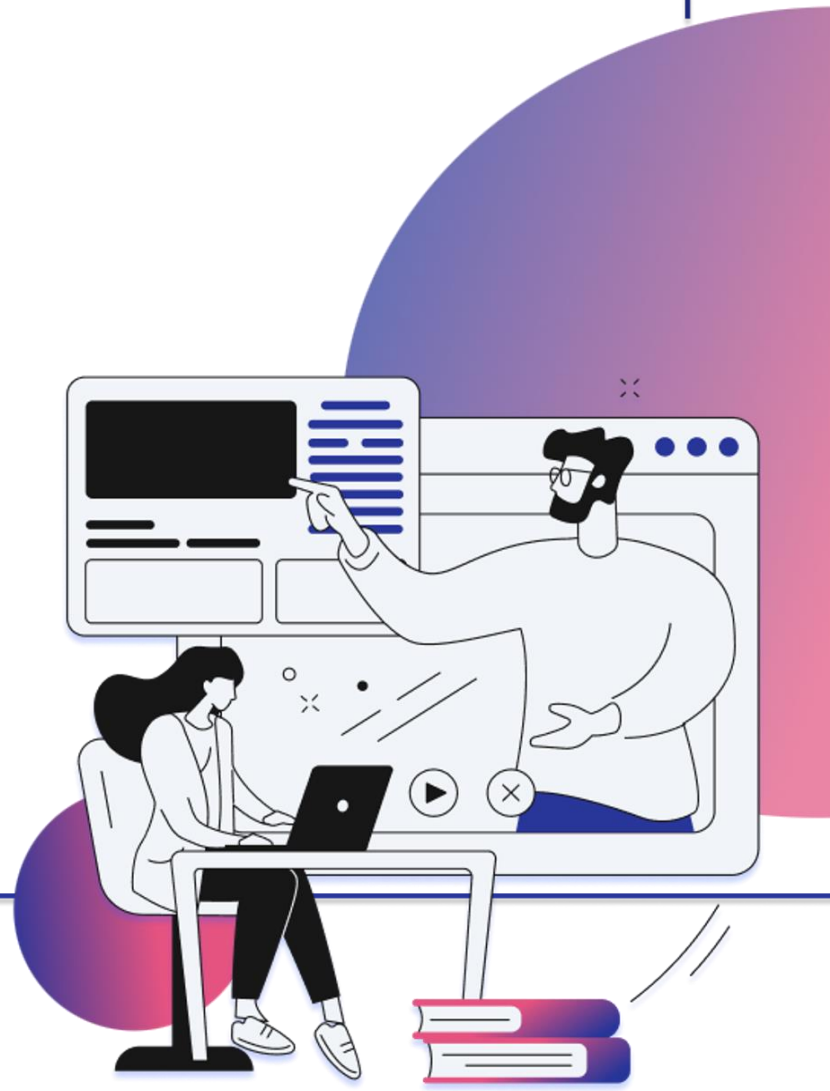
Темы и направления деятельности:

- Администрирование Linux систем
- Внедрение и эксплуатация Zabbix



Учебный центр АйТи Клауд

Разработка скриптов BASH для администраторов РЕД ОС

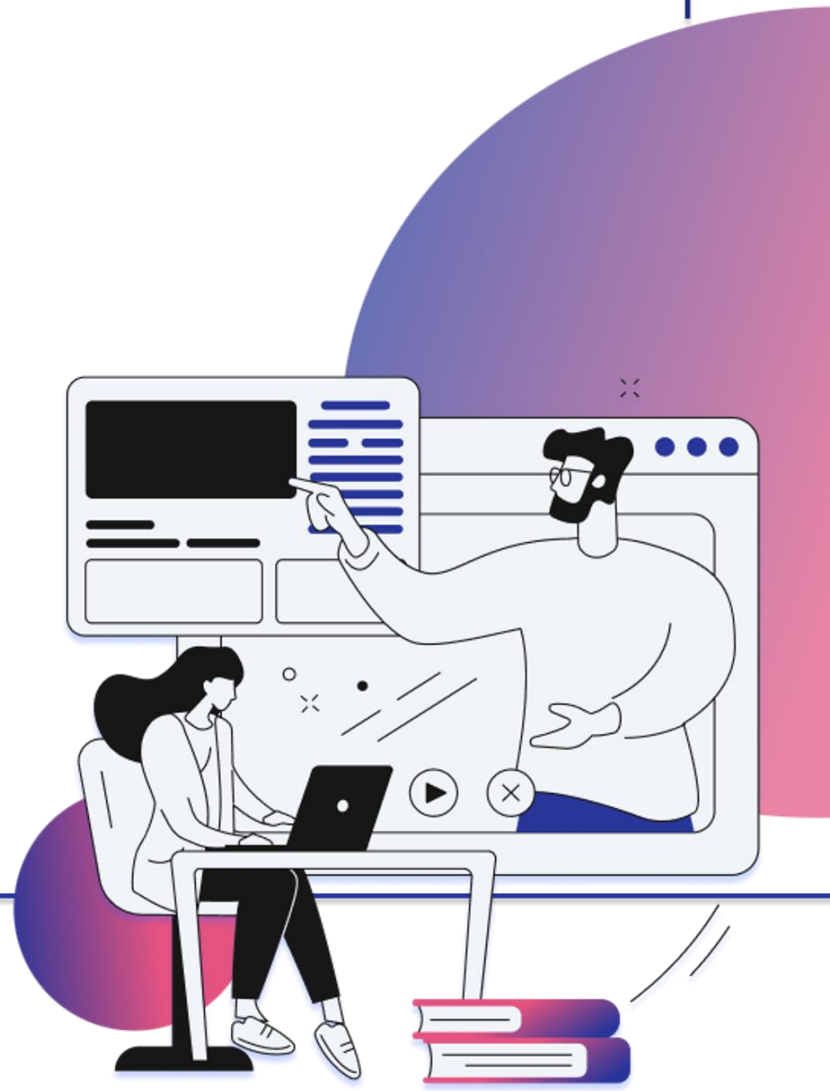


современные технологии простым языком

Разработка скриптов BASH. План курса

- Текстовые файлы и потоки
- Регулярные выражения
- Написание сценариев BASH

Учебный центр АйТи Клауд



Модуль 1. Текстовые файлы и потоки

современные технологии простым языком

Модуль 1. Текстовые файлы и потоки. План

- Перенаправление потоков ввода-вывода.
- Конвейеры и фильтры.
- Команда echo.
- Получение даты – date.
- Просмотр файлов с помощью more и less.
- Объединение файлов с помощью cat.
- Команды head и tail.
- Вырезание текста с помощью cut.
- Потоковый редактор sed.
- Потоковый редактор awk.
- Сравнение файлов и каталогов.
- Сортировка строк и вывод неповторяющихся строк.
- Подсчет количества и нумерация строк.
- Замена символов в строках с помощью команды tr.
- Команда xargs.

Перенаправление потоков ввода-вывода

- Три стандартных потока ввода-вывода
  - stdin – 0
  - stdout – 1
  - stderr – 2

Перенаправление потоков ввода-вывода

- Перенаправление файлов:
  - `< файл` **или** `0< файл`
  - `> файл` **или** `1> файл`
  - `2> файл`

Перенаправление потоков ввода-вывода

- Перенаправление в один файл:
  - `>  имя_файла 2>&1`
  - `2> имя_файла 1>&2`
  - `>& имя_файла`
  - `&> имя_файла`

Перенаправление потоков ввода-вывода

- noclobber
  - >|
  - 2>|
- set +o noclobber
- set -o noclobber

Перенаправление потоков ввода-вывода

- Дописывание
  - `>>`
  - `2>>`
- Here document
  - `<< teg`

Конвейеры и фильтры

- Конвейер (pipe) связывает stdin и stdout двух процессов
- `tee` – расщепляет поток
- Фильтр – команда принимающая поток ввода, преобразующая его, и выводящая в поток вывода

Команда echo

- `echo` – выводит строку в stdout, заданную в качестве аргумента.
  - `-n` – подавить вывод перевода строки
  - `-e` – интерпретрировать восьмеричные коды символов
  - `echo -ne '\033c'` – сброс терминала

Получение даты – date

- `date` – выводит строку даты в заданном формате.
  - `-d` – показать указанную дату
  - `+` – формат даты

Задание

- Перенаправление потоков
- Дата
- Конвейеры и фильтры
- `echo`

Просмотр файлов с помощью more и less

- more и less – пейджеры

Объединение файлов с помощью cat

- `cat` – сокращение от catenate (сцеплять)
- По умолчанию использует стандартные потоки
- Опции позволяют выводить непечатаемые символы в виде специальных кодов
- `tac` – выводит строки в обратной последовательности

Команды head и tail

- `head` - выводит первые строки
- `tail` – выводит последние строки
- `tail -f` – выводит вновь появляющиеся строки

Задание

- Работа с пейджерами
- Вывод текста

Вырезание текста с помощью cut

- `cut` выводит только указанные в командной строке символы, байты или поля из строк файла
  - `-c` – символы
  - `-b` – байты
  - `-f` – поля
  - `-d` – определить разделитель

Потоковый редактор sed

- Фильтр и потоковый (не интерактивный) редактор
- Не изменяет исходный поток (файл)
- По умолчанию ничего не меняет
- Удобен для работы с целыми строками
- Может использовать сценарии
- Общий синтаксис:

```
sed адрес_строки команды файл
```

Потоковый редактор sed

- Выбор строки для редактирования:
  - Номер
  - Диапазон
  - Регулярное выражение (шаблон)

Потоковый редактор sed

- Команды редактирования:
  - `d` – удаление
  - `p` – вывод строки (print)
  - `s` – замена
  - `i` – вставка строки
  - `a` – добавление строки
  - `c` – изменение всей строки на заданную строку
  - `q` – выход без дальнейшей обработки и вывода строк
  - `y` – команда транслитерации символ-в-символ
  - `=` - команда печати номера строки

Потоковый редактор awk

- Потоковый редактор текста
- Имеет специализированный язык программирования
- Удобен для обработки структурированных данных
- Может использовать сценарии
- Общий синтаксис:

```
awk 'условие{сценарий}' файл
```

Потоковый редактор awk

- Сценарий по умолчанию – печать строк
- Язык сценариев – аналог программы на языке C
- В сценариях можно использовать функции и переменные. Как встроенные так и свои

Потоковый редактор awk

- Обработка полей:
- `$0` – вся строка
- `$1` – первое поле, `$2` – второе, …
- Опция `-F` – задает разделитель полей

Потоковый редактор awk

- Форматированный вывод – функция `printf`
- Функция `print` – неформатированный вывод

Потоковый редактор awk

- Встроенные переменные:
  - `NR` – номер строки
  - `NF` – количество полей
  - `FS` – знак разделителя полей
  - `FILENAME` – имя файла
  - Весь список смотрите в `man`

Потоковый редактор awk

- Встроенные функции:
  - Числовые (sin, cos, sqrt, …)
  - Строковые (length, asort, substr, …)
  - Обработка времени (systime, mktime, strftime)
  - Манипуляции битами (and, or, compl, …)
  - Проверка типа (isarray)
  - Интернализации (bindtextdomain, dcgettext, dcngettext)

Задание

- Обработка текста sed и awk

Сравнение файлов

- Команда `diff` сравнивает содержимое двух текстовых файлов
- `diff` Может сравнивать каталоги
- Опции `-c` и `-u` позволяют получить патч
- Патч может быть использован для восстановления одного из файлов, командой `patch`
- Утилита `cmp` сравнивает бинарные файлы

Сортировка строк

- Команда `sort`
- По умолчанию сортирует по алфавиту
  - `-r` – обратный порядок
  - `-n` – числовая сортировка
  - `-k` – номер поля
  - `-M` – сортировка по месяцам
  - `-h` – сортировка по понятным человеку единицам измерения

Вывод неповторяющихся строк

- Команда `uniq`
  - `-c` – подсчет количества
  - `-d` – вывод дубликатов
  - `-u` – вывод уникальных строк

Объединение строк двух файлов по общему полю

- Команда `join`
  - `-t` – задает разделитель
  - `-j` – номера полей для объединения
  - `-v` – вывод непарных строк

Задание

- Работа с патчами
- Сортировка текстового потока
- Объединение строк

Подсчет количества и нумерация строк

- Команда `wc` – подсчет строк, слов и символов в потоке
- Команда `nl` – форматированная нумерация

Замена символов в строках с помощью команды tr

- Фильтр `tr` – изменяет, удаляет или удаляет повторы символов в потоке
- Использует концепцию набора символов [:…:]

Задание

- Подсчет и нумерация строк
- Замена символов в потоке

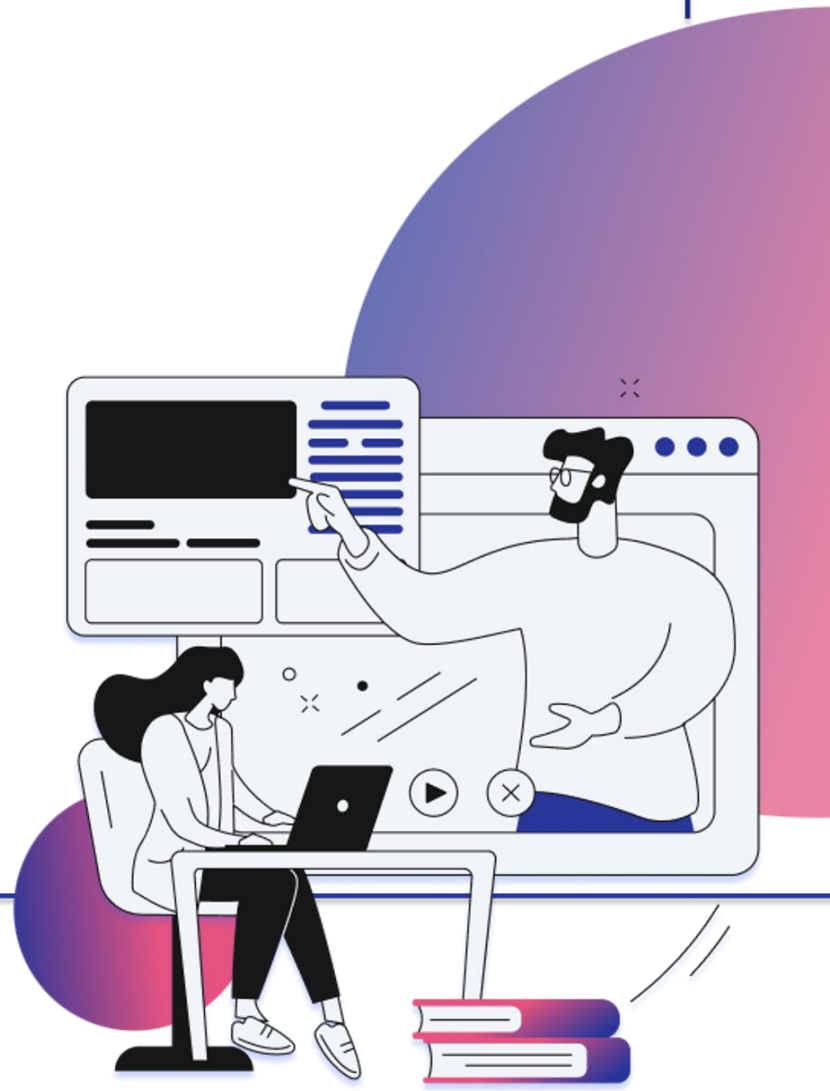
Команда xargs

- Команда `xargs` – формирует команды из стандартного потока ввода

Что изучили в данном модуле

- Перенаправление потоков ввода-вывода.
- Конвейеры и фильтры.
- Команда echo.
- Получение даты – date.
- Просмотр файлов с помощью more и less.
- Объединение файлов с помощью cat.
- Команды head и tail.
- Вырезание текста с помощью cut.
- Потоковый редактор sed.
- Потоковый редактор awk.
- Сравнение файлов и каталогов.
- Сортировка строк и вывод неповторяющихся строк.
- Подсчет количества и нумерация строк.
- Замена символов в строках с помощью команды tr.
- Команда xargs.

Учебный центр АйТи Клауд



Модуль 2. Регулярные выражения

современные технологии простым языком

Модуль 2. Регулярные выражения. План

- Классификация регулярных выражений.
- Поиск текста с помощью grep.
- Использование обратных ссылок.
- Использование регулярных выражений с sed.
- Регулярные выражения в awk.

Классификация регулярных выражений

- Регулярное выражение – шаблон для поиска текста
- Обычные и расширенные
- Особые регулярные выражения, например в PERL

Классификация регулярных выражений

- Метасимволы – шаблоны и квантификаторы

Шаблоны обычные	Квантификаторы обычные
. – любой символ	* - любое количество
^ - начало строки	? – ноль или один раз
$ - конец строки	
[] - диапазон	

Шаблоны расширенные	Квантификаторы расширенные
\< – начало слова	+ - один и более раз
\> – конец слова	{n} – n раз
() – группа символов	{n,m} – от n до m раз
	{n,} – не менее n раз

Поиск текста с помощью grep

- Команда `grep` производит поиск в указанных файлах строк по регулярному выражению
  - `grep` – обычный синтаксис
  - `egrep` – расширенный синтаксис
  - `fgrep` – не использовать регулярные выражения

Задание

- Составление регулярных выражений
- Использование grep

Использование обратных ссылок

- Обратная ссылка (back reference) – позволяет обратиться к найденной подстроке

Задание

- Использование обратных ссылок

Использование регулярных выражений с sed

- Используются для нахождения строк, подлежащих обработке
- Указываются в косых чертах
- Поддерживает обычный синтаксис
- Не поддерживает обратных ссылок
- Можно использовать символ &
- Опция `-r` – включает расширенный синтаксис и обратные ссылки

Регулярные выражения в awk

- Позволяют указать строки подлежащие обработке
- Используется оператор соответствия ~
- Может сравнивать отдельные поля
- Не поддерживает обратных ссылок

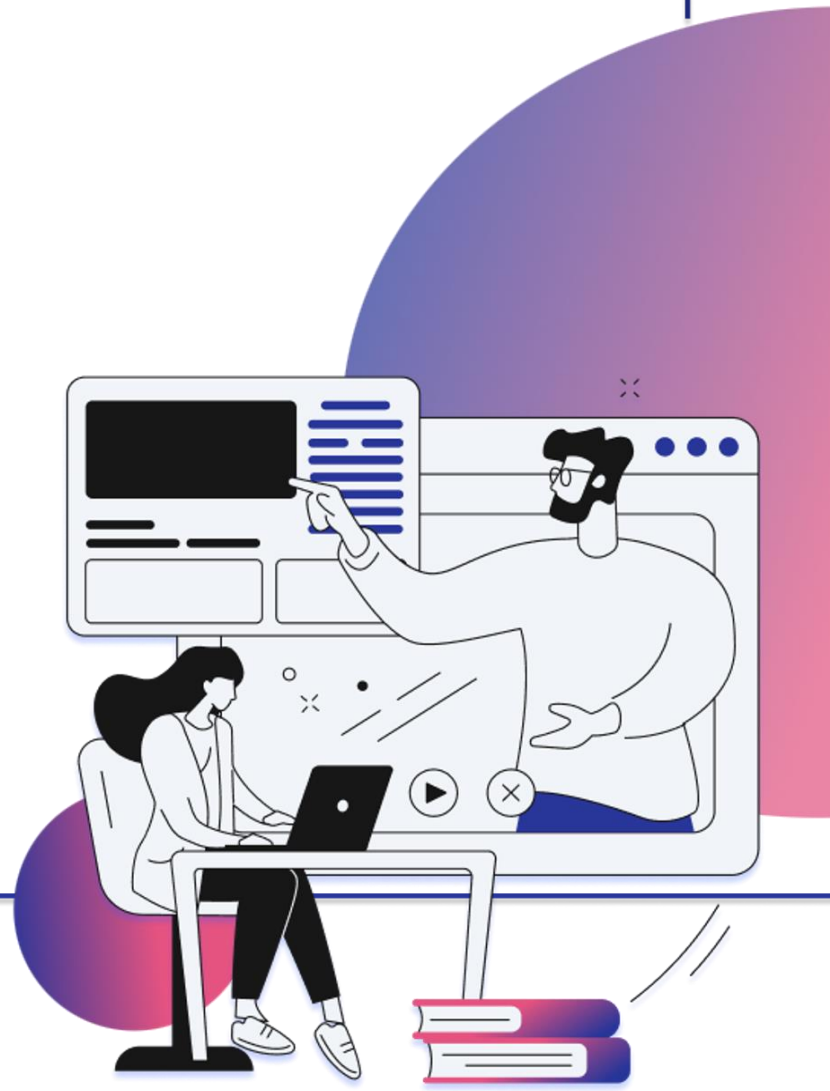
Задание

- Использование регулярных выражений в sed и awk

Что изучили в данном модуле

- Классификация регулярных выражений.
- Поиск текста с помощью grep.
- Использование обратных ссылок.
- Использование регулярных выражений с sed.
- Регулярные выражения в awk.

Учебный центр АйТи Клауд



Модуль 3. Написание сценариев Bash

современные технологии простым языком

Модуль 3. Написание сценариев Bash. План

- Сценарии оболочки.
- Использование переменных оболочки.
- Интерактивная установка значений переменных.
- Позиционные параметры.
- Оператор test.
- Условное исполнение команд.
- Оператор case.
- Циклы.
- Функции.

Сценарии оболочки

- Сценарий – текстовый файл с командами и синтаксическими конструкциями
- Команды выполняются последовательно
- Могут запускаться как команды или как аргумент при запуске оболочки

Сценарии оболочки

- Для отладки полезны опции `-v` или `-x`
- Конструкция #! – указывает оболочку, которая исполняет команды (shebang)
- Команда `source` или `.` – inline подстановка

Задание

- Создание сценария
- Запуск сценария
- Шебанг

Использование переменных оболочки

- Переменные – строки
- Можно выполнять простейшие арифметические действия
- Имена чувствительны к регистру
- Имя должно начинаться с буквы или символа _

Использование переменных оболочки

- `${}`
- `${имя:-значение}`
- `${имя:=значение}`
- `${имя:+значение}`

Массивы

- `declare`
  - `-A`
  - `-a`
- `unset`

Интерактивная установка значений переменных

- Команда `read` – считывает значения переменных из стандартного потока ввода

Позиционные параметры

- Позиционные параметры – аргументы скрипта
- `$0` – имя команды;
- `$1` до `$9` – аргументы с 1 по 9
- `${10}…` – аргументы с 10 и далее
- Команда `shift` – сдвигает позиционные параметры

Позиционные параметры

- `$*` - строка из значений всех аргументов командной строки, разделенных символом разделителем по умолчанию, заданному в переменной IFS;
- `$@` - строка из значений всех аргументов командной строки, разделенных пробелами;
- `$#` - количество аргументов командной строки;
- `$?` - код возврата предыдущей команды;
- `$$` - PID оболочки.

Задание

- Использование переменных
- Считывание потока ввода
- Отладка
- Сдвиг и установка новых аргументов запуска

Оператор test

- Проверка существования и атрибутов файлов
- Сравнение файлов
- Проверка установки опций оболочки
- Сравнение строк
- Сравнение целых чисел
- Конструкция `[ ]` – синоним `test`

Условное исполнение команд

- `&&` - успешный запуск предыдущей команды
- `||` - неудачный запуск предыдущей команды
- ```
  if условие
  then
       команды
  fi
  ```
  ```

Оператор case

```
case слово in
    шаблон1 )
        команды
        ;;
    шаблон2 )
        команды
        ;;
    *)
        команды
        ;;
esac
```

Задание

- Проверка файлов и переменных
- Использование условных операторов

Циклы

- Цикл for для работы со списком

```
for переменная in список
do
        команда
done
```

- Цикл for со счетчиком

```
for ((i=1; i<=N ; i++))
do
        команда
done
```

Циклы

- Цикл while работает с предпроверкой условия

```
counter=0
while [[ $counter -le N ]]
do
        echo $counter
        (( counter++ ))
done
```

- Цикл until работает аналогично while

Задание

- Использование циклов

Функции

```
function имя()
{
    команды
}
```

Задание

- Использование функций

Что изучили в данном модуле

- Сценарии оболочки.
- Использование переменных оболочки.
- Интерактивная установка значений переменных.
- Позиционные параметры.
- Оператор test.
- Условное исполнение команд.
- Оператор case.
- Циклы.
- Функции.

Спасибо, что выбрали обучение в УЦ АйТи Клауд!

Желаем успешно применить полученные знания на практике!

И будем ждать новых встреч!

Связаться с преподавателем pgruzdev@itcloud-edu.ru

Наш сайт