Лабораторная работа №12

Программирование в командном процессоре OC UNIX. Расширенное программирование

Рытов Алексей Константинович НФИбд-02-21

# 1 Цель работы

Изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 2 Выполнение лабораторной работы

1. Реализовали команду man с помощью командного файла.

Создание файлов для дальнейшей работы

Создание файлов для дальнейшей работы



Скрипт 2.sh

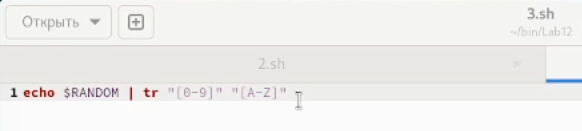
Запуск скрипта

Запуск скрипта

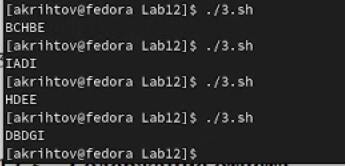


Результат работы скрипта

1. Используя встроенную переменную $RANDOM, написали командный файл, генерирующий случайную последовательность букв латинского алфавита.



Скрипт 3.sh



Результаты работы скрипта

# 3 Выводы

Мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# 4 Ответы на контрольные вопросы:

1. Найдите синтаксическую ошибку в следующей строке:

while [$1 != “exit”]

отсутсвуют пробелы рядом с квадратными скобками

1. Как объединить (конкатенация) несколько строк в одну?

VAR1=“Hello,”

VAR2=” World”

VAR3=“VAR2”

echo “$VAR3”

1. Найдите информацию об утилите seq. Какими иными способами можно реализовать её функционал при программировании на bash?
2. Какой результат даст вычисление выражения $((10/3))?

3,(3)

1. Укажите кратко основные отличия командной оболочки zsh от bash.

Zsh более интерактивный и настраиваемый, чем Bash.

У Zsh есть поддержка с плавающей точкой, которой нет у Bash.

В Zsh поддерживаются структуры хеш-данных, которых нет в Bash.

Функции вызова в Bash лучше по сравнению с Zsh.

1. Проверьте, верен ли синтаксис данной конструкции for ((a=1; a <= LIMIT; a++))

верен

1. Сравните язык bash с какими-либо языками программирования. Какие преимущества у bash по сравнению с ними? Какие недостатки?

hell-сценарий — исходный файл, специально созданный для интерпретатора командной строки типа Bash. Программисты обычно пишут Shell-сценарии для повышения производительности за счет автоматизации повторяющихся задач, таких как:

обработка файлов; установка среды; запуск тестовых стеков и развертывания. Кроме того, Shell-сценарии используются внутри виртуальных машин или CI/CD сервисов, обеспечивая чистые и настраиваемые тестовые прогоны или развертывания.