Шаблон отчёта по лабораторной работе 12

Программирование в командном процессоре ОС UNIX.

Абдуллахи Бахара

Содержание

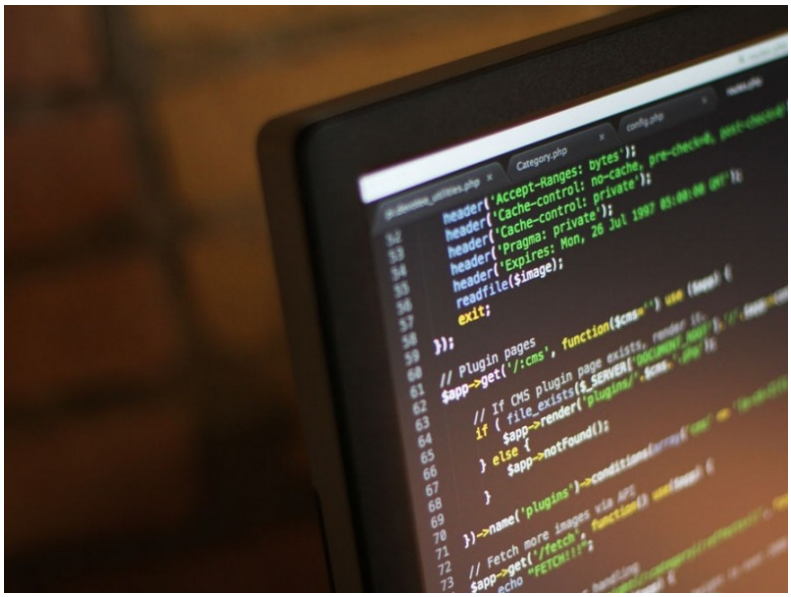
Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. [-@tbl:std-dir] приведено краткое описание стандартных каталогов Unix.

Описание некоторых каталогов файловой системы GNU Linux {#tbl:std-dir}

Имя каталога	Описание каталога
/	Корневая директория, содержащая всю файловую
/bin	Основные системные утилиты, необходимые как в однопользовательском режиме, так и при обычной работе всем пользователям
/etc	Общесистемные конфигурационные файлы и файлы конфигурации установленных программ
/home	Содержит домашние директории пользователей, которые, в свою очередь, содержат персональные настройки и данные пользователя
/media	Точки монтирования для сменных носителей
/root	Домашняя директория пользователя root
/tmp	Временные файлы
/usr	Вторичная иерархия для данных пользователя

Более подробно про Unix см. в [@tanenbaum_book_modern-os_ru; @robbins_book_bash_en; @zarrelli_book_mastering-bash_en; @newham_book_learning-bash_en].



Название рисунка

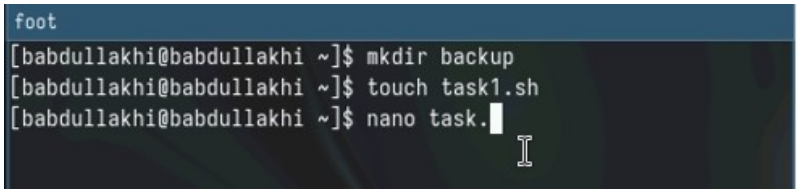
Цель работы:

Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы.

Выполнение лабораторной работы:

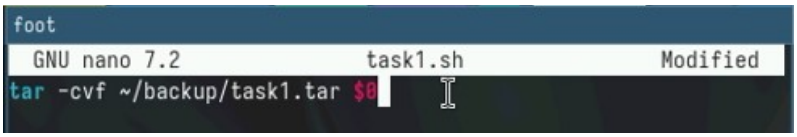
1. Написать скрипт, который при запуске будет делать резервную копию самого себя (то есть файла, в котором содержится его исходный код) в другую директорию backup в вашем домашнем каталоге. При этом файл должен архивироваться одним из архиваторов на выбор zip, bzip2 или tar. Способ использования команд архивации необходимо узнать, изучив справку.

- в этом разделе сначала я создала каталог backup после этого я создала файл task1.sh, в этом я написала следушие код чтобы сделала архивироваться.



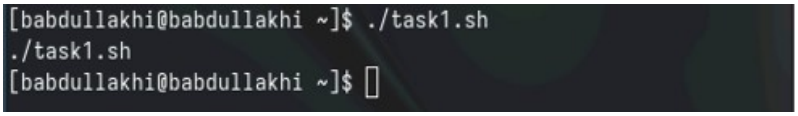
создание каталог и файл

- Написание первого скрипта.



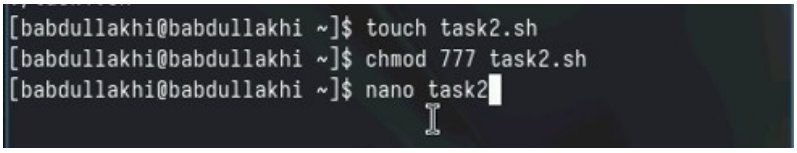
Написано код для архивирование

- Право на выполнение, запуск файла и проверка.



проверка

2. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов.
- опять я создала файл для втрого скрепта task2.sh

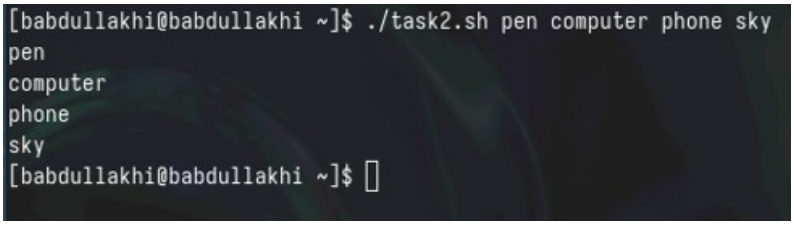


Создание файла для скрепта

- Печатаем все аргументы командной строки • обрабатывающего любое произвольное число аргументов командной строкию

Написано код для печатние аргументы

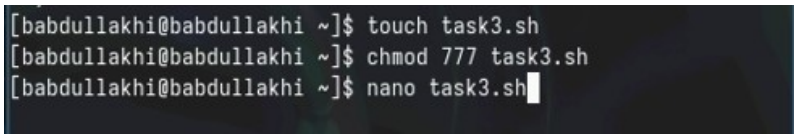
- Право на выполнение, запуск файла и проверка



Название рисунка

3. Написать командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога.

- опять я создала файл для 3 скрепта task3.sh

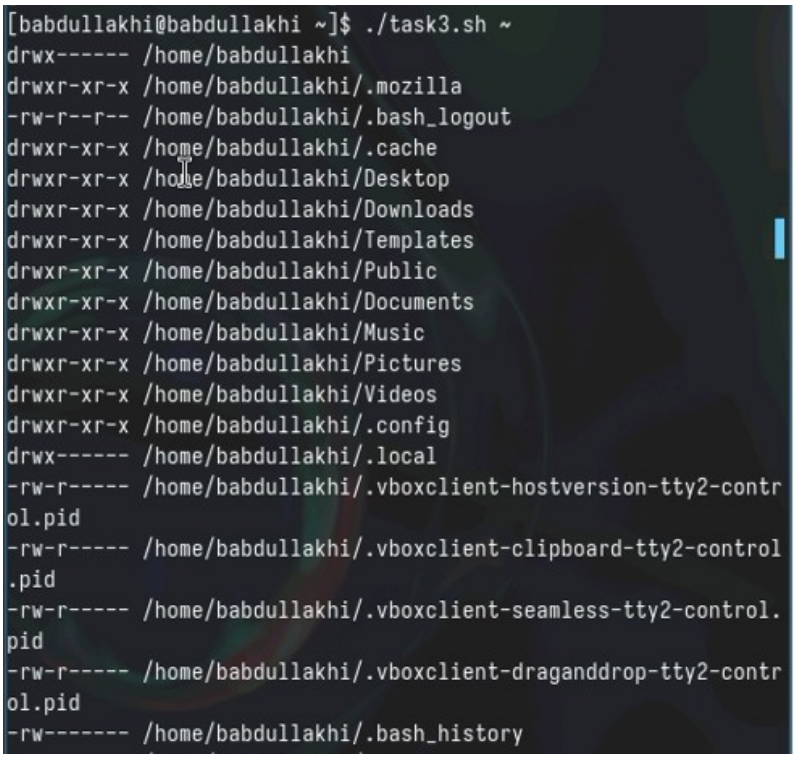


Создание файла для скрепта

- Командный файл — аналог команды ls (без использования самой этой ко манды и команды dir)

Командный файл

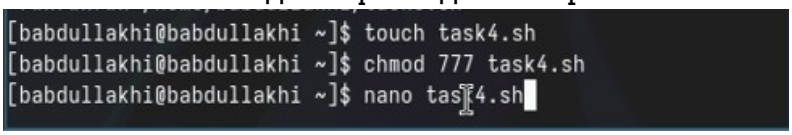
- Право на выполнение, запуск файла и проверка



Название рисунка

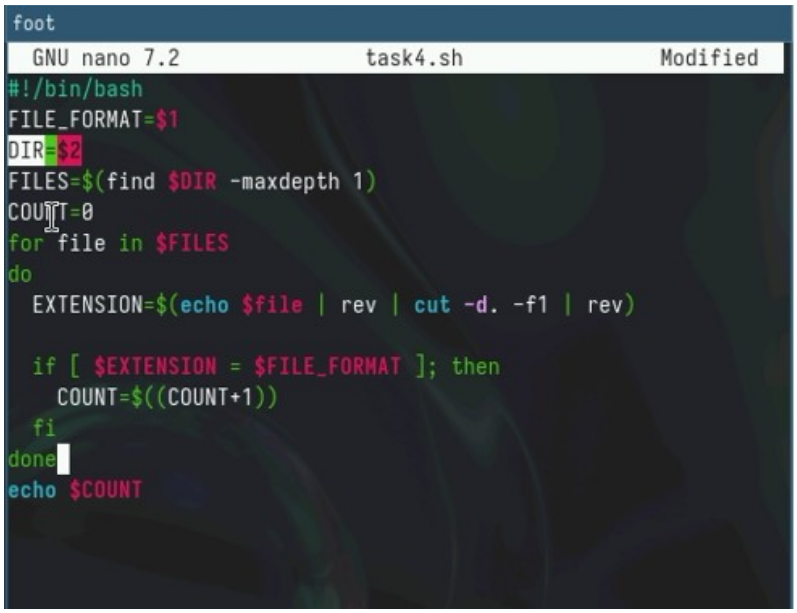
4. Написать командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории. Путь к директории также передаётся в виде аргумента командной строки.

- опять я создала файл для 3 скрепта



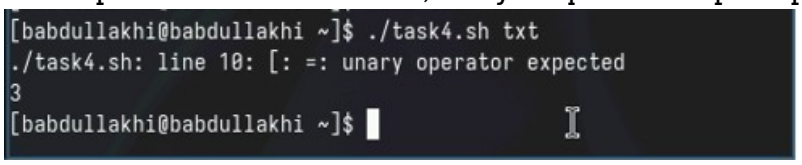
Название рисунка

- Командный файл, который получает в качестве аргумента командной строки формат файла (.txt, .doc, .jpg, .pdf и т.д.) и вычисляет количество таких файлов в указанной директории.



Создание файла для скрепта

- Право на выполнение, запуск файла и проверка



проверка

Ответы на контрольные вопросы:

1. Объясните понятие командной оболочки. Приведите примеры командных оболочек. Чем они отличаются? Ответ: Командная оболочка - это программа, которая предоставляет пользо вателю интерфейс для взаимодействия с операционной системой. Она прини мает команды от пользователя и выполняет их. Примеры командных оболочек: • Bash • Zsh • Tcsh • Ksh • Csh Отличия между ними заключаются в наборе встроенных команд, синтаксисе и возможностях настройки.
2. Что такое POSIX? Ответ: POSIX (Portable Operating System Interface for Unix) - это набор стандар тов, которые определяют интерфейс между операционной системой и прило жениями. Он обеспечивает переносимость программного обеспечения между различными Unix-подобными системами.
3. Как определяются переменные и массивы в языке программирования bash? Ответ: • Переменные: Объявляются с помощью оператора присваивания (=). Напри мер: VAR=value. •

Массивы: Объявляются с использованием круглых скобок и разделяются пробелами. Например: ARRAY=(value1 value2 value3).

4. Каково назначение операторов let и read? Ответ: • let: Используется для выполнения арифметических операций и присваива ния значений переменным. • read: Считывает ввод с устройства ввода и присваивает его переменной.
5. Какие арифметические операции можно применять в языке программи рования bash? Ответ: • Сложение (+) • Вычитание (-) • Умножение (*) • Деление (/) • Остаток от деления (%) • Возведение в степень (**)
6. Что означает операция (( ))? Ответ: Операция (( )) используется для выполнения более сложных арифмети ческих операций, включая логические и условные операции.
7. Какие стандартные имена переменных Вам известны? Ответ: • $? - код возврата последней выполненной команды 12 • $$ - идентификатор текущего процесса • $! - идентификатор последней запущенной фоновой задачи • $# - количество аргументов, переданных в командный файл • $@ - массив всех аргументов, переданных в командный файл
8. Что такое метасимволы? Ответ: Метасимволы - это специальные символы, которые имеют особое зна чение в командной оболочке. Например: • * - совпадает с любым количеством любых символов • ? - совпадает с любым одним символом • [ ] - совпадает с любым символом внутри квадратных скобок
9. Как экранировать метасимволы? Ответ: Метасимволы можно экранировать с помощью обратной косой черты (). Например: * будет совпадать с символом звездочки (*).
10. Как создавать и запускать командные файлы? Ответ: • Создание: Используйте текстовый редактор, чтобы создать файл с расшире нием .sh и ввести команды bash. • Запуск: Введите ./filename.sh в командной строке.
11. Как определяются функции в языке программирования bash? Ответ: Функции определяются с использованием ключевого слова function, за которым следует имя функции и список параметров (если есть). Например: function my_function() { # Код функции }
12. Каким образом можно выяснить, является файл каталогом или обычным файлом? Ответ: Используйте оператор -d для каталогов и -f для обычных файлов. На пример: if [-d filename]; then echo “Файл является каталогом” fi
13. Каково назначение команд set, typeset и unset? Ответ: • set: Отображает или устанавливает значения переменных. • typeset: Объявляет переменные и указывает их тип. • unset: Удаляет переменные.

14. Как передаются параметры в командные файлы? Ответ: Параметры передаются в командные файлы в виде аргументов команд ной строки. Они доступны через переменную $@.
15. Назовите специальные переменные языка bash и их назначение. Ответ: • $0: Имя текущего командного файла. • $#: Количество аргументов, переданных в командный файл. • $@: Массив всех аргументов, переданных в командный файл. • $?: Код возврата последней выполненной команды. • $$: Идентификатор текущего процесса. • $!: Идентификатор последней запущенной фоновой задачи.