Лабораторная работа №12

Простой способ выполнения лабораторной работы №12

Атанесов Александр Николаевич

Содержание

# 1 Цель работы

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

# 2 Задание

1.Взаимодействрвать с ОС через терминал посредством команд;

# 3 Теоретическое введение

Здесь описываются теоретические аспекты, связанные с выполнением работы.

Например, в табл. [[1](#tbl:std-dir)] приведено краткое описание стандартных каталогов Unix.

Table 1: Описание некоторых каталогов файловой системы GNU Linux

| 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 об Unix см. в [1–6].

# 4 Выполнение лабораторной работы

1. Создаю файл command.sh. (рис. [??])

Использую команду touch

Использую команду touch

## 4.1

1. Делаю файл исполняемым. (рис. [??])

Использую команду chmod

Использую команду chmod

## 4.2

1. Создаю файл semaphore.lock. (рис. [??])

Использую команду touch

Использую команду touch

## 4.3

4.Создаю файл output.txt для автоматической записи изменений . (рис. [??])

Использую команду touch

Использую команду touch

## 4.4

1. Открываю файл command.sh через nano. (рис. [??])

Использую команду nano

Использую команду nano

## 4.5

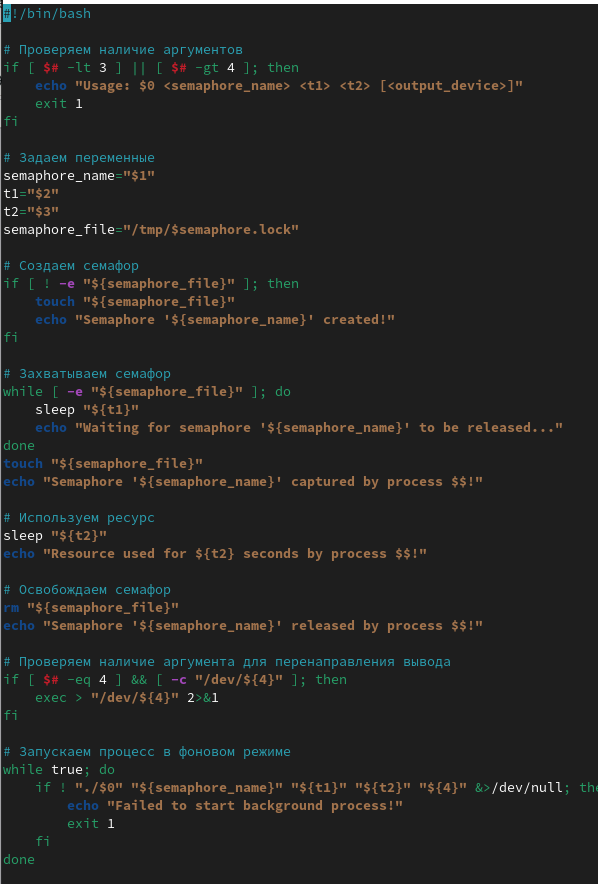
1. Открываю файл command.sh через nano. (рис. [??])

Использую команду nano

Использую команду nano

## 4.6

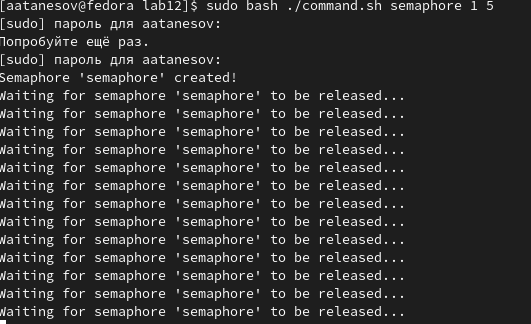
1. Пишу необходимый код для выполнения условий задачи 1 . (рис. [??])



Использую редактор nano

## 4.7

1. Запускаю файл commmand.sh с semaphore . (рис. [??])



Использую команду ./

## 4.8

1. Создаю файл man.sh . (рис. [??])

Использую команду touch

Использую команду touch

## 4.9

1. Делаю файл исполняемым. (рис. [??])

Использую команду chmod

Использую команду chmod

## 4.10

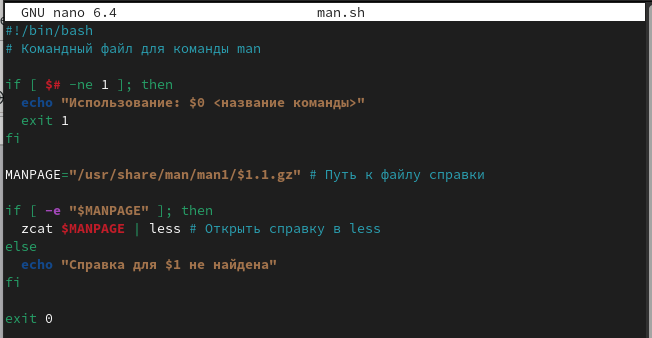
1. Открываю файл man.sh через nano. (рис. [??])

Использую команду nano

Использую команду nano

## 4.11

1. Пишу необходимый код. (рис. [??])



Использую редактор nano

## 4.12

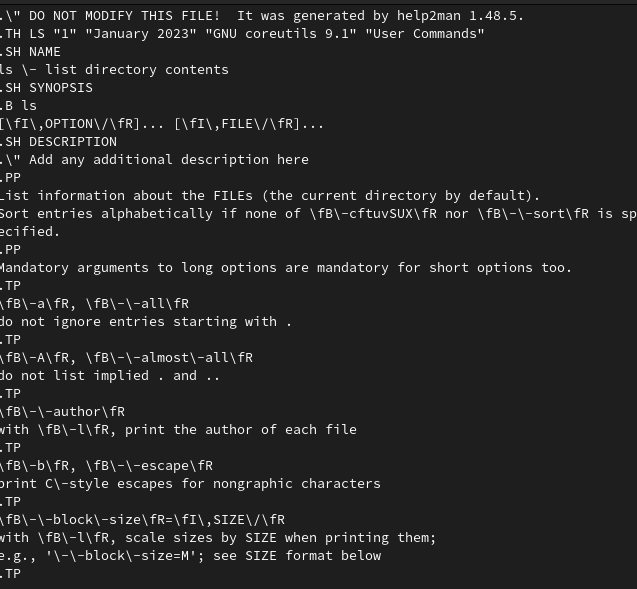
1. Запускаю файл man.sh с выводом справки для команды ls. (рис. [??])

Использую команду ./

Использую команду ./

## 4.13

1. Вижу вывод на команду ls. (рис. [??])

(image/13.png){#fig:014 width=90%}

## 4.14

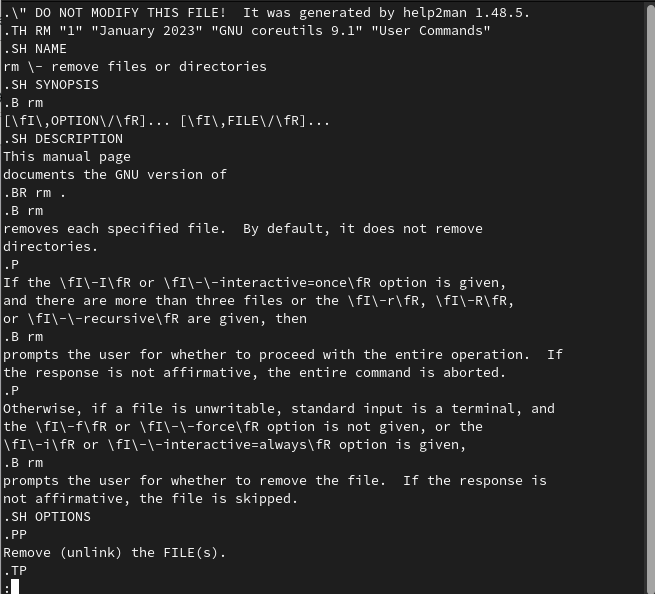
1. Запускаю файл man.sh с выводом справки для команды rm. (рис. [??])

Открываю файл через ./

Открываю файл через ./

## 4.15

1. Вижу вывод на команду rm. (рис. [??])

(image/18.png){#fig:018 width=90%}

## 4.16

1. Создаю файл latyn.sh для выполнения третьего этапа. (рис. [??])

Использую команду ./

Использую команду ./

## 4.17

1. Делаю файл исполняемым . (рис. [??])

Использую команду chmod

Использую команду chmod

## 4.18

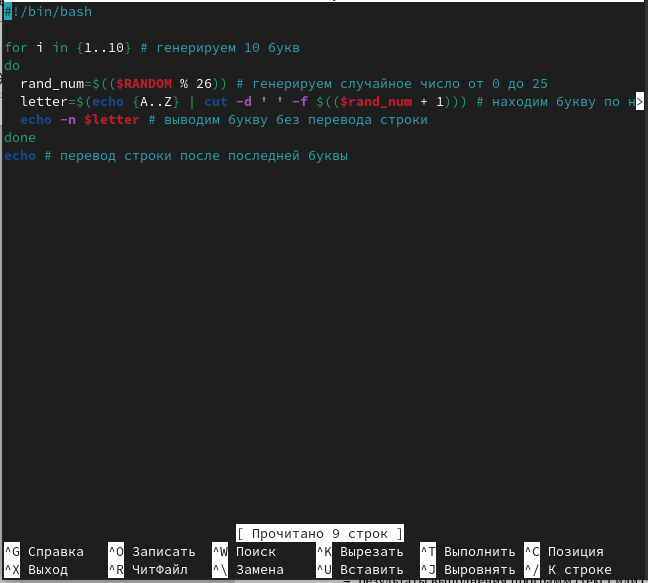
1. Открываю файл latyn.sh через nano. (рис. [??])

Использую команду nano

Использую команду nano

## 4.19

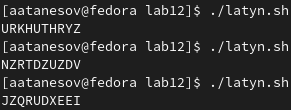
1. Ввожу необходимый код. (рис. [??])



Использую редактор nano

## 4.20

1. Запускаю файл latyn.sh . (рис. [??])



Использую команду ./

# 5 Выводы

* Познал основы программирования в UNIX. # Ответы на контрольные вопросы
  1. Отсутствует пробел между скобками и оператором условия. Правильно: while [ $1 != “exit” ].
  2. Можно использовать оператор конкатенации - символ “+” или переменную, содержащую объединяемые строки.
  3. Утилита seq используется для генерации числовых последовательностей. Её функционал можно реализовать с помощью циклов с оператором перебора и арифметических операций.
  4. Результатом вычисления будет число 3.3333333, но в bash результат целочисленного деления (оператор “//”) будет без округления, т.е. равным 3.
  5. zsh имеет более широкие возможности для настройки и расширения, например, более продвинутые автодополнение команд и параметров. Она также поддерживает более мощный синтаксис и некоторые удобные функции, например, поддержку массивов. Однако bash является более распространенным и стабильным в использовании.
  6. Синтаксис верен, но значение переменной LIMIT необходимо задать заранее.
  7. Bash относится к скриптовым языкам программирования. Он отличается от императивных языков (например, C++, Java) тем, что команды выполняются последовательно, без явного объявления переменных и типов данных. Преимуществами bash является его простота в использовании, поддержка большинства UNIX-систем и гибкость в написании скриптов. Однако он может иметь низкую производительность при обработке больших объемов данных.

# Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016. URL: <https://www.gnu.org/software/bash/manual/>.

2. Newham C. [Learning the bash Shell: Unix Shell Programming](http://www.amazon.com/Learning-bash-Shell-Programming-Nutshell/dp/0596009658). O’Reilly Media, 2005. 354 с.

3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.

4. Robbins A. [Bash Pocket Reference](https://www.ncbi.nlm.nih.gov/pubmed/25246403). O’Reilly Media, 2016. 156 с.

5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.

6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.