

Front matter

lang: ru-RU title: "Лабораторная работа №11" subtitle: "Программирование в командном процессоре ОС UNIX. Командные файлы" author: "Данилова Анастасия Сергеевна"

Formatting

toc-title: "Содержание" toc: true # Table of contents toc_depth: 2 lof: true # List of figures lot: true # List of tables
fontsize: 12pt linestretch: 1.5 papersize: a4paper documentclass: scrreprt polyglossia-lang: russian polyglossia-
otherlangs: english mainfont: PT Serif romanfont: PT Serif sansfont: PT Sans monofont: PT Mono mainfontoptions:
Ligatures=TeX romanfontoptions: Ligatures=TeX sansfontoptions: Ligatures=TeX,Scale=MatchLowercase
monofontoptions: Scale=MatchLowercase indent: true pdf-engine: lualatex header-includes:

- `\linepenalty=10` # the penalty added to the badness of each line within a paragraph (no associated penalty node) Increasing the value makes tex try to have fewer lines in the paragraph.
- `\interlinepenalty=0` # value of the penalty (node) added after each line of a paragraph.
- `\hyphenpenalty=50` # the penalty for line breaking at an automatically inserted hyphen
- `\exhyphenpenalty=50` # the penalty for line breaking at an explicit hyphen
- `\binoppenalty=700` # the penalty for breaking a line at a binary operator
- `\relpenalty=500` # the penalty for breaking a line at a relation
- `\clubpenalty=150` # extra penalty for breaking after first line of a paragraph
- `\widowpenalty=150` # extra penalty for breaking before last line of a paragraph
- `\displaywidowpenalty=50` # extra penalty for breaking before last line before a display math
- `\brokenpenalty=100` # extra penalty for page breaking after a hyphenated line
- `\prelispdisplaypenalty=10000` # penalty for breaking before a display
- `\postdisplaypenalty=0` # penalty for breaking after a display
- `\floatingpenalty = 20000` # penalty for splitting an insertion (can only be split footnote in standard LaTeX)
- `\raggedbottom` # or `\flushbottom`
- `\usepackage{float}` # keep figures where there are in the text
- `\floatplacement{figure}{H}` # keep figures where there are in the text

Цель работы

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

Задание

1. Написать скрипт, который при запуске будет делать резервную копию самого себя
2. Написать пример командного файла, обрабатывающего любое произвольное число аргументов командной строки
3. Написать командный файл — аналог команды `ls` (без использования самой этой команды и команды `dir`)
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.

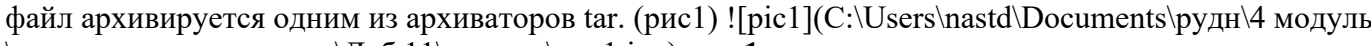
Теоретическое введение

Командный процессор — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек:

– оболочка Борна (Bourne shell или `sh`) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций; – C-оболочка (или `csh`) — надстройка на оболочкой Борна, использующая C-

подобный синтаксис команд с возможностью сохранения истории выполнения команд; – оболочка Корна (или ksh) — напоминает оболочку C, но операторы управления программой совместимы с операторами оболочки Борна; – BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек C и Корна .

Выполнение лабораторной работы

1. Напишем 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 архивируется одним из архиваторов tar. (рис1)  `![[pic1](C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис1.jpg)]` **рис 1**

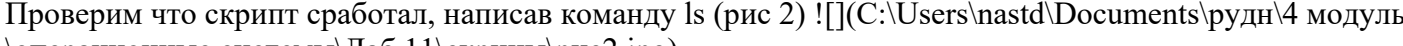
Проверим что скрипт сработал, написав команду ls (рис 2)  `![[C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис2.jpg)]`

рис 2

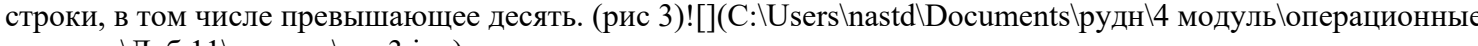
2. Напишем пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. (рис 3)  `![[C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис3.jpg)]`

рис 3

Убедимся что все работает(рис 4)  `![[C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис4.jpg)]`

рис 4


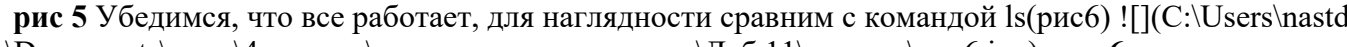
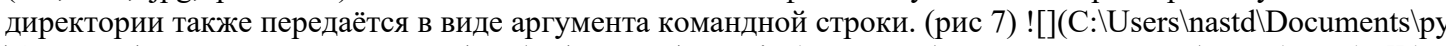
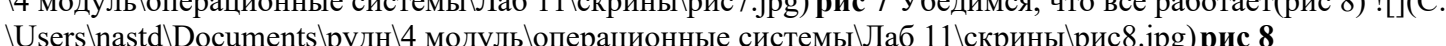
3. Напишем командный файл — аналог команды ls (без использования самой этой команды и команды dir). Требуется, чтобы он выдавал информацию о нужном каталоге и выводил информацию о возможностях доступа к файлам этого каталога. (рис 5)  `![[C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис5.jpg)]`

рис 5 Убедимся, что все работает, для наглядности сравним с командой ls(рис6)  `![[C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис6.jpg)]` **рис 6**

4. Напишем 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. (рис 7)  `![[C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис7.jpg)]` **рис 7** Убедимся, что все работает(рис 8)  `![[C:\Users\nastd\Documents\рудн\4 модуль\операционные системы\Лаб 11\скрины\рис8.jpg)]` **рис 8**

Выводы

Мы изучили основы программирования в оболочке ОС UNIX/Linux. Научились писать небольшие командные файлы.

Библиография:

<https://losst.ru/komanda-ls-linux>

<https://losst.ru/kak-zapustit-programmu-na-linux->