Университет ИТМО, кафедра ВТ

Лабораторная работа №1 по “Системному Программному Обеспечению”

Работу выполнил

студент группы P3200

Рогов Я. С.

Преподаватели:

Дергачёв А.М.

Жмылёв С.А.

Санкт-Петербург, 2016

Задание 1: команды для повторения

ls — выводит список файлов в рабочей директории, либо в директории, указанной как аргумент

Основные ключи:

-a — выводит список всех файлов, включая скрытые директории (начинающиеся с точки, ".filename")

-d — если аргументом является директории, то выводится только её название, а не её содержимое.

-l — выводит список файлов подробно: каждый файл выводится в виде:

`права_доступа колво_жестких_ссылок владелец группа_владельца размер время_последней_модификации`

-R — рекурсивно выводит содержимое субдиректорий

-i — выводит номер i-node рядом с названием файла

Фильтр: нет

pwd — выводит название рабочей директории.

Фильтр: нет

cd — меняет рабочую директорию на домашнюю директорию, либо в директориию, указанную как аргумент

Фильтр: нет

rm - удаляет файл(ы), указанный как аргумент(ы)

Основные ключи:

-r — если аргумент является директорией, то рекурсивно удаляет его содержимое

-f - удаляет все файлы (даже защищённые на запись) в директории, если она не защищена на запись. При попытке удаления защищённых на запись субдиректорий выведется ошибка.

-i — спрашивает подтверждение об удалении каждого файла

Фильтр: нет

mv — перемещает файлы из одного места в другое. Также может использоваться для переименования.

Примеры использования:

mv file1 file2 — переименовывает файл

mv file1 dir1 — перемещает файл в директорию

mv file1 file2 … fileN dir1 — перемещает файлы в директорию

Основные ключи:

-f - не спрашивает пользователя о перезаписи файла

-i — спрашивает пользователя перед каждой перезаписью существующего файла.

Фильтр: нет

cp - копирует файлы

Основные ключи:

Те же, что и у ***mv.***

-r — рекурсивно копирует директории. При попытке скопировать директорию с файлами, команда "пропустит" (omit) директорию.

-R — аналогично ***-r,*** однако при попытке копирования именованного канала (пайпа) действительно копирует его, а не просто перенаправляет в новый пайп, как бы это произошло при отсутствии ключа

-P, -L, -H — различное поведение при копировании символьных ссылок: копирование символьной _ссылки_, "следование" по одной символьной ссылке и копирование файла, "следование" по нескольким символьным ссылкам подряд и копирование файла

Фильтр: нет

mkdir - создаёт директорию (-и)

Основные ключи:

-m, --mode - устанавливает права у созданных директорий. Доступны как и числовой режим установки прав, так и символьный (как в ***chmod***)

-p — создавать родительские директории по необходимости (при их отсутствии).

Фильтр: нет

rmdir - удаляет пустую (-ые) директорию (-ии)

Основные ключи:

-p — позволяет удалять вместе с указанной директорией все её родительские директории, указанные в пути к директории (в аргументе), если они пустые

-s — отключает сообщения для стандартного потока ошибках о возникающих ошибках при удалении с ключом **-p.**

Фильтр: нет

type — выводит описание типа команд (-ы): встроенная в интерпретатор, функция, псевдоним (alias), хэшированная команда, или ключевое слово.

Фильтр: нет

file — определяет тип файла (-ов) путём проведения нескольких тестов на содержание файла (-ов).

Основные ключи:

-f ffile — изучает список файлов, указанных в ffile

-h — при попытке изучить символическую ссылку, будет указывать её именно как символическую ссылку, а не попытается изучить файл, на который эта ссылка указывает.

-i — классифицирует файл как один из типов файлов в Unix: обычный файл, директория, блочное устройство и т.д.

Фильтр: нет

find — производит поиск файлов, удовлетворяющих выражению, в директории, переданному как аргумент

Основные выражения, используемые при поиске*:*

-follow — всегда Истина. Переходит по символьным ссылкам

-group группа — Истина, если файл принадлежит группе с именем/номером группа

-inum n — Истина, если файл имеет номер inode равный n

-ls — всегда Истина. Выводит описание файла в формате, подобном при выполнении команды "***ls -libs***":

```
inode size_in_kb mode hard_links user group size_in_bytes mtime file
```

-name <u>шаблон</u> — Истина, если имя файла удовлетворяет шаблону (wildcard), поддерживающий те же метасимволы, что и командный интерпретатор: *, ?, [,], - и ~

-perm <u>в_число</u> — Истина, в случае, если полного совпадения прав на файле

-type <u>тип</u> — Истина, если файл имеет данный <u>тип</u> (b — блочное устройство, c — символьное устройство, d — директория и т.д.)

Фильтр: нет

chmod — изменяет права доступа файла (-ов)

Основные ключи:

-R — рекурсивно меняет права доступа всех файлов в директориию

-f — в случае неудачной попытки изменить права файла ошибка не будет выведена

Фильтр: нет

ln — создаёт ссылку (-и) (и жёсткие, и символьные) на файлы

Основные ключи:

-f — создаёт ссылку без спрашивания разрешения пользователя в случае, если права доступа файла не разрешают записи.

-s — создаёт <u>символьную</u> ссылку.

-n — если при создании ссылки файл с данным названием уже существует, ошибка записывается в стандартный поток ошибок и продолжает. Подавляется ключом **-f**

***Фильтр*:** нет

wc — выводит количество строк, слов, и байт в файле (-ах) при отсутствии ключа, отвечающего за вывод чего-то конкретного.

Основные ключи:

-c — выводит количество байт

-C/-m — выводит колиество символов

-w — выводит количество слов

-l — выводит количество строк

Фильтр: да

tee — копирует стандартный ввод в стандартный вывод и в файлы, переданные как аргументы.

Основные ключи:

-a — вместо перезаписи содержимого файла будет происходить добавление в файл (append)

Фильтр: да

cat — соединяет файлы и выводит их содержимое

Основные ключи:

-n — в начале каждой строки печатает её номер

-s — команда не выводит сообщения об ошибке при обнаружении несуществующего файла

-v — непечатаемые символы (кроме символов new-line, page-break, и табуляций) печатаются в виде ^n (символы с кодами с 0x0 по 0x1F) и M-x (все остальные непечатаемые символы)

Фильтр: да

tail — выводит конец (последние число байт, строк, блоков) файла

Основные ключи:

-c — число обозначает количество байт

-l — число обозначает количество строк

-f — программа начинает выполняться в бесконечной петле, раз в секунду печатая новые данные, добавленные в файл.

*Ключи указываются в формате -[число][ключи]. Ключи -**c** и -**l** требуют указания числа.*

Фильтр: да

head — выводит первые число (по умолчанию — 10) строк файла (-ов).

Основные ключи:

-n число, -число - выводит первые число линий файла.

Фильтр: да

more — постраничный просмотр текстового файла

Основные ключи:

-s — при выводе заменяет несколько пустых строк подряд одной.

-w — после окончания вывода файла, не выходит из программы, как это задано по-умолчанию.

-число — выводит число строк на каждой странице, вместо (максимальное_число_строк_в_терминале — 2)

+число — вывод начинается с строки с номером число.

+/шаблон — вывод начинается с места, двумя строками выше первого найденного слова, удовлятворяющего шаблону.

Фильтр: да

pg — постранично выводит файл (-ы)

Основные ключи:

-число — число строк в экране, используемом программой.

-f — запрещение разделения строк на несколько. По-умолчанию, программа разделяет строки с длинной большей, чем ширина экрана, на несколько строк.

+число — вывод начинается со строки с номером число.

+/шаблон/ - вывод начинается со строки, содержащий строку, удовлетворяющую шаблону.

Фильтр: да

touch — изменяет время модификации и доступа к файлу (-ам) (по-умолчанию).

Основные ключи:

-a — меняет время доступа

-c — не создаёт файл, если он не существует. По-умолчанию, файл создаётся, если он не существует.

-d дата_время— указывает дату и время, которые нужно записать как время доступа и/или модифкации для файла.

-m — меняет время модификации

-t <u>время</u>— указывает время, которое нужно записать как время доступа и/или модификации для файла.

Фильтр: нет

<u>Задание 2</u>: команды для изучения

su - "стать" (создать новый процесс интерпретатора командной строки как) супер-пользователем или другим пользователем (без перелогина)

Основные ключи:

-, — изменение окружения пользователя, вызвавшего команду, на окружение пользователя, переключение на которого осуществляется.

-c <u>команда</u> — исполнение команды в стандартной для указанного пользователя интерпретатора командной строки

Примеры использования:

su dummy -c "mkdir smbwashere" - создаёт директорию "smbwashere" от имени пользователя "dummy"

su root -c "rm -rf /" - делит на ноль.

Фильтр: нет

chown — изменяет владельца у файла (-ов).

Основные ключи:

-f — не выводить сообщения об ошибках.

-R — рекурсивно меняет владельца у файлов в директории

Примеры использования:

chown -R me:us base — меняет владельца директории "base" и её содержимого рекурсивно на пользователя "me" и группу "us"

Фильтр: нет

chgrp — меняет группу-владельца у файла (-ов)

Основные ключи:

те же, что и у **<u>chown</u>**

Фильтр: нет

less - "расширенная" версия **<u>more</u>**, позволяющая и обратную прокрутку

Основные команды:

*Некоторые команды могут принимать опциональный аргумент — число **N** — и интерпретировать его как, например, на длину прокрутки, т. е. команда **5e** совершит прокрутку на 5 строк вперёд, а не 1 по-умолчанию. Для таких команд указано **[N]** в начале команды*

h — вывести список всех команд

[N] f, b — прокрутить на один экран (N строк) вперёд или назад соответственно

[N] e, y — прокрутить на одну строку (N строк) вперёд или назад соответственно

[N] /<u>шаблон</u>, ?<u>шаблон</u> — найти первое (N-ое) вхождение шаблона ниже или выше по тексту соответственно

n, N — повторить поиск ниже или выше по тексту соответственно

[N] g, G — перейти в начало или конец файла (или N-ую строку) соответственно

:e [имяфайла] — открыть файл с именем имяфайла.

[N] :n, :p — открыть следующий или предыдущий (или N-ый) файл, переданный как аргумент соответственно

!команда — выполнить команду через интерпретатора, указанную в переменной $SHELL

v — открыть файл в редакторе, указанном в переменных $VISUAL или $EDITOR

Основные ключи:

-s — при выводе заменяет несколько пустых строк подряд одной.

-f — позволяет открывать файлы, отличные от "обычного" (regular) — директории, файлы устройств

-N - показывать номер строки в её начале

-zчисло — устанавливает прокрутку на число строк, если число положительное, иначе уменьшает размер прокрутки сложением с число.

Примеры использования:

chgrp -R people lifes — рекурсивно меняет группу-владельца директории "lifes" и её содержимого на группу "people".

Фильтр: да

split — разбивает файл на несколько файлов с одинаковым количеством строк или байт

Основные ключи:

-a число — устанавливает длину суффикса, равного числу. Суффикс — часть, добавляемая к названию при создании частей исходного файла. Т.о. можно установить максимальное количество частей, равного 26^число.

-b размер — устанавливает размер каждой части. Размер можно указать как число_байт, число_килобайт**k**, или же число_мегабайт**m**.

-число / -linecount число — устанавливает количество строк текста для каждой части.

Примеры использования:

split -1 100lines line -разбивает файл "100lines" на файлы вида line?? (?? = aa, ab,…) по одной строке каждая.

Фильтр: да

join — выводит объединение двух файлов на основе совпадений определённый полей. Схож с оператором JOIN в SQL.

Основные ключи:

-1/j1/2/j2/j число — устанавливает номер поля в первом/втором/обоих файле (-ax), по которому и будет производиться объединение. По-умолчанию — 1 для обоих файлов.

-t символ — устанавливает символ, который будет рассматриваться программой как разделитель (полей)

Примеры использования:

join people_id ids_phone_numbers — выводит список имён людей и их номеров телефонов на основе совпадения id.

Фильтр: да

paste — выводит построчное соединение двух и более файлов.

Основные ключи:

-d <u>символы</u> — устанавливает символы, которые будут использоваться как разделители между соединёнными строками (по-умолчанию — табуляция)

-s — вывод в "вертикальном виде": т.о. будет вывод не "a b\nc d", а "a c\n b d", т. е. Каждой строчке одного файла будет соответствовать строчка из другого файла, находящаяся под ним, а не справа от него.

Примеры использования:

paste list1 list2 > fulllist — соединяет содержимое двух списков и записывает в файл

Фильтр: да

cut - "вырезает" указанные поля из файлов

Основные ключи:

<u>список</u> — список номеров строк и диапазонов (1-2, -5, 10-), разделённый запятыми.

-b,c,f — устанавливает, что указано в **<u>списке</u>** — байты, символы, или поля.

-d <u>символ</u> — устанавливает символ, интерпретируемый как разделитель *полей*.

-s — игнорирует строки, в которых нет **<u>разделителей</u>** (при установке ключа **<u>-f</u>**)

Также имеется поддержка "классов символов" - строк вида [:class:], где class — одно из определённых слов, которые указывают на определённый набор символов. upper — на буквы в верхнем регистре, alnum — все буквы и цифры, :punct: - символы пунктуации и др.

Примеры использования:

cut -1,3 -f phone_book — выводит список имён людей и их телефонов.

Фильтр: да

tr - "переводит" символы: копирует данные со стандартного ввода на стандартный вывод с заменой/удалением указанных символов.

Основные ключи:

-d - удаляет символы, указанные в <u>строке1</u>

-s — заменяет повторяющиеся подряд одинаковые символы, указанные в аргументах, одним

-c — использует дополнение (все символы, кроме указанных) символов, указанных в <u>строке1</u>.

Примеры использования:

tr "[:lower:]" "{:angry:}" <shyfile >angryfile — копирует текст из shyfile в angryfile, меняя регистр всех букв на верхний.

Фильтр: да

cmp — сравнивает два файла и, в случае найденного несовпадения, выводит его позицию. По-умолчанию — номер символа и строки первого несовпадения.

Основные ключи:

-l — выводит номер байта и различающиеся байты для <u>каждого</u> несовпадения

-s — не выводит ничего, различие файлов передаётся как код возврата. 0 — файлы одинаковые, 1 — файлы различаются

Фильтр: да

diff — сравнивает содержимое двух файлов (или содержимое файлов в них, если это

директории) и выводит список различий (изменения, необходимые для внесения в первый файл/файлы первой директори, чтобы он стал эквивалентен второму/файлам второй директории).

Основные ключи:

-b — игнорировать пробелы и табуляции в конце строк

-i — игнорировать регистр букв.

-w — игнорировать пробелы и табуляции полностью.

-c — подробный режим. Напротив строк ставятся символы: "-" - строки для удаления, "+" - строки для добавления, "!" - строки для изменения.

Основные ключи для работы с директориями:

Сравнение директорий: либо сообщение о наличии файла лишь в одной директории, либо различия между двумя файлами с одинаковым названием (как в файловом режиме работы программы).

-r — рекурсивное сравнение субдиректорий и их содержимого

-s — также выводит сообщение о найденых эквивалентных файлах.

Примеры использования:

diff -r project_v2015 project_v2016 — показывает изменения за год работы над проектом.

Фильтр: да

sort — выводит отсортированный и соединённый текст.

Основные ключи:

-m — простое соединение (как предполагается, уже отсортированных) файлов

-o имяфайла — указывает имя файла, куда будет производится вывод.

-t символ — использует символ как разделитель полей. По-умолчанию — пробел и табуляция.

-k первое_поле [тип] [,последнее_поле [тип]], где **первое_поле** = **номер_поля [.первый_символ]** и **последнее поле [.последний символ] -** "ключ" сортировки. Позволяет указать, с какого символа и какого поля происходит сортировка, и по какой символ какого поля, а также как их интерпретировать (поле "тип"). Может быть несколько, сортировка происходит по ключам, указанным в команде раньше.

Типы:

b — игнорирует пустые символы при определении границ поля

d - "словарный" порядок сортировки: при сравнении учитываются только пробелы, табуляции, цифры, и буквы.

i — игнорировать непечатаемые символы

f — интерпретирует буквы в нижнем регистре как в верхнем

M — сравнивает поля как месяца

n — сравнивает как числа

r — обратная сортировка

Также все эти "типы" могут быть указаны как обычные ключи.

Примеры использования:

sort -t : -k 7,7 /etc/passwd

Фильтр: да

uniq — находит в вводе одинаковые идущие подряд строки. В зависимости от опций выводит повторяющиеся/уникальные/все строки. При отсутствии соответствующих опций по-умолчанию выводит все строки.

Основные ключи:

-d — выводит только повторяющиеся строки

-u — выводить только уникальные строки

-c — перед каждой строкой показывать число её повторений

-f число / -число— игнорирует первые число полей (поле — последовательность символов, разделённых пустыми символами)

-s число / +число- игнорирует первые число символов.

Примеры использования:

sort myprogram.c | uniq -c > repeats.log — просматривает исходный код программы на количество повторения отдельных строк и записывает в файл.

Фильтр: да

echo — выводит на экран переданные аргументы. При их отсутствии — пустую строку (перевод строки \n). Также поддерживаются следующие последовательности символов:

\a — Alert/Bell

\b — backspace

\c — конец текста. Следующие за ним символы игнорируются

\f — form-feed

\n — перевод строки

\r — возврат каретки

\t — табуляция

\v — вертикальная табуляция

**** — обратный косая черта

Примеры использования:

clear && echo Type whatever you want && cat > text — выводит приглашение к печати некоторого текста

Фильтр: нет

alias — создаёт "псевдонимы" (сокращения) для команд или выводит их список при отсутствии аргументов в зависимости от опций.

Основные ключи:

-x — создаёт экспортируемый псевдоним/выводит список экспортируемых псевдонимов.

-t — создаёт "отслеживаемые" псевдонимы для команд на основе переменной окружения PATH.

Примеры использования:

alias goodnight='clear && sleep 1 && echo Good Night && sleep 2 && echo Sleep Tight && sleep 2 && poweroff' — выводит сообщение и завершает работу системы.

Фильтр: нет

ulimit (версия ksh)— устанавливает или показывает ограничения по размеру ресурсов, используемых интерпретатором. По-умолчанию используется опция **-f.**

Основные ключи:

-H, -S — устанавливает "жёсткие" и "мягкие" ограничения соответственно. Жёсткие ограничения нельзя увеличить, мягкие можно. Мягкое ограничение всегда должно быть меньше или равно жёсткого ограничения, иначе — ошибка. Если никакая из опций не указана, применяются обе.

-a — показывает все ограничения

-c [число] — размер дампов ядра в блоках по 512 байт

-d [число] - размер области данных (кучи) в килобайтах

-f [число] — размер записываемого файла в блоках по 512 байт

-n [число] — верхняя граница доступных файловых дескрипторов

-s [число] — размер стека в килобайтах.

-t [число] — время (процессорное) для каждого процесса в секундах

-v [число] -размер виртуальной памяти в килобайтах

Примеры использования:

ulimit -f 0 && echo SampleText > f && cat f — ничего не записывает в файл и, соответственно, ничего не выводит из него, т. к. по сути ulimit устанавливает запрет на запись в любые файлы.

Фильтр: нет

umask — создаёт маску установки прав доступа у создаваемых файлов для данного интерпретатора. Если определённый бит в маске равен 1, значит у создаваемого файла будет отключено соответствующее право доступа. При отсутствии аргументов выводит маску, применяемую в данный момент в числовом варианте.

Основные ключи:

-S — выводит символьную маску, применяемую в данный момент.

Примеры использования:

umask 377 — создаваемые файлы будут доступны только для чтения владельцем.

Фильтр: нет

Задание 3: Интерпретация команды ls -l

При введение команды ls с опцией -l просходит вывод информации о каждом указанном файле/файлах в указанной директории:

1. тип_файла — чем данный файл является: обычным файлом (-), директорией (d), символьной ссылкой (l), блочное устройство (b), символьное устройство (c), пайп (p), дверь (D), сокет (s) или порты событий (event ports, P)

директория -

жёсткая ссылка -

символьная ссылка -

2. права_доступа — права доступа к файлу различным пользователям. Разбиты на тройки:

Первая тройка: биты set uid, set gid и sticky. При установке первых двух файл будет выполняться с правами владельца группы, при установке бита sticky какой-либо файл в директории может редактировать только его владелец, владелец директории или суперпользователь.

Последние три тройки — определяют права на чтение, запись, и исполнение для владельца, группы владельца и остальные соответственно: rwxrwxrwx.

3. индикация_ACL — + при нетривиальной ACL, ассоциированной с файлом. Иначе пробел

4. количество_жёстких_ссылок — количество файлов, асоциированных с данным номером inode.

5. владелец — указывает UID владельца файла.

6. группа — указывает GID владельца файла.

7. размер_в_байтах — размер файла в байтах

8. время/время и дата последнего изменения — когда в последний раз менялось содержимое файла

Задание 4: примеры использования фильтров в конвейерах

sort phonebook1 phonebook2 -k4,4 | more

Сортирует две телефонных книжки (файлы со строками вида Фамилия Имя Отчество Номер) по номеру человека и постранично выводит на экран

who | cut -f 1 -d \ | uniq -c

Выводит список пользователей и количество псевдотерминалов, запущенных от его имени.

head -1 *.sh | tr -d '#!' | paste - - - | pg

Выводит список командных оболочек, используемых для выполнения скриптов в рабочей директории в виде ==> имя_файла <== #!путь_к_интерпретатору. Если скрипт всего один, то выводит путь интерпретатора, им используемой

sort - | join - people | tee chosenlist | split -l 10 -a 1 - chosenpart.

Выводит список выбранных и отсортированных по ID людей (подразумевается, что people - текстовый файл со строками вида "ID Фамилия Имя Отчество [дополнительная информация]") посредством ввода их ID в стандартный ввод, пишет их в файл chosenlist, а также дробит его на файлы по 10 строк. Максимальное количество "запросов" — 260.

cat part.* | cmp -l complete - | tail -10

Сравнивает ранее разделённый файл complete (файлы part.*) с его оригиналом и выводит последние побайтовых различий файлов.

diff lib_100916.c lib_110916.c | cat changelog - | tee changelog_with_code_changes | less

Создаёт список изменений библиотеки с действительными изменениями в коде и постранично выводит для просмотра.

Задание 5: Использование переменных окружения в командах

Локаль

Множество программ работают с переменными окружения, задающими *локаль*, т. е. различные лингвистические и географические параметры, которые помогают этим программам выводить в виде, понятном пользователю , будь то предпочитаемый язык, или часовой пояс. К таким переменным относятся:

LC_ALL — определяет "общую" локаль для всех переменных, т. е. подавляет значение всех остальных переменных.

LC_CTYPE — определяет классификацию символов и правила изменения между регистрами, например классы символов в **tr**

LC_NUMERIC — определяет разделители в числах (тысяч и между целой и дробной частью) и способы группировки

LC_TIME — определяет формат отображения времени

LC_COLLATE — определяет правила сортировки

LC_MONETARY — определяет формат и символы, используемые при отображении денежкой (валютной) информации

LC_MESSAGES — определяет формат и значения ответов программ

TZ — определяет часовой пояс

NLSPATH — National Language Support PATH, определяет директорию хранения "каталогов сообщений", т. е. своеобразных "локализационных баз данных", откуда программы берут различные сообщения для показа пользователю на нужном языке.

Использование переменных окружения в программах:

Следующие программы используют <u>только</u> переменные окружения, задающие локаль.

su

LANG, LC_ALL, LC_CTYPE, LC_COLLATE, LC_MESSAGES, NLSPATH, LC_NUMERIC и LC_MONETARY

sort

LANG, LC_ALL, LC_CTYPE, LC_COLLATE, LC_MESSAGES, NLSPATH и LC_NUMERIC

rm, cp, rmdir, find, join, tr, diff, umask

LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, NLSPATH и LC_COLLATE

mv, mkdir, type, file, chmod, ln, wc, tee, cat, tail, head, touch, chown, chgrp, split, paste, cut, cmp, alias, ulimit

LANG, LC_ALL, LC_CTYPE, LC_MESSAGES и NLSPATH

more

LANG, LC_ALL, LC_CTYPE, LC_MESSAGES

Остальные программы:

ls

LANG, LC_ALL, LC_COLLATE, LC_CTYPE, LC_TIME, LC_MESSAGES, NLSPATH, and TZ — см. выше пункт **Локаль**

COLUMNS — ширина столбцов, предпочитаемая пользователем. Если какую-то часть строки можно интерпретировать как целое десятичное число, то ls будет выводить столбцы именно такой ширины.

echo

LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH — см. выше пункт **Локаль**

SYSV3 — используется для предоставления совместимости с INTERACTIVE UNIX System и скриптами установки SCO UNIX (устаревшее, используется ТОЛЬКО для совместимости).

cd

CDPATH — список путей к директориям, разделённых двоеточиями. Если при выполнении команды путь оказался относительным, cd ищет среди указанных директорий подходящую, и только после этого в рабочей.

HOME — домашняя директория пользователя. В неё происходит переход, если никаких аргументов не было передано.

OLDPWD — путь предыдущей рабочей директории. Используется в "cd - ". Изменяется при переходе в другую директорию.

PWD — путь к текущей рабочей директории. Изменяется после перехода в другую директорию.

pg

LC_CTYPE, LC_MESSAGES, NLS PATH — см. выше пункт **Локаль**

COLUMNS — см. выше

LINES — количество линий для отображения на странице

SHELL — название интерпретатора командной строки

TERM — "атрибуты терминала (эмулятора терминала)", в том числе название эмулятора терминала

less

LANG, LC_CTYPE — см. выше пункт **Локаль**

COLUMNS — см. **cd**

VISUAL , EDITOR— название редактора для опции **v**, сначала проверяется переменная VISUAL, потом EDITOR.

HOME — см. **cd**

HOMEDRIVE, HOMEPATH — конкатенация этих переменных используется как путь домашней директории пользователя, если переменная HOME не определена.

LESS - список опций для **less**, применяемый при запуске программы.

LINES, SHELL, TERM — см. **page**

PATH, INIT — определяют путь для поиска файла конфигурации lesskey (устаревшее, используется только для совместимости с MS-DOS и OS/2)

LESSANSIENDCHARS, LESSANSIMIDCHARS — определяют отображение цветных символов в less

LESSBINFMT, LESSCHARDEF, LESSCHARSET, LESSUTFBINFMT — определяют отображение символов (в смысле кодировок и спецсимволов)

LESSCLOSE, LESSECHO, LESSEDIT, LESSGLOBALTAGS — определяет программы и команды для каких-либо команд **less**

LESSHISTFILE, LESSHISTSIZE — определяет название и максимальное количество команд

для сохранения в файл истории команд

LESSKEY, LESSKEY_SYSTEM — определяет названия стандартных файлов конфигурации lesskey (клавишные привязки, key bindings, для команд программы) для пользователя и системы

LESSMETACHARS, LESSMETAESCAPE — определяет метасимволы, поддерживаемые оболочкой, и escape-символ для них

LESSOPEN — команда (в терминах UNIX) для препроцессинга входных данных для **less**.

LESSSECURE — определяет запуск **less** безопасный режим

LESSSEPARATOR — определяет строку, добавляемая к имени директории при автодополнении

LESS_IS_MORE — определяет эмуляцию команду **more**

Задание 6: команда, утилита, программа

Программа — исполняемый файл (-ы), выполняющий некоторую функцию.

Утилита — программа выполняющая, системные, а не прикладные задачи, т. е. она управляет работой операционной системы или машины, а не решает каких-либо задач пользователя.

Команда — инструкция командному интерпретатору (оболочке) для исполнения. Может включать в себя несколько утилит и в целом программ. Философия UNIX — маленькие, но хорошо выполняющие свои функции программы — способствует комбинированию программ в команды таким способом, чтобы решать большие задачи.

Задание 7: семантика названия каждой программы

Большинство программ либо полностью отражают своё предназначение в названии:

Find (*поиск* файлов), **join** (аналогично команде JOIN из SQL, *соединяет* две таблицы в виде текстовых файлов по значениям какого-либо столбца), **file** (на основе тестов пытается предположить смысл содержимого *файла*: текст, скрипт и т. д.), **cut** (*вырезает* нужный столбец из файла), **paste** (*вставляет* параллельно построчно содержимое нескольких/одного файлов), **sort** (*сортирует* файл(-ы) в определённом порядке), **alias** (создать *"псевдоним"* для чего-либо), **split** (*расколоть* файл на несколько частей);

некоторые являются аббревиатурами:

Rm (*remove*, удалить файл), **cp** (*copy*, копировать файл), **rmdir** (*remove directory,* удаляет директорию), **tr** (*translate*, "перевести" текст по определённым правилам (простым, но правилам)), **diff** (*difference/differentiate*, найти различия в двух файлах), **mv** (*move*, переместить файл), **mkdir** (*make directory*, создать директорию), **chmod** (*change (permission) mode,* изменить права доступа), **ln** (*link*, создать ссылку/связать), **wc** *(word count,* посчитать слова (и не только)), **cat** (*(con)catenate*, сцеплять, соединить содержимое файлов), **chown** (*change owner(ship),* сменить владельца файла), **chgrp** (*change group*, сменить группу-владельца файла), **ls** (*list*, вывести список файлов), **cd** (*change directory*, сменить директорию), **pg** (*page/paginate*, разделить текст на страницы), **cmp** (*compare*, определяет, различаются ли файлы);

однако иногда точное назначение программ нельзя угадать и нужно пояснение:

su (*superuser,* стать суперпользователем *либо* другим пользователем),

umask — (создать "*маску*" установления прав при создании файлов), **tee** (*тройник,* разветвляет ввод на несколько потоков),

tail (посмотреть файл с *"хвоста"*, т. е. последние n строк, байт или блоков),

head (посмотреть файл с *"головы"*, т. е. первые n строк, байт или блоков),

touch (изменить состояние файла, "*прикоснувшись"* к нему, т. е. изменяя время модификации и доступа),

ulimit (устанавливает *лимит* ресурсов, доступных пользователю),

echo (вывести те же аргументы, повторить их, словно *эхо*)

more (постраничный вывод текста, при листании получаем *больше* информации)

less (выбрано как антоним к **more**, и расширяет его функционал, добавляя листание вверх)

type (определяет, как бы каждое имя, переданное как аргумент, интерпретировалось как команда)

Задание 8 и 9: Составление команд для вывода изменного текста из файла

Файл формируется командой:

```
$ echo '
the
answer
to this
code
no-one
would guess' > file
```

Задание 8: Необходимо составить команду, обрабатывающую содержимое файла file так, чтобы на стандартный поток ошибок была выведена строка "no one would guess the answer to this code".

Команда:

```
python -c "print(''.join(map(lambda x: x[:-1]+' ',(2*open('file').readlines()
[1:])[4:10])))"
```

Задание 9: Составить команду, обрабатывающую содержимое файла file так, чтобы на стандартный поток ошибок была выведена строка "answer to THAT code -- no one would guess".

Команда:

```
python -c "print(' '.join([i.rstrip('\n').replace('this',
'THAT').replace('code', 'code --') for i in open('file').readlines()[2:]]))"
```

Вывод

В ходе выполнения данной лабораторной работы мной были получены ценные знания продвинутой работы в оболочке командной строки ksh, а также по её работе. Например то, что переменные среды с префиксом LC_ задают какие-либо региональные настройки, будь то предпочитаемые язык или валюта. Также были закреплены навыки применения конвейеров в интерпретаторе командной строки, позволяющие комбинировать маленькие программы таким образом, чтобы выполнять серьёзные задачи: так, с помощью команды:

```
sort -t : -k3,3n /etc/group | cut -d : -f1,3 | tr : ' '
```

можно узнать имена групп, существующих в системе, и отсортированных по их GID.