МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Лабораторная работа №3

Факультет: РЭФ

Группа: РЭ3-91

Студент: Бабенко В.Ю.

Преподаватель: Плеханов В.А.

Дата сдачи: 21.10.2021

Отметка о защите:

Новосибирск 2021

**Задание:**

Создать скрипты для перечисленных заданий.

1. Скрипт remftrash

a. Скрипту передается один параметр – имя файла в текущем каталоге

вызова скрипта.

b. Скрипт проверяет, создан ли скрытый каталог trash в домашнем

каталоге пользователя. Если он не создан – создает его.

c. После этого скрипт создает в этом каталоге жесткую ссылку на

переданный файл с уникальным именем (например, присваивает

каждой новой ссылке имя, соответствующее следующему натуральному числу) и удаляет файл в текущем каталоге.

d. Затем в скрытый файл trash.log в домашнем каталоге

пользователя помещается запись, содержащая <имя ссылки> - полный исходный путь к удаленному файлу .

2. Скрипт unftrash

a. Скрипту передается один параметр – имя файла, который нужно

восстановить (без полного пути – только имя).

b. Скрипт по файлу trash.log должен найти все записи, содержащие в качестве имени файла переданный параметр, и выводить по одному на экран полные имена таких файлов с запросом подтверждения на восстановление.

c. Если пользователь отвечает на подтверждение положительно, то

предпринимается попытка восстановить файл по указанному полному пути (создать в соответствующем каталоге жесткую ссылку на файл из trash и удалить соответствующий файл из trash). Если каталога, указанного в полном пути к файлу, уже не существует, то файл восстанавливается в домашний каталог пользователя с выводом соответствующего сообщения.

3. Скрипт makebackup

a. Скрипт создаст в /home/user/ каталог с именем Backup-YYYYMM-

DD, где YYYY-MM-DD – дата запуска скрипта, если в

/home/user/ нет каталога с именем, соответствующим дате,

отстоящей от текущей менее чем на 7 дней. Если в /home/user/

уже есть «действующий» каталог резервного копирования

(созданный не ранее 7 дней от даты запуска скрипта), то новый

каталог не создается. Для определения текущей даты можно

воспользоваться командой date.

b. Если новый каталог был создан, то скрипт скопирует в этот каталог

все файлы из каталога /home/user/listtask (для тестирования

скрипта создайте такую директорию и набор файлов в ней). После

этого скрипт выведет в режиме дополнения в файл

/home/user/backup-report следующую информацию: строка

со сведениями о создании нового каталога с резервными копиями с

указанием его имени и даты создания; список файлов из /home/user/listtask/, которые были скопированы в этот каталог.

c. Если каталог не был создан (есть «действующий» каталог

резервного копирования), то скрипт должен скопировать в него все

файлы из /home/user/listtask/ по следующим правилам: если

файла с таким именем в каталоге резервного копирования нет, то он

копируется из /home/user/listtask. Если файл с таким именем

есть, то его размер сравнивается с размером одноименного файла в

действующем каталоге резервного копирования. Если размеры

совпадают, файл не копируется. Если размеры отличаются, то файл

копируется c автоматическим созданием версионной копии, таким

образом, в действующем каталоге резервного копирования

появляются обе версии файла (уже имеющийся файл

переименовывается путем добавления дополнительного

расширения «.YYYY-MM-DD» (дата запуска скрипта), а

скопированный сохраняет имя). После окончания копирования в

файл /home/user/backup-report выводится строка о

внесении изменений в действующий каталог резервного

копирования с указанием его имени и даты внесения изменений,

затем строки, содержащие имена добавленных файлов с новыми

именами, а затем строки с именами добавленных файлов с

существовавшими в действующем каталоге резервного

копирования именами с указанием через пробел нового имени,

присвоенного предыдущей версии этого файла.

4. Скрипт restorebackup

a. Скрипт должен скопировать в каталог /home/user/listtask/

все файлы из актуального на данный момент каталога резервного

копирования (имеющего в имени наиболее свежую дату), за исключением файлов с предыдущими версиями.

**Ход работы:**

1. Скрипт remftrash.sh

#!/bin/bash

# Лабораторная работа №3 (Задача 1)

# Если нет каталога ~/.trash, то создаем его

if ! ls -a ~ | grep -q .bashrc; then

mkdir ~/.trash

fi

# Если есть файлы в каталоге, узнаем номер последнего

name=$(ls ~/.trash | grep -oE "[[:alnum:]]" | sort -rn | head -n1)

# Если файла нет, то начинаем нумерацию с единицы. Если есть, то +1

if [[ -z "$name" ]]

then

name=1

else

let name++

fi

# Создаем жесткую ссылку

ln $1 ~/.trash/$name

# Удаляем переданный скрипту файл

rm $1

# Выводим информацию об удаленном файле в лог

echo $name - $PWD/$1 >> trash.log

#Завершаем скрипт

exit 0

1. Скрипт unftrash.sh

#!/bin/bash

# Лабораторная работа №3 (Задача 2)

# Устанавливаем счетчик

count=1

# Ищем путь полное название удаленного файла

path=$(grep "$1$" trash.log | sed -n ${count}p | cut -d ' ' -f3)

# Пока строка не станет пустой

while [ -n "$path" ]

do

# Узнаем название файла в папке .trash из trash.log

name=$(grep "$1$" trash.log | sed -n ${count}p | cut -d ' ' -f1)

echo "Вы хотите восстановить файл $path? (y/n)"

# Считываем ввод пользователя и выполняем выбранное им действие

read

case $REPLY in

y ) # Создаем жесткую ссылку

ln ~/.trash/$name $path

# Удаляем из trash.log соответсвующую восстановленному файлу запись

sed -in "/${name} - /d" trash.log

# Удаляем файл

rm ~/.trash/$name;;

n ) # Увеличиваем счетчик для перехода к следующему файлу

let count++;;

esac

# Ищем путь полное название удаленного файла

path=$(grep "$1$" trash.log | sed -n ${count}p | cut -d ' ' -f3)

done

# Завершаем скрипт

exit 0

1. Скрипт makebackup.sh

#!/bin/bash

# Лабораторная работа №3 (Задача 3)

# Ищем предыдущую папку с бэкапом

lastBackup=$(ls ~ | grep ^Backup- | sort -rn | head -1 | cut -d "-" -f 2,3,4)

if [[ -z "$lastBackup" ]]; then

lastBackup="1970-01-02"

fi

# Получаем текущую дату

currentDate=$(date +%Y-%m-%d)

# Посчитаем сколько прошло секунд с того времени

let days=($(date -d $currentDate +%s)-$(date -d $lastBackup +%s))/24/60/20

# Если число дней превышает 7 или бэкапа не существует, то создаем новый бэкап

if [[ days -gt 7 ]]

then

mkdir ~/Backup-$currentDate

cp ~/listtask/\* ~/Backup-$currentDate/

echo "Был создан новый каталог с бэкапом: Backup-$currentDate" >> ~/backup-report

echo "Были скопированы файлы:\n $(ls ~/listtask)" >> ~/backup-report

# Если время с момента последнего бэкапа не превышает 7 дней

else

cd ~/Backup-$lastBackup

echo "В каталог бэкапа Backup-$lastBackup были внесены изменения $currentDate"

echo "Были добавлены новые файлы:"

# Получаем список файлов в ~/listtask

fileList="$(ls ~/listtask)"

for filename in $fileList

do

# Если такой файл уже существует в папке бэкапа

if [[ -f "$filename" ]]

then

# Если его версия изменилась, то переименовываем старый файл в версионную копию

if [[ $(ls -l ~/listtask/$filename | cut -d ' ' -f5) -ne $(ls -l $filename | cut -d ' ' -f5) ]]

then

mv $filename $filename.$currentDate

fi

else

echo $filename

fi

# Копируем файл в папку бэкапа

cp -r ~/listtask/$filename ./

done

echo "Были добавлены новые версии файлов, старые версии которых теперь называются так:"

ls | grep -oE ".\*\.$currentDate"

fi

# Завершаем скрипт

exit 0

1. Скрипт restorebackup.sh

#!/bin/bash

# Лабораторная работа №3 (Задача 4)

lastBackup=$(ls ~ | grep ^Backup- | sort -rn | head -1 | cut -d "-" -f 2,3,4)

cd ~/Backup-$lastBackup

# Получаем список файлов в каталоге бэкапа

fileList="$(ls)"

# Перебираем файлы

for filename in $fileList

do

# Составляем список старых версионных копий

fileSkip=$(echo $filename | grep -oE ".\*\.[0-9]{4}-[0-9]{2}-[0-9]{2}" )

# Копируем только последнюю версию

if [ "$filename" != "$fileSkip" ]

then

cp -r $filename ~/listtask/

fi

done

# Завершаем скрипт

exit 0

1. Диалоговый скрипт start.sh

#!/bin/bash

# Диалоговый интерфейс для скриптов

# Используемая команда (dialog/Xdialog)

DIALOG=${DIALOG=dialog}

choice=`$DIALOG --stdout --clear --title "Список действий" \

--menu "Выберите действие" 13 50 6 \

"1" "Удалить файл" \

"2" "Восстановить файл" \

"3" "Произвести бэкап файлов" \

"4" "Произвести восстановление файлов" \

"5" "Архивировать файл в .gz" \

"6" "Разархивировать файл .gz"`

case $? in

0)

case $choice in

1) FILE=`$DIALOG --stdout --clear --title "Выберите файл для удаления" --fselect ./ 10 60`

clear -x

./remftrash.sh $FILE;;

2) filename=`$DIALOG --stdout --clear --title "Ввод данных" --clear \

--inputbox "Введите имя файла для восстановления:" 8 50`

clear -x

./unftrash.sh $filename;;

3) clear -x

./makebackup.sh;;

4) clear -x

./restorebackup.sh;;

5) FILE=`$DIALOG --stdout --clear --title "Выберите файл для архивации" --fselect ./ 10 60`

clear -x

gzip $FILE;;

6) FILE=`$DIALOG --stdout --clear --title "Выберите файл для рвзархивации" --fselect ./ 10 60`

clear -x

gunzip $FILE;:

esac;;

# Если была выбрана кнопка отмена

1) clear -x

echo "Действие отменено.";;

# Если была нажата клавиша ESC

255) clear -x

echo "Отмена.";;

esac

# Завершаем скрипт

exit 0

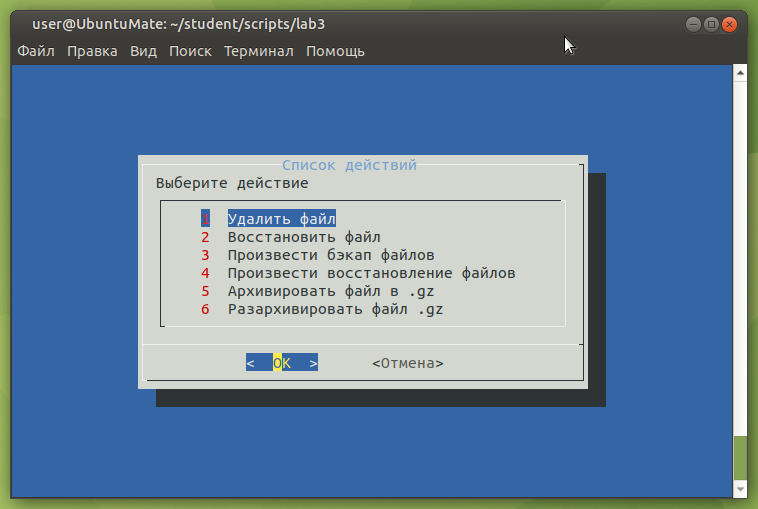


Рисунок 1 — Диалоговый скрипт

**Вывод:** в ходе лабораторной работы было создано 4 скрипта для работы с файлами и один скрипт, представляющий собой графический интерфейс для удобного взаимодействия пользователя со скриптами.