

Титульный лист

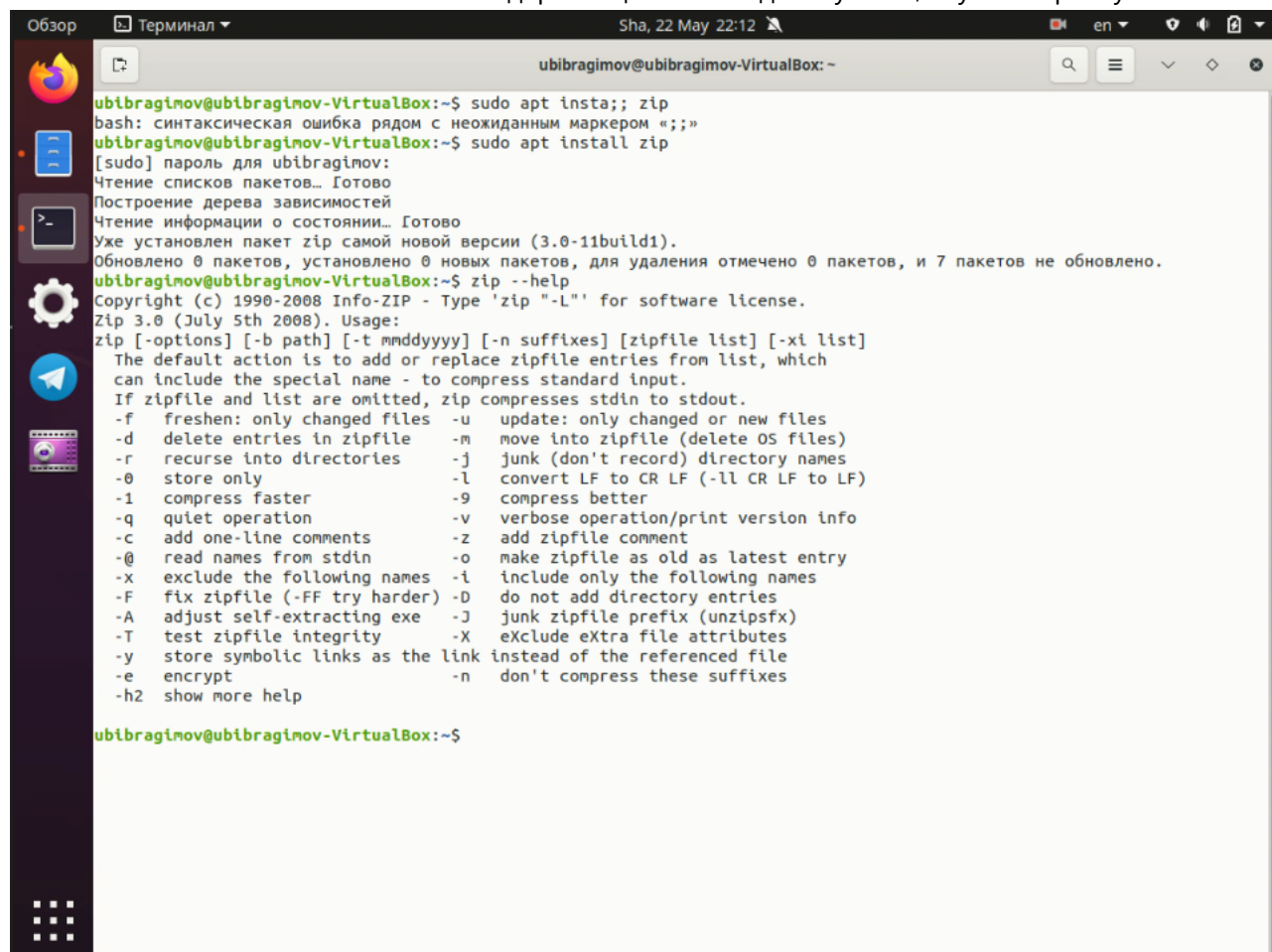
Отчёт по лабораторной работе №11 Ибрагимов Улугбек Ботырхонович

Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux, научиться писать небольшие командные файлы

Ход выполнения 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 командархивации необходимо узнать, изучив справку.



```
Обзор Терминал Sha, 22 May 22:12
ubibragimov@ubibragimov-VirtualBox: ~
ubibragimov@ubibragimov-VirtualBox:~$ sudo apt install zip
bash: синтаксическая ошибка рядом с неожиданным маркером «;»
ubibragimov@ubibragimov-VirtualBox:~$ sudo apt install zip
[sudo] пароль для ubibragimov:
Чтение списков пакетов... Готово
Построение дерева зависимостей
Чтение информации о состоянии... Готово
Уже установлен пакет zip самой новой версии (3.0-11build1).
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов, и 7 пакетов не обновлено.
ubibragimov@ubibragimov-VirtualBox:~$ zip --help
Copyright (c) 1990-2008 Info-ZIP - Type 'zip "-L"' for software license.
Zip 3.0 (July 5th 2008). Usage:
zip [-options] [-b path] [-t mmdyyy] [-n suffixes] [zipfile list] [-xi list]
The default action is to add or replace zipfile entries from list, which
can include the special name - to compress standard input.
If zipfile and list are omitted, zip compresses stdin to stdout.
-f freshen: only changed files      -u update: only changed or new files
-d delete entries in zipfile        -m move into zipfile (delete OS files)
-r recurse into directories         -j junk (don't record) directory names
-o store only                       -l convert LF to CR LF (-ll CR LF to LF)
-1 compress faster                  -9 compress better
-q quiet operation                  -v verbose operation/print version info
-c add one-line comments            -z add zipfile comment
@ read names from stdin             -o make zipfile as old as latest entry
-x exclude the following names      -i include only the following names
-F fix zipfile (-FF try harder)     -D do not add directory entries
-A adjust self-extracting exe       -J junk zipfile prefix (unzipsfx)
-T test zipfile integrity            -X eXclude eXtra file attributes
-y store symbolic links as the link -N instead of the referenced file
-e encrypt                          -n don't compress these suffixes
-h2 show more help

ubibragimov@ubibragimov-VirtualBox:~$
```

```

# /bin/bash
mkdir ~/backup_directory
cp lab11-1.sh ~/backup_directory/backup_file.sh
cd ~/backup_directory
zip -r backup.zip backup_file.sh

U:**- lab11-1.sh All L5 (Shell-script[sh])

ubibraginov@ubibraginov-VirtualBox:~/work/os$ mkdir lab11
ubibraginov@ubibraginov-VirtualBox:~/work/os$ cd ~/work/os/lab11
ubibraginov@ubibraginov-VirtualBox:~/work/os/lab11$ emacs

(emacs:18513): Gtk-WARNING **: 22:14:16.326: Theme parsing error: colors.css:71:44: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:72:44: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:74:53: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:75:53: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:76:56: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:77:65: Invalid number for color value
ubibraginov@ubibraginov-VirtualBox:~/work/os/lab11$ bash lab11-1.sh
  adding: backup_file.sh (deflated 44%)
ubibraginov@ubibraginov-VirtualBox:~/work/os/lab11$ cd ~/backup_directory
ubibraginov@ubibraginov-VirtualBox:~/backup_directory$ tree
.
├── backup_file.sh
└── backup.zip
0 directories, 2 files
ubibraginov@ubibraginov-VirtualBox:~/backup_directory$

```

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

```

emacs@ubibragimov-VirtualBox
File Edit Options Buffers Tools Sh-Script Help
Save Undo
#!/bin/bash
head -1
lab11-2.sh All L2 (Shell-script[sh])
Wrote /home/ubibragimov/work/os/lab11/lab11-2.sh

```

```

Терминал
ubibragimov@ubibragimov-VirtualBox: ~/work/os/lab11

(emacs:18513): Gtk-WARNING **: 22:14:16.326: Theme parsing error: colors.css:71:44: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:72:44: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:74:53: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:75:53: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:76:56: Invalid number for color value
(emacs:18513): Gtk-WARNING **: 22:14:16.327: Theme parsing error: colors.css:77:65: Invalid number for color value
ubibragimov@ubibragimov-VirtualBox:~/work/os/lab11$ bash lab11-1.sh
  adding: backup_file.sh (deflated 44%)
ubibragimov@ubibragimov-VirtualBox:~/work/os/lab11$ cd ~/backup_directory
ubibragimov@ubibragimov-VirtualBox:~/backup_directory$ tree
.
├── backup_file.sh
└── backup.zip
0 directories, 2 files
ubibragimov@ubibragimov-VirtualBox:~/backup_directory$ cd ~/work/os/lab11
ubibragimov@ubibragimov-VirtualBox:~/work/os/lab11$ emacs
(emacs:22656): Gtk-WARNING **: 13:02:36.040: Theme parsing error: colors.css:71:44: Invalid number for color value
(emacs:22656): Gtk-WARNING **: 13:02:36.040: Theme parsing error: colors.css:72:44: Invalid number for color value
(emacs:22656): Gtk-WARNING **: 13:02:36.040: Theme parsing error: colors.css:74:53: Invalid number for color value
(emacs:22656): Gtk-WARNING **: 13:02:36.040: Theme parsing error: colors.css:75:53: Invalid number for color value
(emacs:22656): Gtk-WARNING **: 13:02:36.040: Theme parsing error: colors.css:76:56: Invalid number for color value
(emacs:22656): Gtk-WARNING **: 13:02:36.041: Theme parsing error: colors.css:77:65: Invalid number for color value
ubibragimov@ubibragimov-VirtualBox:~/work/os/lab11$ chmod +x lab11-2.sh
ubibragimov@ubibragimov-VirtualBox:~/work/os/lab11$ ./lab11-2.sh
ssssdpddas
ssssdpddas
ubibragimov@ubibragimov-VirtualBox:~/work/os/lab11$ ./lab11-2.sh
HELLO WORLD, MY NAME IS ULUGBEK
HELLO WORLD, MY NAME IS ULUGBEK
ubibragimov@ubibragimov-VirtualBox:~/work/os/lab11$

```

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

The top screenshot shows the Emacs editor with a shell script named `lab11-3.sh`. The script checks if a file is a directory, file, readable, writable, or executable, and echoes the result.

```
#!/bin/bash
for something in *
do if test -d $something
then echo $something - directory
else echo -n $something - file, which we can search:
if test -r $something
then echo read
else echo -
fi
if test -w $something
then echo write
else echo -
fi
if test -x $something
then echo execute
else echo -
fi
fi
done
```

The bottom screenshot shows a terminal window where the script is being executed. It shows the script being made executable with `chmod +x lab11-2.sh` and then run with `./lab11-2.sh`, which outputs `HELLO WORLD, MY NAME IS ULUGBEK`. The terminal also shows the script being run with `./lab11-3.sh`, which outputs the results of the file checks for `lab11-1.sh`, `lab11-2.sh`, and `lab11-3.sh`.

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

The screenshot shows a Linux desktop environment. On the left is a vertical dock with icons for Firefox, a file manager, a terminal, a settings gear, a Telegram icon, a music player, and a purple application icon. The top panel displays the time '06:30p', the application 'Emacs (GUI)', and the system date/time 'Ju, 28 May 13:13'. The main workspace contains two windows. The foreground window is titled 'emacs@ubibragimov-VirtualBox' and shows a shell script in a text editor. The script defines variables for 'type' and 'directory', prompts the user for input, and uses 'find' commands to count files. The background window is titled 'ubibragimov@ubibragimov-VirtualBox: ~/work/os/lab11' and displays a list of files with their sizes and permissions. The terminal window at the bottom shows the command prompt 'U:***-' followed by 'lab11-4.sh' and 'All L12 (Shell-script[sh])'.

```
#!/bin/bash
type=""
directory=""
echo "cin type"
read type
echo "cin directory"
read directory
echo "number of unit:"
find "$directory" -type f -name "*.$type" | wc -l
echo "unit:"
find "$directory" -type f -name "*.$type" < ls
```

U:***- lab11-4.sh All L12 (Shell-script[sh])

6 / 8

Вывод

В ходе выполнения Лабораторной работы №11, были приобретены навыки по языку **bash**, а также изучил возможности которые дает этот язык в повседневном использовании

Контрольные вопросы

1. Командная строка - промежуточный слой связи между пользователем и операционной системой, пользователь дает команды операционной системе через командную строку, а операционная система в свою очередь дает ответ пользователю через командную строку. Пример: POSIX, bash, zsh, cmd
2. POSIX - стандартизированный интерфейс оболочки между пользователем и операционной системой. В теории, абсолютно все дистрибутивы выполненные на основе *nix, обладают возможностью кроссплатформенного исполнения инструкций, например инструкция для Ubuntu, MacOS, Solaris должна выглядеть одинаково при использовании гайдлайнов POSIX, отдельно все встраиваемые операционные системы обладают стандартом **POSIX Ready**, что означает почти 99% (кроме специальных инструкций, например из-за различных архитектур ОС) поддерживаемость и на других дистрибутивах с такой пометкой
3. В **bash** могут быть определены одномерные массивы - переменные специального вида, доступ к которым осуществляется с одним именем, но с разным индексом. Для управления используются команды: **name[%n] = value**
4. Оператор **let** - используется для суммирования Оператор **read** - считывает строку из стандартного ввода и разбивает на слова
5. Арифметические операции в **bash**:
 - **+** - сложение
 - **-** - вычитание
 - ***** - умножение
 - **/** - деление
 - **%** - вычисление остатка
 - ****** - возведение в степень
 - **+=** - декремент
 - **--** - инкремент
 - ***=** - умножение на заданное число
 - **/=** - деление на заданное число
6. **(())** - запись условия **if-else** в оболочке **bash**
7. Стандартные переменные:
 - **HOME** - домашний каталог пользователя
 - **CDPATH** - список каталогов разделенных двоеточиями и используемые при поиске пути встроеной командой **cd**
 - **IFS** - символы, с помощью которых разделяются поля
 - **MAIL** - если не задан **MAILPATH**, то пользователь будет информирован
 - **MAILPATH** - список имен файлов, которые ОС периодически проверяет на наличие письма
 - **PATH** - список каталогов, в которых командная оболочка ищет команды

8. ' < > * ? | \ " & - метасимволы, которые значительно упрощают использование командной оболочки путем написания регулярных выражений
9. Экранирование осуществляется с помощью обратного слеша \, для выражений используются двойные кавычки, а для экранирования группы метасимволов используют одинарные кавычки
10. Создаем текстовый файл с необходимой кодировкой, открываем терминал, переходим в директорию с файлом, и исполняем команду: `bash my_command [arguments]`, вводим необходимые аргументы после названия нашего файла и нажимаем Enter, так же можно изменить доступ и дать права `sudo chmod u+x my_command` и после изменения доступа можно вызывать командный файл без использования слова `bash`
11. Для создания функций используется ключевое слово `function`, после которого можно написать команды которые при исполнении файла будут исполнены в одном блоке `function`
12. Заходим в терминал, пишем `ls -lrt`, смотрим вывод, если в выводе есть символ `d`, то значит что искомый нами файл является директорией (каталогом или папкой)
13. Команда `set` - выводит список переменных окружений Команда `typeset` - ограничивает переменные (изменяет доступ, числа, массив или вообще осуществляет экспорт данных) Команда `unset` - удаляет переменную, а на самом деле устанавливает ее значение `null`
14. В командный файл можно передать от одного до девяти файлов
15.
 - `$*` — отображается вся командная строка или параметры оболочки
 - `$?` — код завершения последней выполненной команды
 - `$$` — уникальный идентификатор процесса, в рамках которого выполняется командный процессор
 - `$!` — номер процесса, в рамках которого выполняется последняя вызванная на выполнение в командном режиме команда
 - `$-` — значение флагов командного процессора
 - `${#*}` — возвращает целое число — количество слов, которые были результатом `$`
 - `${#name}` — возвращает целое значение длины строки в переменной `name`
 - `${name[n]}` — обращение к `n-му` элементу массива
 - `${name[*]}` — перечисляет все элементы массива, разделённые пробелом
 - `${name[@]}` — то же самое, но позволяет учитывать символы пробелы в самих переменных
 - `${name:-value}` — если значение переменной `name` не определено, то оно будет заменено на указанное `value`
 - `${name:value}` — проверяется факт существования переменной
 - `${name=value}` — если `name` не определено, то ему присваивается значение `value`
 - `${name?value}` — останавливает выполнение, если имя переменной не определено, и выводит `value` как сообщение об ошибке
 - `${name+value}` — это выражение работает противоположно `${name-value}`. Если переменная определена, то подставляется `value`
 - `${name#pattern}` — представляет значение переменной `name` с удалённым самым коротким левым образцом (`pattern`)
 - `${#name[*]}` и `${#name[@]}` — эти выражения возвращают количество элементов в массиве `name`