

Лабораторная работа № 12. Программирование в командном процессоре ОС UNIX. Командные файлы

Отчёт

Сергеев Д. О.

26 апреля 2025

Российский университет дружбы народов, Москва, Россия

Информация

- Сергеев Даниил Олегович
- Студент
- Направление: Прикладная информатика
- Российский университет дружбы народов
- 1132246837@pfur.ru

Цель работы

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

Задание

Написать скрипты для задач лабораторной работы

Ход выполнения лабораторной работы

Перейдем в каталог lab07 и создадим файл first.sh, открыв emacs в фоновом режиме.

```
[dosergeev@vbox ~]$ cd lab07/  
[dosergeev@vbox lab07]$ ls  
[dosergeev@vbox lab07]$ emacs first.sh &  
[1] 6726  
[dosergeev@vbox lab07]$
```

Рис. 1: Создание первого скрипта.

Напишем код для первого скрипта. Он будет создавать резервную копию самого себя в каталоге ~/backup. Созданный файл будет архивироваться с помощью tar. Проверим работу командного файла.

```
backup=$HOME/backup
function archive {
    mkdir -p $backup
    tar -cf $backup/copy.tar first.sh
}
if [ -d $backup ]
then
    rm -r $backup
    archive
else
    archive
fi
```

Рис. 2: Код первого скрипта

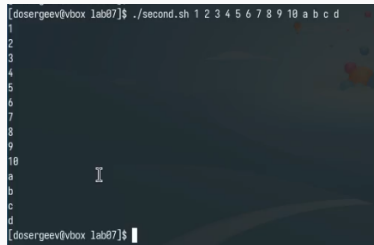
```
[dosergeev@vbox lab07]$ chmod a+x first.sh
[dosergeev@vbox lab07]$ ./first.sh
[dosergeev@vbox lab07]$ ls ~/backup/
copy.tar
[dosergeev@vbox lab07]$
```

Рис. 3: Проверка первого скрипта

Теперь напишем командный файл для распечатывания значений всех переданных ему аргументов. Проверим его работу, обязательно на 11 и более аргументах.

```
while (( $#>0 ))  
do  
    echo $1  
    shift  
done
```

Рис. 4: Код второго скрипта



A terminal window showing the execution of a script named `./second.sh`. The command prompt is `[dosergeev@vbox lab07]$`. The script is run with 11 arguments: `1 2 3 4 5 6 7 8 9 10 a b c d`. The output shows the first argument `1` on the first line, followed by lines 2 through 10, and then the remaining arguments `a`, `b`, `c`, and `d` on lines 11 through 14. The prompt is `[dosergeev@vbox lab07]$`.

Рис. 5: Проверка второго скрипта

Выполнение упражнений

Следующий скрипт станет аналогом ls - он будет выдавать информацию о нужном каталоге и выводить информацию о возможностях доступа к его файлам. Проверим его работу на домашнем каталоге. Укажем путь к нему в качестве аргумента.

```
if (($!=0))
then
  while (($#>0))
  do
    if [ -d $1 ]
    then
      echo "$(stat -t -c "%A %s %U %G %y %n" $1)"
      echo "Permission User Group Filename"
      for file in $1/*
      do
        echo "$(stat -t -c "%A %U %G %n" $file)"
      done
      shift
    elif [ -f $1 ]
    then
      echo "Permission User Group Filename"
      echo "$(stat -t -c "%A %U %G %n" $1)"
      shift
    else
      echo "$1 file not found"
      shift
    fi
  done
else
  echo "$(stat -t -c "%A %s %U %G %y %n" $(pwd))"
  echo "Permission User Group Filename"
  for file in *
  do
    echo "$(stat -t -c "%A %U %G %n" $file)"
  done
fi
```

Рис. 6: Код третьего скрипта

```
[dosergeev@vbox lab07]$ ./third.sh ~/
drwx----- 1442 dosergeev dosergeev 2025-05-03 20:59:59.979325465 +0300
/home/dosergeev/
Permission User Group Filename
-rw-r--r-- dosergeev dosergeev /home/dosergeev/abc1
drwxr-xr-x dosergeev dosergeev /home/dosergeev/backup
drwxr-xr-x dosergeev dosergeev /home/dosergeev/bin
-rw-r--r-- dosergeev dosergeev /home/dosergeev/conf.txt
drwxr-xr-x dosergeev dosergeev /home/dosergeev/Downloads
-rw-r--r-- dosergeev dosergeev /home/dosergeev/file.txt
drwxr-xr-x dosergeev dosergeev /home/dosergeev/lab07
-rw-r--r-- dosergeev dosergeev /home/dosergeev/LICENSE
-rw-r--r-- dosergeev dosergeev /home/dosergeev/may
-rw-r--r-- dosergeev dosergeev /home/dosergeev/package.json
drwxr-xr-x dosergeev dosergeev /home/dosergeev/reports
drwxr-xr-x dosergeev dosergeev /home/dosergeev/work
drwxr-xr-x dosergeev dosergeev /home/dosergeev/Видео
drwxr-xr-x dosergeev dosergeev /home/dosergeev/Документы
drwxr-xr-x dosergeev dosergeev /home/dosergeev/Загрузки
drwxr-xr-x dosergeev dosergeev /home/dosergeev/Изображения
drwxr-xr-x dosergeev dosergeev /home/dosergeev/Музыка
drwxr-xr-x dosergeev dosergeev /home/dosergeev/Общедоступные
```

Рис. 7: Проверка третьего скрипта

Выполнение упражнений

Последний скрипт будет выводить все файлы указанного формата (.txt, .doc, .jpg, .pdf) в заданной директории. Для этого попробуем использовать команду `getopts`. Будем принимать директорию и формат файла в качестве аргумента ключей `-d` и `-f`. Проверим его работу на домашнем каталоге. Укажем путь `~/` и формат `.txt`

```
workingdir=$(pwd)
fileformat=".txt"
while getopts d:f: optletter
do
    case $optletter in
        d)
            if [ -d $OPTARG ]
            then
                workingdir=$OPTARG
            else
                echo Directory not found. Using current directory
            fi
            ;;
        f)
            for format in txt doc jpg pdf
            do
                if [ "$OPTARG" == "$format" ]
                then
                    fileformat=$format
                    break
                fi
            done
            ;;
        *)
            echo Incorrect arguments
        esac
done
let counter=0
for file in $workingdir/*
do
    if [ "${file##*.}" == "$fileformat" ]
    then
        let counter+=1
    fi
done
echo $counter found
```

Рис. 8: Код четвертого скрипта

```
[dosergeev@vbox lab07]$ ./fourth.sh -f txt -d ~/
2 found
[dosergeev@vbox lab07]$ ls ~/
abc1      LICENSE      Загрузки
backup    may          Изображения
bin       package.json Музыка
conf.txt  reports      Общедоступные
Downloads work          'Рабочий стол'
file.txt  Видео       Шаблоны
lab07     Документы
```

Рис. 9: Проверка четвертого скрипта

1. Командная оболочка - это программа, позволяющая пользователю взаимодействовать с операционной системой. Примеры:
 - Bourne shell (sh)
 - C shell (csh)
 - Korn shell (ksh)
 - BASH (Bourne Again Shell) Основные отличия: синтаксис, функциональность, совместимость с POSIX.
2. POSIX (Portable Operating System Interface) - набор стандартов, описывающих интерфейсы взаимодействия ОС и прикладных программ для обеспечения совместимости UNIX-подобных систем.

3. В bash:

- Переменные: `name=value`
- Массивы: `set -A array_name value1 value2 "value 3"` или `array_name[index]=value`

4. Операторы:

- **let** - вычисление арифметических выражений
- **read** - чтение значений переменных со стандартного ввода

5. Можно применять данные арифметические операции в bash: + (сложение), - (вычитание), * (умножение), / (деление), % (остаток от деления), битовые операции и операции сравнения.

6. Операция `(())` используется для вычисления арифметических выражений и возвращает статус 0 (истина), если результат не нулевой.

7. Стандартные переменные:

- `PATH` - пути поиска команд
- `PS1`, `PS2` - промптеры
- `HOME` - домашний каталог
- `IFS` - разделители полей
- `MAIL` - файл почты
- `TERM` - тип терминала
- `LOGNAME` - имя пользователя

8. Метасимволы - символы с особым значением в командной оболочке: ' < > * ? | \ " & и другие.
9. Экранирование метасимволов:
- Перед символом: *
 - Группа в одинарных кавычках: '*\|*'
 - В двойных кавычках (кроме \$, ', \, ")

- 10.
- Создать текстовый файл с командами
 - Дать права на выполнение: `chmod +x filename`
 - Запустить: `./filename` или `bash filename`
11. Функции определяются так: `bash function name { команды }`
Удаление: `unset -f name`

12. Проверить тип файла можно командой `test`:

- `test -f file` - обычный файл
- `test -d file` - каталог Или: `[-f file], [-d file]`

13. Назначение команд:

- `set` - установка параметров оболочки/вывод переменных
- `typeset` (или `declare`) - задание типа переменной
- `unset` - удаление переменной/функции

14. Параметры в командные файлы передаются как аргументы при вызове и доступны через `$1`, `$2`, ..., `$9`, `$0` - имя скрипта.

15. Специальные переменные `bash`:

- `$*`, `$@` - все параметры
- `$#` - количество параметров
- `$?` - код завершения последней команды
- `$$` - PID текущего процесса
- `$!` - PID последнего фонового процесса
- `$-` - флаги оболочки
- `${#*}` - количество слов в `$*`
- `${#name}` - длина строки переменной

Вывод

В результате выполнения лабораторной работы я изучил основы программирования в оболочке ОС UNIX/Linux и научился писать небольшие командные файлы.