

# **Отчёт по лабораторной работе №12**

**Операционные системы**

Бережной Иван Александрович

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Теоретическое введение</b>	<b>7</b>
<b>4</b>	<b>Выполнение лабораторной работы</b>	<b>9</b>
4.1	Написание скрипта, создающего резервную копию самого себя . . . . .	9
4.2	Написание командного файла, выводящего в терминал введённые строки . . . . .	10
4.3	Написание аналога команды ls . . . . .	11
4.4	Написание командного файла, вычисляющего количество файлов с указанным расширением . . . . .	12
<b>5</b>	<b>Выводы</b>	<b>14</b>
	<b>Список литературы</b>	<b>15</b>

# Список иллюстраций

4.1	Создание первого файла . . . . .	9
4.2	Написание первого файла . . . . .	9
4.3	Создание второго файла . . . . .	10
4.4	Написание второго файла . . . . .	10
4.5	Результат работы второго файла . . . . .	10
4.6	Создание третьего файла . . . . .	11
4.7	Написание третьего файла . . . . .	11
4.8	Результат работы третьего файла . . . . .	12
4.9	Создание четвёртого файла . . . . .	13
4.10	Написание четвёртого файла . . . . .	13
4.11	Результат работы четвёртого файла . . . . .	13

## **Список таблиц**

# 1 Цель работы

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

## 2 Задание

1. Написать скрипт, создающий резервную копию самого себя
2. Написать командный файл, выводящий в терминал введённые строки
3. Написать аналог команды `ls`
4. Написать командный файл, вычисляющий количество файлов с указанным расширением

### 3 Теоретическое введение

Командный процессор (командная оболочка, интерпретатор команд shell) — 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; – С-оболочка (или csh) — надстройка на оболочкой Борна, использующая С-подобный синтаксис команд с возможностью сохранения истории выполнения команд; – оболочка Корна (или ksh) — напоминает оболочку С, но операторы управления программой совместимы с операторами оболочки Борна; – BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек С и Корна (разработка компании Free Software Foundation). POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ. Стандарты POSIX разработаны комитетом IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости различных UNIX/Linux-подобных операционных систем и переносимости прикладных программ на уровне исходного кода. POSIX-совместимые оболочки разработаны на базе оболочки

Корна.



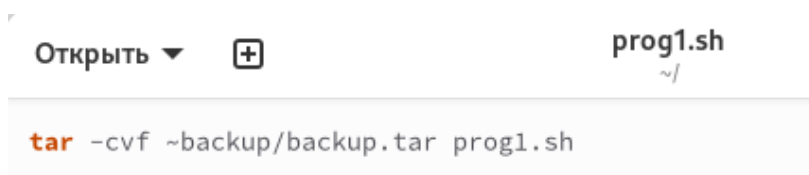
## 4 Выполнение лабораторной работы

### 4.1 Написание скрипта, создающего резервную копию самого себя

Для начала создадим файл и дадим ему разрешение на исполнение (рис. 4.1). Откроем его и пропишем логику выполнения (рис. 4.2). В результате выполнения мы получим архив с копией нужного нам файла.

```
iaberezhnoy@iaberezhnoy:~$ touch prog1.sh  
iaberezhnoy@iaberezhnoy:~$ chmod +x prog1.sh
```

Рис. 4.1: Создание первого файла



The image shows a file manager interface. At the top, there is a toolbar with a button labeled 'Открыть' (Open) and a plus icon. To the right, the file 'prog1.sh' is listed with a tilde icon below it. Below the toolbar, a terminal window is open, displaying the command `tar -cvf ~backup/backup.tar prog1.sh`.

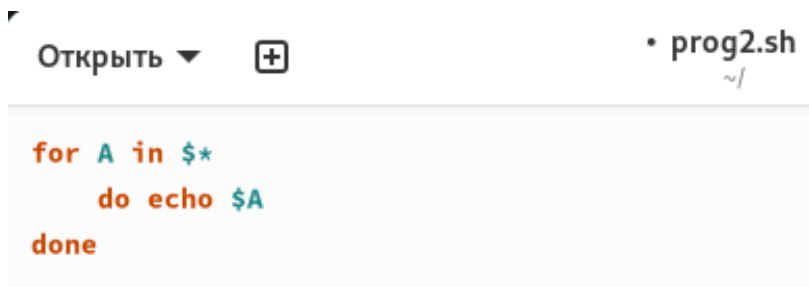
Рис. 4.2: Написание первого файла

## 4.2 Написание командного файла, выводящего в терминал введённые строки

Создадим новый файл (рис. 4.3) и напишем скрипт (рис. 4.4) - цикл, который проходит по каждому введённому значению и выводит его на экран (рис. 4.5).

```
iaberezhnoy@iaberezhnoy:~$ touch prog2.sh
iaberezhnoy@iaberezhnoy:~$ chmod +x prog2.sh
```

Рис. 4.3: Создание второго файл

A screenshot of a code editor window. At the top, there is a toolbar with a dropdown menu labeled 'Открыть' (Open) and a plus icon. To the right of the toolbar, the file name 'prog2.sh' is displayed with a tilde icon below it. The main area of the editor contains a shell script with the following lines: 'for A in \$\*', 'do echo \$A', and 'done'. The text is color-coded: 'for' is blue, 'A' is red, 'in' is blue, '\$\*' is red, 'do' is blue, 'echo' is red, '\$A' is red, and 'done' is blue.

```
for A in $*
do echo $A
done
```

Рис. 4.4: Написание второго файла

```
iaberezhnoy@iaberezhnoy:~$ bash prog2.sh ooo pp 1234
ooo
pp
1234
```

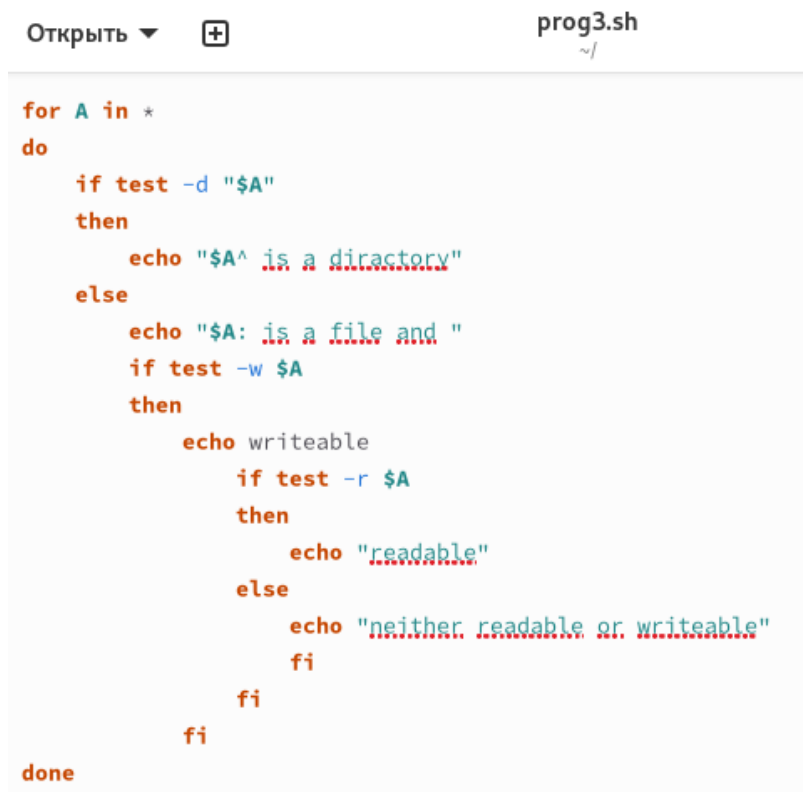
Рис. 4.5: Результат работы второго файла

## 4.3 Написание аналога команды ls

Снова создаём файл (рис. 4.6) и пишем в нём скрипт (рис. 4.7), который будет давать информацию о нужном каталоге и выводить информацию о возможностях доступа к его файлам (рис. 4.8).

```
iaberezhnoy@iaberezhnoy:~$ touch prog3.sh
iaberezhnoy@iaberezhnoy:~$ chmod +x prog3.sh
```

Рис. 4.6: Создание третьего файла



```
Открыть ▼  +  prog3.sh
~/

for A in *
do
    if test -d "$A"
    then
        echo "$A^ is a diractory"
    else
        echo "$A: is a file and "
        if test -w $A
        then
            echo writeable
            if test -r $A
            then
                echo "readable"
            else
                echo "neither readable or writeable"
            fi
        fi
    fi
done
```

Рис. 4.7: Написание третьего файла

```
iaberezhnoy@iaberezhnoy:~$ bash prog3.sh
bin^ is a diractory
prog1.sh: is a file and
writeable
readable
prog2.sh: is a file and
writeable
readable
prog3.sh: is a file and
writeable
readable
work^ is a diractory
Видео^ is a diractory
Документы^ is a diractory
Загрузки^ is a diractory
Изображения^ is a diractory
Музыка^ is a diractory
Общедоступные^ is a diractory
Рабочий стол^ is a diractory
Шаблоны^ is a diractory
```

Рис. 4.8: Результат работы третьего файла

## 4.4 Написание командного файла, вычисляющего количество файлов с указанным расширением

Создадим последний на сегодня файл (рис. 4.9) и напишем логику (рис. 4.10). Он будет вычислять количество файлов с указанным расширением внутри указанного каталога (рис. 4.11).

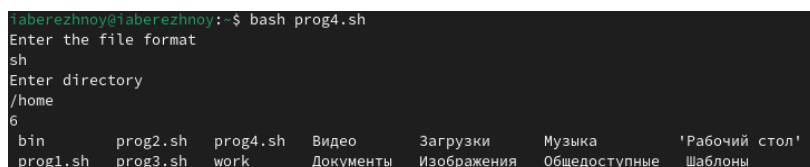
```
iaberezhnoy@iaberezhnoy:~$ touch prog4.sh
iaberezhnoy@iaberezhnoy:~$ chmod +x prog4.sh
```

Рис. 4.9: Создание четвёртого файла



```
format=""
directory=""
echo "Enter the file format"
read format
echo "Enter directory"
read directory
find "${directory}" -name "*.${format}" -type f | wc -l
ls
```

Рис. 4.10: Написание четвёртого файла



```
iaberezhnoy@iaberezhnoy:~$ bash prog4.sh
Enter the file format
sh
Enter directory
/home
6
bin      prog2.sh  prog4.sh  Видео    Загрузки  Музыка    'Рабочий стол'
prog1.sh prog3.sh  work      Документы Изображения Общедоступные Шаблоны
```

Рис. 4.11: Результат работы четвёртого файла

## 5 Выводы

В ходе выполнения лабораторной работы мы изучили основы программирования в оболочке ОС UNIX/Linux и научились писать небольшие командные файлы.

## **Список литературы**