

# **Лабораторная работа №12**

**Программирование в командном процессоре ОС UNIX. Командные  
файлы**

Кижваткина Анна Юрьевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Выполнение лабораторной работы</b>	<b>6</b>
<b>3</b>	<b>Выводы</b>	<b>11</b>

# Список иллюстраций

2.1	Создание директории . . . . .	6
2.2	Создание файла . . . . .	6
2.3	Программа . . . . .	6
2.4	Предоставление доступа . . . . .	7
2.5	Запуск программы . . . . .	7
2.6	Создание файла . . . . .	7
2.7	Предоставление доступа . . . . .	7
2.8	Программа . . . . .	7
2.9	Запуск программы . . . . .	8
2.10	Создание файла . . . . .	8
2.11	Предоставление доступа . . . . .	8
2.12	Программа . . . . .	8
2.13	Запуск программы . . . . .	9
2.14	Создание файла . . . . .	9
2.15	Предоставление доступа . . . . .	9
2.16	Программа . . . . .	9
2.17	Запуск программы . . . . .	10

## **Список таблиц**

# 1 Цель работы

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

## 2 Выполнение лабораторной работы

Создаем директорию backup. (рис. 2.1)

```
[aykizhvatkina@vbox ~]$ mkdir backup
```

Рис. 2.1: Создание директории

Создаем файл task1.sh. (рис. 2.2)

```
[aykizhvatkina@vbox ~]$ touch task1.sh
```

Рис. 2.2: Создание файла

Пишем 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 команд архивации необходимо узнать, изучив справку. (рис. 2.3)

```
GNU nano 8.1 task1.sh Изменён
tar -cvf ~/backup/1.tar $0
```

Рис. 2.3: Программа

Предоставляем полный доступ (чтение, запись и выполнение) к файлу или каталогу для всех пользователей. (рис. 2.4)

```
[aykizhvatkina@vbox ~]$ chmod 777 task1.sh
```

Рис. 2.4: Предоставление доступа

Проверяем выполнение программы. (рис. 2.5)

```
[aykizhvatkina@vbox ~]$ ./task1.sh  
./task1.sh
```

Рис. 2.5: Запуск программы

Создаем файл task2.sh. (рис. 2.6)

```
[aykizhvatkina@vbox ~]$ touch task2.sh
```

Рис. 2.6: Создание файла

Предоставляем полный доступ (чтение, запись и выполнение) к файлу или каталогу для всех пользователей. (рис. 2.7)

```
[aykizhvatkina@vbox ~]$ chmod 777 task2.sh
```

Рис. 2.7: Предоставление доступа

Пишем пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять. Например, скрипт может последовательно распечатывать значения всех переданных аргументов. (рис. 2.8)

```
GNU nano 8.1      task2.sh      Изменён  
for i in "$@"  
do echo ${i}  
done
```

Рис. 2.8: Программа

Проверяем выполнение программы. (рис. 2.9)

```
[aykizhvatkina@vbox ~]$ ./task2.sh jhg jhgdfu isufj hucbis "jhbvis"
jhg
jhgdfu
isufj
hucbis
jhbvis
```

Рис. 2.9: Запуск программы

Создаем файл task3.sh. (рис. 2.10)

```
[aykizhvatkina@vbox ~]$ touch task3.sh
```

Рис. 2.10: Создание файла

Предоставляем полный доступ (чтение, запись и выполнение) к файлу или каталогу для всех пользователей. (рис. 2.11)

```
[aykizhvatkina@vbox ~]$ chmod 777 task3.sh
```

Рис. 2.11: Предоставление доступа

Пишем 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. (рис. 2.12)

```
GNU nano 8.1 task3.sh
echo "$1/ " | tr -d "\n";
stat --printf "%A" "$1/";
echo
for i in $1/*
do echo "${i} " | tr -d "\n";
stat --printf "%A" "${i}";
echo
done
```

Рис. 2.12: Программа

Проверяем выполнение программы. (рис. 2.13)



```
[aykizhvatkina@vbox ~]$ ./task3.sh ~
/home/aykizhvatkina/ drwx-----
/home/aykizhvatkina/#1# -rw-r--r--
/home/aykizhvatkina/#2# -rw-r--r--
/home/aykizhvatkina/#3# -rw-r--r--
/home/aykizhvatkina/4 -rw-r--r--
/home/aykizhvatkina/abc1 -rw-rw-r--
/home/aykizhvatkina/australia drwxr-xr-x
/home/aykizhvatkina/backup drwxr-xr-x
```

Рис. 2.13: Запуск программы

Создаем файл task4.sh. (рис. 2.14)

```
[aykizhvatkina@vbox ~]$ touch task4.sh
```

Рис. 2.14: Создание файла

Предоставляем полный доступ (чтение, запись и выполнение) к файлу или каталогу для всех пользователей. (рис. 2.15)

```
[aykizhvatkina@vbox ~]$ chmod 777 task4.sh
```

Рис. 2.15: Предоставление доступа

Пишем 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 директории. Путь к директории также передаётся в виде аргумента командной строки. (рис. 2.16)

```
GNU nano 8.1 task4.sh
let COUNT=0
for i in $2/*. $1
do let COUNT++
done
echo $COUNT
```

Рис. 2.16: Программа

Проверяем выполнение программы. (рис. 2.17)

```
[aykizhvatkina@vbox ~]$ ./task4.sh txt ~  
5
```

Рис. 2.17: Запуск программы

## **3 Выводы**

В ходе данной лабораторной работы мы изучили основы программирования в оболочке ОС UNIX/Linux. Научились писать небольшие командные файлы.