

# **Лабораторная работа № 10**

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

Юсупова Ксения Равилевна

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>5</b>
<b>2</b>	<b>Задание</b>	<b>6</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>7</b>
<b>4</b>	<b>Ответы на контрольные вопросы</b>	<b>11</b>
<b>5</b>	<b>Вывод</b>	<b>13</b>

## Список иллюстраций

3.1	код для первой программы . . . . .	7
3.2	проверили первый код . . . . .	7
3.3	код для второй программы . . . . .	8
3.4	Проверили код на работу . . . . .	8
3.5	код для третьей программы . . . . .	8
3.6	Проверили код на работу . . . . .	9
3.7	код для четвертой программы . . . . .	9
3.8	Проверили код на работу . . . . .	10

## **Список таблиц**

# 1 Цель работы

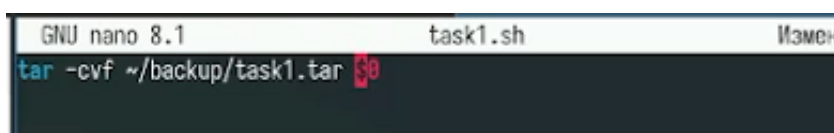
Изучить основы программирования в оболочке ОС UNIX/Linux. Научиться писать небольшие командные файлы.

## 2 Задание

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 команд архивации необходимо узнать, изучив справку.
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.
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.
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 директории. Путь к директории также передаётся в виде аргумента командной строки.

### 3 Выполнение лабораторной работы

Напишем код для первой программы (Написать скрипт, который при запуске будет делать резервную копию самого себя в другую директорию backup в вашем домашнем каталоге)(рис. 3.1).



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

Рис. 3.1: код для первой программы

Проверили код на работу (рис. 3.2).

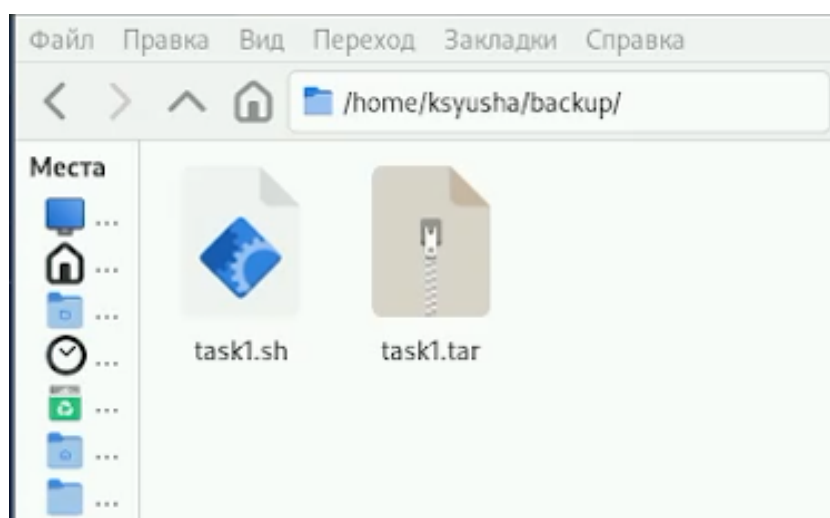


Рис. 3.2: проверили первый код

Напишем код для второй программы (пример командного файла, обрабатывающего любое произвольное число аргументов командной строки, в том числе превышающее десять)(рис. 3.3).

```

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

```

Рис. 3.3: код для второй программы

Проверили код на работу (рис. 3.4).

```

[ksyusha@ksyusha ~]$ ./task2.sh hi ben
hi
ben
[ksyusha@ksyusha ~]$ ./task2.sh qwert ghjkl bnmpk
qwert
ghjkl
bnmpk

```

Рис. 3.4: Проверили код на работу

Написали код для третьей программы (Написать командный файл — аналог команды ls (без использования самой этой команды и команды dir).)(рис. 3.5).

```

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" "$1/";
echo
done

```

Рис. 3.5: код для третьей программы

Проверили код на работу (рис. 3.6).



```

[ksyusha@ksyusha ~]$ ./task3.sh ~
/home/ksyusha/ drwx-----
/home/ksyusha/#10# drwx-----
/home/ksyusha/#11# drwx-----
/home/ksyusha/#12# drwx-----
/home/ksyusha/#14# drwx-----
/home/ksyusha/abc1 drwx-----
/home/ksyusha/ao2 drwx-----
/home/ksyusha/australia drwx-----
/home/ksyusha/backup drwx-----
/home/ksyusha/bin drwx-----
/home/ksyusha/blog drwx-----
/home/ksyusha/conf.txt drwx-----
/home/ksyusha/Desktop drwx-----
/home/ksyusha/Documents drwx-----
/home/ksyusha/Downloads drwx-----
/home/ksyusha/feathers drwx-----
/home/ksyusha/file.old drwx-----
/home/ksyusha/file.txt drwx-----
/home/ksyusha/git-extended drwx-----
/home/ksyusha/#lab07.sh# drwx-----
/home/ksyusha/lab07.sh drwx-----
/home/ksyusha/LICENSE drwx-----
/home/ksyusha/may drwx-----
/home/ksyusha/monthly drwx-----
/home/ksyusha/my_os drwx-----

```

Рис. 3.6: Проверили код на работу

Написали код для четвертой программы (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 директории) (рис. 3.7).

```

GNU nano 8.1 task4.sh
let COUNT=0
for i in $(ls *. $1)
do let COUNT++
done
echo $COUNT

```

Рис. 3.7: код для четвертой программы

Проверили код на работу (рис. 3.8).

```
[ksyusha@ksyusha ~]$ ./task4.sh txt ~  
3
```

Рис. 3.8: Проверили код на работу

## 4 Ответы на контрольные вопросы

1. Командная оболочка (shell) - программа-интерпретатор для взаимодействия пользователя с ядром ОС. Примеры: bash (стандартная), zsh (с автодополнением), fish (интерактивная), dash (легковесная). Отличаются синтаксисом, функционалом и скоростью работы.
2. POSIX - стандарт для совместимости UNIX-систем, унификации API и командных интерфейсов.
3. В bash переменные: var="значение". Массивы: arr=("эл1" "эл2") - индексированный, declare -A dict=([ "кл" ]="зн") - ассоциативный.
4. let - для арифметики: let "sum=5+5". read - чтение ввода: read -p "Имя:" name.
5. Арифметические операции: + - \* / % (остаток), сравнения == != > <, битовые & | ^ « ».
6. (( )) - для арифметики: res=\$((x+y)) и сравнений: if ((x>y)).
7. Стандартные переменные: \$HOME (домашний каталог), \$PATH (пути программ), \$USER (пользователь), \$SHELL (оболочка).
8. Метасимволы - спецсимволы: \* (любые символы), ? (один символ), > (перенаправление).
9. Экранирование: \ (обратный слэш), '' (полное), "" (частичное).
10. Создать файл script.sh с #!/bin/bash, дать права chmod +x script.sh, запустить ./script.sh.

11. Функции: `func(){ команды; return; }`. Вызов: `func`.
12. Проверка типа: `[ -f file ]` - файл, `[ -d dir ]` - каталог.
13. `set` - управление shell, `typeset` - тип переменной, `unset` - удаление.
14. Параметры передаются при вызове `./script.sh p1 p2`. В скрипте: `$1`, `$2` - параметры, `$@` - все, `$#` - количество.
15. Спецпеременные: `$0` - имя скрипта, `$?` - код возврата, `$$` - PID, `$_` - PID фонового.

## **5 Вывод**

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