

Лабораторная работа №11

Операционные системы

Савурская Полина

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	7
4	Выводы	14

Список иллюстраций

3.1	создание файлов	7
3.2	пишем текст	7
3.3	пишем код	8
3.4	запускаем файл	8
3.5	создание файлов	8
3.6	пишем код	9
3.7	пишем код	10
3.8	запуск файла file2.sh	10
3.9	создание файла file3.sh	10
3.10	пишем код	11
3.11	запуск файла file3.sh	11
3.12	создание файла file4.sh	12
3.13	пишем код	12
3.14	запуск файла file4.sh	13

1 Цель работы

Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

2 Задание

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами: `-i`—прочитать данные из указанного файла;

– `-o`—вывести данные в указанный файл;

– `-r`—указать шаблон для поиска;

– `-C`—различать большие и малые буквы;

– `-n`—выдавать номера строк.

а затем ищет в указанном файле нужные строки, определяемые ключом `-r`.

2. Написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем программа завершается с помощью функции `exit(n)`, передавая информацию в о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.

3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до `N` (например `1.tmp`, `2.tmp`, `3.tmp`, `4.tmp` и т.д.). Число файлов, которые необходимо создать, передается в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).

4. Написать командный файл, который с помощью команды `tar` упаковывает в архив все файлы в указанной директории. Модифицировать

его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду `find`).

3 Выполнение лабораторной работы

1. Создаем файлы file1.sh и file1.txt. Задаем им необходимые разрешения. Проверяем, появились ли у нас эти файлы.

```
[pasavurskaya@username ~]$ touch file1.txt  
[pasavurskaya@username ~]$ touch file1.sh  
[pasavurskaya@username ~]$ chmod +x file1.sh
```

Рис. 3.1: создание файлов

2. В файле file1.txt пишем текст, который будет выводиться на экран. В файле file1.sh пишем код.

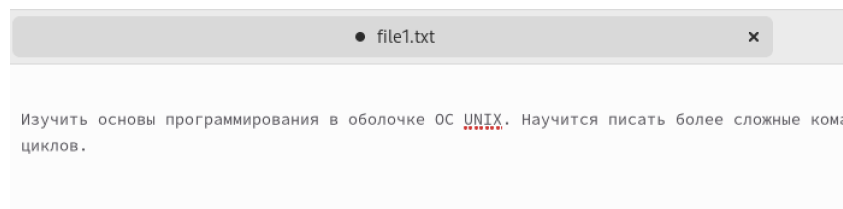


Рис. 3.2: пишем текст

```

file1.sh

#!/bin/bash
iflag=0; oflag=0; pflag=0; Cflag=0; nflag=0;
while getopts i:o:p:C:n optletter
do case $optletter in
    i) iflag=1; ival=$OPTARG;;
    o) oflag=1; oval=$OPTARG;;
    p) pflag=1; pval=$OPTARG;;
    C) Cflag=1;;
    n) nflag=1;;
    *) echo illegal option $optletter
        esac
done
if (($pflag==0))
then echo "Шаблон не найден"
else
    if (($iflag==0))
    then echo "Файл не найден"
    else
        if (($oflag==0))
        then if (($Cflag==0))
            then if (($nflag==0))

```

Рис. 3.3: пишем код

3. Задаем файлам необходимые конфигурации и запускаем.

```

[pasavurskaya@username ~]$ bash file1.sh -ifile1.txt -ofile1-1.txt -pice
[pasavurskaya@username ~]$ cat ~/file1.txt

```

Рис. 3.4: запускаем файл

4. Создаем файлы file2.sh и file2.c. Задаем им необходимые разрешения.

```

[pasavurskaya@username ~]$ touch file2.c *file2.sh
[pasavurskaya@username ~]$ chmod +x *.sh

```

Рис. 3.5: создание файлов

5. Открываем эти файлы и пишем там нужные коды.



```
file2.c

#include <stdio.h>
#include <stdlib.h>

int main() {
    printf("Введите число:");
    int a;
    scanf("%d", &a);
    if (a<0) exit(0);
    if (a>0) exit(1);
    if (a==0) exit(2);
    return 0;
}
```

Рис. 3.6: пишем код

```

file2.c

#!/bin/bash

gcc file2.c -o file2
./file2
code=$?
case $code in
    0) echo "Число меньше 0";;
    0) echo "Число больше 0";;
    0) echo "Число равно 0";;
esac

```

Рис. 3.7: пишем код

6. Запускаем файл file2.sh и вводим числа. Они выводятся с пояснением относительно нуля. Все сделано правильно.

```

[pasavurskaya@username ~]$ ./file2.sh
Введите число:4

```

Рис. 3.8: запуск файла file2.sh

7. Создаем файл file3.sh. Задаем ему необходимые разрешения. Проверяем, появился ли у нас этот файл.

```

pasavurskaya@username ~]$ touch file3.sh
pasavurskaya@username ~]$ chmod +x *.sh
pasavurskaya@username ~]$ ls
file1-1.txt  file2      file3.sh  Документы  Музыка     Шаб
file1.sh     file2.c   work      Загрузки   Общедоступные
file1.txt    file2.sh  Видео     Изображения 'Рабочий стол'

```

Рис. 3.9: создание файла file3.sh

8. Открываем этот файл и пишем там нужный код.

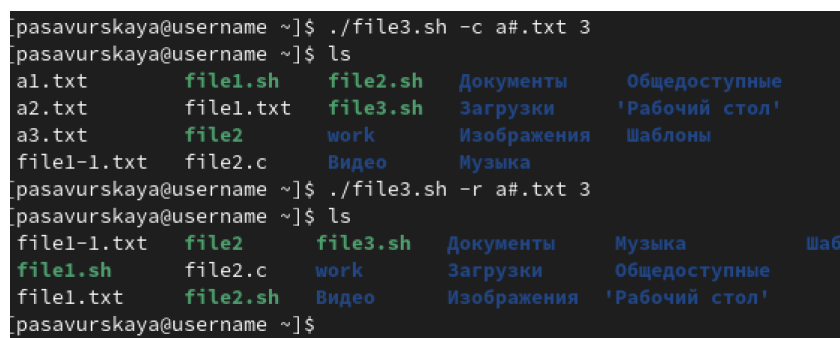


```
#!/bin/bash

opt=$1;
form=$2;
num=$3;
function Files() {
    for ((i=1; i<=$num; i++)) do
        file=$(echo $form | tr '#' "$i")
        if [ $opt == "-r" ]
        then
            rm -f $file
        elif [ $opt == "-c" ]
        then
            touch $file
        fi
    done
}
Files
```

Рис. 3.10: пишем код

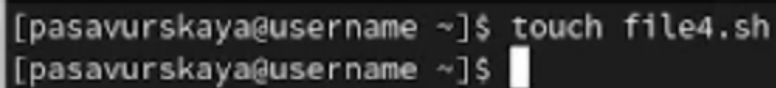
9. Запускаем файл file3.sh. Он создает указанное число файлов, пронумерованных последовательно от 1 до (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передается в аргументы командной строки. Этот же файл удаляет все созданные им файлы.



```
pasavurskaya@username ~]$ ./file3.sh -c a#.txt 3
pasavurskaya@username ~]$ ls
a1.txt      file1.sh  file2.sh  Документы  Общедоступные
a2.txt      file1.txt file3.sh  Загрузки  'Рабочий стол'
a3.txt      file2     work      Изображения  Шаблоны
file1-1.txt file2.c   Видео     Музыка
pasavurskaya@username ~]$ ./file3.sh -r a#.txt 3
pasavurskaya@username ~]$ ls
file1-1.txt file2     file3.sh  Документы  Музыка  Шаб.
file1.sh    file2.c   work      Загрузки  Общедоступные
file1.txt   file2.sh  Видео     Изображения  'Рабочий стол'
pasavurskaya@username ~]$
```

Рис. 3.11: запуск файла file3.sh

10. Создаем файл file4.sh.



```
[pasavurskaya@username ~]$ touch file4.sh
[pasavurskaya@username ~]$
```

Рис. 3.12: создание файла file4.sh

11. Открываем этот файл и пишем там нужный код.



```
#!/bin/bash

files=$(find ./ -maxdepth 1 -mtime -7)
listing=""
for file in "$files" ; do
    file=$(echo "$file" | cut -c 3-)
    listing="$listing $file"
done
dir=$(basename $(pwd))
tar -cvf $dir.tar $listing
```

Рис. 3.13: пишем код

12. Запускаем файл file4.sh.

```

[pasavurskaya@username ~]$ chmod +x *.sh
[pasavurskaya@username ~]$ ls -l
итого 48
-rw-r--r--. 1 pasavurskaya pasavurskaya    0 апр 18 21:03 file1-1.txt
-rwxr-xr-x. 1 pasavurskaya pasavurskaya  924 апр 18 21:02 file1.sh
-rw-r--r--. 1 pasavurskaya pasavurskaya  299 апр 18 21:22 file1.txt
-rwxr-xr-x. 1 pasavurskaya pasavurskaya 80448 апр 18 21:18 file2
-rw-r--r--. 1 pasavurskaya pasavurskaya   196 апр 18 21:17 file2.c
-rwxr-xr-x. 1 pasavurskaya pasavurskaya   187 апр 18 21:23 file2.sh
-rwxr-xr-x. 1 pasavurskaya pasavurskaya   231 апр 18 21:27 file3.sh
-rwxr-xr-x. 1 pasavurskaya pasavurskaya   204 апр 18 21:34 file4.sh
drwxr-xr-x. 1 pasavurskaya pasavurskaya    10 мар 11 14:21 work
drwxr-xr-x. 1 pasavurskaya pasavurskaya     0 фев 20 10:23 Видео
drwxr-xr-x. 1 pasavurskaya pasavurskaya     0 фев 20 10:23 Документы

```

Рис. 3.14: запуск файла file4.sh

4 Выводы

Я изучить основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.