Отчет по лабораторной работе №11

Алмазова Елизавета Андреевна

28.05.2022

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №11

Цель работы и задание

Цель данной лабораторной работы - изучить основы программирования в оболочке ОС UNIX. Научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Задание:

1. Используя команды getopts grep, написать командный файл, который анализирует командную строку с ключами:

– iinputfile — прочитать данные из указанного

Ход работы

1. Используя команды getopts grep, написала командный файл getops.sh, который анализирует командную строку с ключами inputfile — прочитать данные из указанного файла; outputfile — вывести данные в указанный файл; p - шаблон — указать шаблон для поиска; C — различать большие и малые буквы; n — выдавать номера строк, а затем ищет в указанном файле нужные строки, определяемые ключом -p. Проверила его работу на файле getops.txt (рис.1).

```
[eaalmazova@fedora ~]$ cat getops.txt
Hello world
A sentence to check how the program works
Hello WORLD
[eaalmazova@fedora ~]$ ./getops.sh -i getops.txt -p Hello -n
1:Hello world
3:Hello WORLD
[eaalmazova@fedora ~]$ cat getops.sh
#!/bin/bash

iflag=0                                                 #Initializating flags for options
oflag=0
pflag=0
Cflag=0
nflag=0

while getopts "i:o:p:Cn" optletter                      #Moving along the options
do case $optletter in                                   #Changing flag if found
        i) iflag=1
        ival=$OPTARG;;
        o) oflag=1
        oval=$OPTARG;;
        p) pflag=1
        pval=$OPTARG;;
        C) Cflag=1;;
        n) nflag=1;;
        esac
done
if (($pflag==0))
then echo "Template not found"
else
        if (($iflag==0))
        then echo "File not found"
        else
                if (($oflag==0))
                then if (($Cflag==0))
                        then if (($nflag==0))
                                then grep $pval $ival
                                else grep -n $pval $ival
                                fi
                        else if (($nflag==0))
                                then grep -i $pval $ival
                                else grep -i -n $pval $ival
                                fi
                        fi
                else if (($Cflag==0))
                        then if (($nflag==0))
                                then grep $pval $ival > $oval
                                else grep -n $pval $ival > $oval
                                fi
                        else if (($nflag==0))
                                then grep -i $pval $ival > $oval
                                else grep -i -n $pval $ival > $oval
                                fi
                        fi
                fi
        fi
fi[eaalmazova@fedora ~]$
```

Рисунок 1 - Работа файла getops.sh.

2. Написала на языке Си программу number.c, 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 о коде завершения в оболочку. Командный файл number.sh вызывает эту программу и, проанализировав с помощью команды $?, выдает сообщение о том, какое число было введено. Проверила его работу (рис.2).

```
[eaalmazova@fedora ~]$ cat number.c
#include <stdio.h>
#include <stdlib.h>

int main(){
        printf("Enter number: ");
        int num;
        scanf("%d", &num);
        if (num<0) exit(0);
        else if (num==0) exit(1);
        else if (num>0) exit(2);
        return 0;
}[eaalmazova@fedora ~]$ cat number.sh
#!/bin/bash

gcc number.c -o number          #Compiling program in C
./number                                #Executing program in C
code=$?                         #Getting code
case $code in
        0) echo "Number < 0";;
        1) echo "Number = 0";;
        2) echo "Number > 0";;
esac[eaalmazova@fedora ~]$ ./number.sh
Enter number: 10
Number > 0
```

Рисунок 2 - Работа файла number.sh.

3. Написала командный файл file.sh, 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, передаётся в аргументы командной строки. Этот же командный файл умеет удалять все созданные им файлы. Создание или удаление файлов регулируется с помощью опций -c и -r (рис.3). Проверила его работу (рис.4)

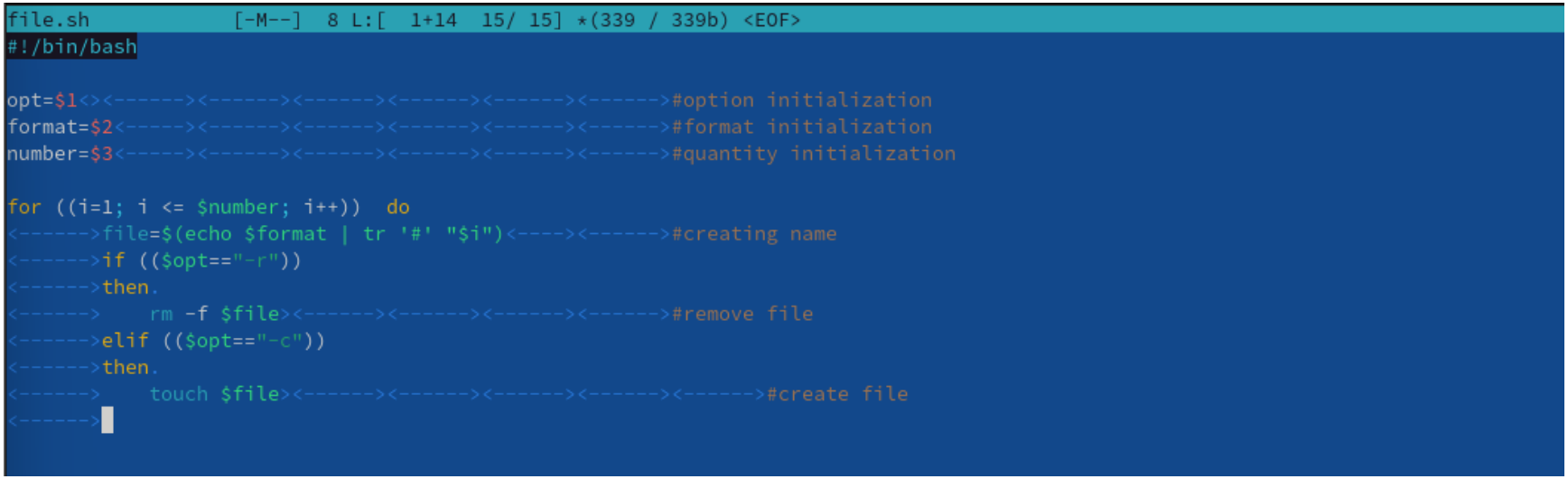

Рисунок 3 - Процесс редактирования file.sh.

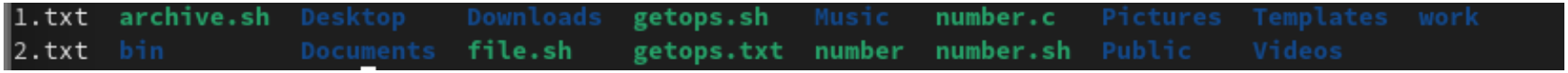

Рисунок 4 - Работа файла file.sh.

4. Написала командный файл archive.sh, который с помощью команды tar запаковывает в архив все файлы в указанной директории и модифицировала его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад с помощью команды find (рис.5). Проверила его работу (рис.6).

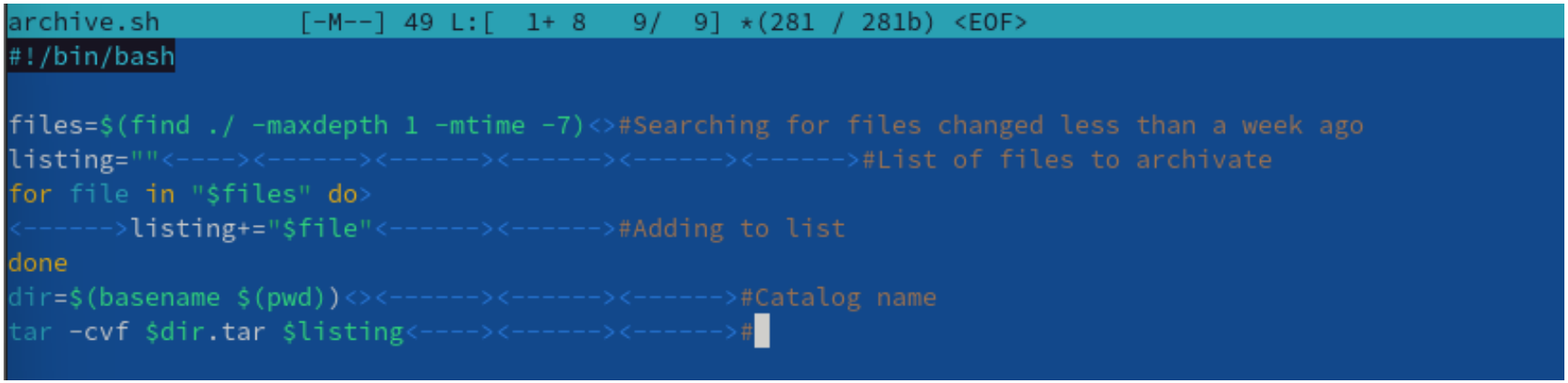

Рисунок 5 - Процесс редактирования archive.sh.

```
[eaalmazova@fedora ~]$ tar -tf ~.tar
1.txt
2.txt
archive.sh
file.sh
getops.sh
getops.txt
number
number.c
number.sh
```

Рисунок 6 - Работа файла archive.sh.

Выводы

В ходе выполнения данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX, научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Спасибо за внимание!