Лабораторная работа №11

Ханина Людмила Константиновна

Table of Contents

Цель работы

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

Задание

- Используя команды getoptsgrep, написать командный файл, который анализирует командную строку с ключами: – -iinputfile — прочитать данные из указанного файла; – -ooutputfile — вывести данные в указанный файл; – -ршаблон — 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.
- 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. Команд- ный файл должен вызывать эту программу и, проанализировав с помощью команды $?, выдать сообщение о том, какое число было введено.
- 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, передаётся в аргументы командной строки. Этот же ко- мандный файл должен уметь удалять все созданные им файлы (если они существуют).
- Написать командный файл, который с помощью команды tar запаковывает 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).

Выполнение лабораторной работы

1. Создадим файл lab111.sh и запишем в него скрипт, который будет анализировать командную строку с ключами. Далее изменим доступ к файлу, чтобы можно было его запускать.

```
[lkkhanina@fedora ~]$ vi lab111.sh
[lkkhanina@fedora ~]$ chmod +x lab111.sh
[lkkhanina@fedora ~]$ cat lab111.sh
#!/bin/bash
ifl=0; ofl=0; pfl=0; cfl=0; nfl=0;
while getopts i:o:p:Cn optletter
do case $optletter in
        i) ifl=1; ival=$OPTARG;;
        o) ofl=1; oval=$OPTARG;;
        p) pfl=1; pval=$OPTARG;;
        C) cfl=1;;
        n) nfl=1;;
        *) echo illegal option $optletter
    esac
done

if (($pfl==0))
then echo "Sample not found"
else
    if (($ifl==0))
    then touch "File not found"
    else
        if (($ofl==0))
        then if (($cfl==0))
            then if (($nfl==0))
                then grep $pval $ival
                else grep -n $pval $ival
                fi
            else if (($nfl==0))
                then grep -i $pval $ival
                else grep -i -n $pval $ival
                fi
            fi
        else if (($cfl==0))
            then if (($nfl==0))
                then grep $pval $ival > $oval
                else grep -n $pval $ival > $oval
                fi
            else if (($nfl==0))
                then grep -i $pval $ival > $oval
                else grep -i -n $pval $ival > $oval
                fi
            fi
        fi
    fi
fi
```

Первый скрипт

2. Проверим, что он работает. Для этого создадим файлы f1.txt и f2.txt. В первый напишем текст. Теперь запустим файл и увидим, что программа отработала корректно.

```
[lkkhanina@fedora ~]$ touch f1.txt f2.txt
[lkkhanina@fedora ~]$ vi f1.txt
[lkkhanina@fedora ~]$ cat f1.txt
hi
i wanna swim
i wanna eat
bye
[lkkhanina@fedora ~]$ ./lab111.sh -i f1.txt -o f2.txt -p wanna -n
[lkkhanina@fedora ~]$ cat f2.txt
2:i wanna swim
3:i wanna eat
[lkkhanina@fedora ~]$ ./lab111.sh -i f1.txt -o f2.txt -p wanna -C -n
[lkkhanina@fedora ~]$ cat f2.txt
2:i wanna swim
3:i wanna eat
[lkkhanina@fedora ~]$ ./lab111.sh -i f1.txt -C -n
Sample not found
```

Запускаем первый скрипт

3. Далее создадим файлы lab112.c и lab112.sh. В первый запишем скрипт, который бдут определять, какое число ввел пользователь. Во второй скрипт, который будет анализировать исполнение первого. Изменим доступ к файлу lab112.sh, чтобы можно было его запускать.

```
[lkkhanina@fedora ~]$ touch lab112.sh lab112.c
[lkkhanina@fedora ~]$ vi lab112.c
[lkkhanina@fedora ~]$ vi lab112.sh
```

Второй скрипт часть 1

```
[lkkhanina@fedora ~]$ cat lab112.c
#include <stdio.h>
#include <stdlib.h>
int main() {
    printf("Enter the number\n");
    int n;
    scanf("%d", &n);
    if (n > 0) exit(1);
    if (n == 0) exit(0);
    if (n < 0) exit(2);
    return 0;
}
[lkkhanina@fedora ~]$ cat lab112.sh
#!/bin/bash
gcc lab112.c -o lab112
./lab112
code=$?
case $code in
    0) echo "Number is 0";;
    1) echo "Number is more than 0";;
    2) echo "Number is less than 0"
esac
[lkkhanina@fedora ~]$ chmod +x lab112.sh
[lkkhanina@fedora ~]$
```

Второй скрипт часть 2

4. Теперь запустим файл и увидим, что программа отработала корректно.

```
[lkkhanina@fedora ~]$ ./lab112.sh
Enter the number
0
Number is 0
[lkkhanina@fedora ~]$ ./lab112.sh
Enter the number
22
Number is more than 0
[lkkhanina@fedora ~]$ ./lab112.sh
Enter the number
-7
Number is less than 0
[lkkhanina@fedora ~]$
```

Запускаем второй скрипт

5. Создадим файл lab113.sh и запишем в него скрипт, который будет с опцией -c создавать заданное количество файлов, а с опцией -r — их удалять. Далее изменим доступ к файлу, чтобы можно было его запускать.

```
[lkkhanina@fedora ~]$ touch lab113.sh
[lkkhanina@fedora ~]$ vi lab113.sh
[lkkhanina@fedora ~]$ chmod +x lab113.sh
[lkkhanina@fedora ~]$ cat lab113.sh
#!/bin/bash
opt=$1;
f=$2;
n=$3;
function files() {
    for ((i=1; i<=n; ++i)) do
        file=$(echo $f | tr '#' "$i")
        if [ $opt == "-r" ]
        then rm -f $file
        elif [ $opt=="-c" ]
        then touch $file
        fi
    done
}
files
```

Третий скрипт

6. Теперь запустим файл и увидим, что программа отработала корректно.

```
[lkkhanina@fedora ~]$ ls
 australia   feathers         lab10third.sh   newcat       Видео
 backup      file.old         lab111          new.txt      Документы
 bin         file.txt         lab111.sh       os-intro     Загрузки
 conf.txt    go               lab112          play         Изображения
 dir         lab07.sh         lab112.c        ski.places   Музыка
 equipment   lab07.sh~        lab112.sh       text.py      Общедоступные
 f1.txt      lab10first.sh    lab113.sh       text.txt    'Рабочий стол'
 f2.txt      lab10fourth.sh   my_os           work         Шаблоны
[lkkhanina@fedora ~]$ ./lab113.sh -c file#.txt 4
[lkkhanina@fedora ~]$ ls
 australia   file1.txt   lab10first.sh    my_os        Видео
 backup      file2.txt   lab10fourth.sh   newcat       Документы
 bin         file3.txt   lab10third.sh    new.txt      Загрузки
 conf.txt    file4.txt   lab111           os-intro     Изображения
 dir         file.old    lab111.sh        play         Музыка
 equipment   file.txt    lab112           ski.places   Общедоступные
 f1.txt      go          lab112.c         text.py     'Рабочий стол'
 f2.txt      lab07.sh    lab112.sh        text.txt     Шаблоны
 feathers    lab07.sh~   lab113.sh        work
[lkkhanina@fedora ~]$ ./lab113.sh -r file#.txt 4
[lkkhanina@fedora ~]$ ls
 australia   feathers         lab10third.sh   newcat       Видео
 backup      file.old         lab111          new.txt      Документы
 bin         file.txt         lab111.sh       os-intro     Загрузки
 conf.txt    go               lab112          play         Изображения
 dir         lab07.sh         lab112.c        ski.places   Музыка
 equipment   lab07.sh~        lab112.sh       text.py      Общедоступные
 f1.txt      lab10first.sh    lab113.sh       text.txt    'Рабочий стол'
 f2.txt      lab10fourth.sh   my_os           work         Шаблоны
[lkkhanina@fedora ~]$
```

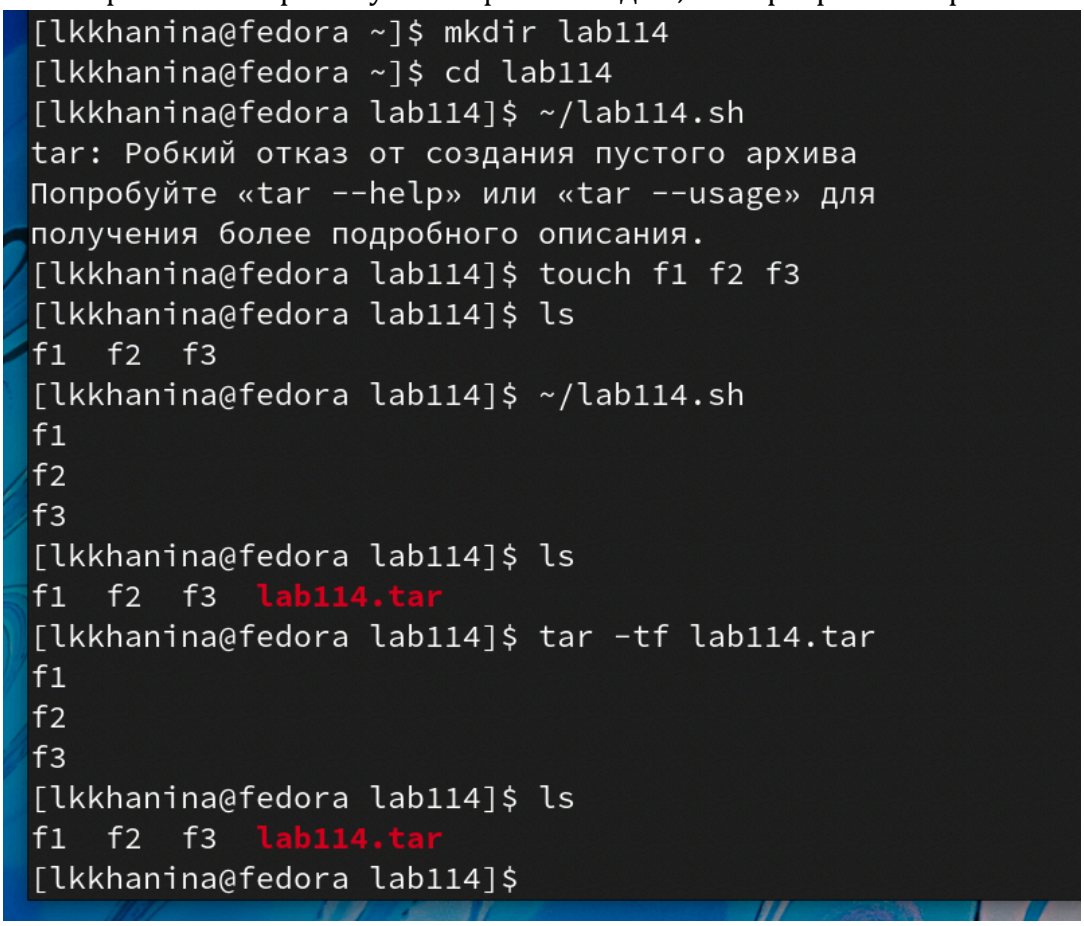
Запускаем третий скрипт

7. Создадим файл lab114.sh и запишем в него скрипт, который будет с помощью команды tar запаковывает в архив все файлы, которые были изменены менее недели тому назад, в указанной директории. Далее изменим доступ к файлу, чтобы можно было его запускать.

```
[lkkhanina@fedora ~]$ vi lab114.sh
[lkkhanina@fedora ~]$ chmod +x lab114.sh
[lkkhanina@fedora ~]$ cat lab114.sh
#!/bin/bash
files=$(find ./ -maxdepth 1 -mtime -7)
l=""
for f in "$files" ; do
    file=$(echo "$f" | cut -c 3-)
    l="$l $file"
done
dir=$(basename $(pwd))
tar -cvf $dir.tar $l
[lkkhanina@fedora ~]$
```

Четвертый скрипт

8. Чтобы проверить корректность скрипта, создаем директорию и добавляем в нее файлы. Теперь запускаем файл и видим, что программа отработала корректно.



Запускаем четвертый скрипт

Контрольные вопросы

1. Команда getopts является встроенной командой командной оболочки bash, предназначенной для разбора параметров сценариев. Она обрабатывает исключительно однобуквенные параметры как с аргументами, так и без них и этого вполне достаточно для передачи сценариям любых входных данных.

2. При генерации имен используют метасимволы:

- ’*’ — произвольная (возможно пустая) последовательность символов;
- ? — один произвольный символ;
- […] — любой из символов, указанных в скобках перечислением и/или с указанием диапазона;
- cat f* — выдаст все файлы каталога, начинающиеся с “f”;
- cat *f* — выдаст все файлы, содержащие “f”;
- cat program.? – выдаст файлы данного каталога с однобуквенными расширениями, скажем * “program.c” и “program.o”, но не выдаст “program.com”;
- cat [a-d]* – выдаст файлы, которые начинаются с “a”, “b”, “c”, “d”.

3. Операторы управления действиями: if, for, while, case.
4. Чтобы прервать цикл, можно использовать оператор break.
5. Операция сравнения возвращает: значение true («истина»), если высказывание с оператором правдивое (условие выполняется), и false («ложь») — если высказывание с оператором ложное (условие не выполняется).
6. Строка if test -f man$s$/i.sпроверяет, существуетлифайл$mans/i.s$ и является ли этот файл обычным файлом. Если данный файл является каталогом, то команда вернет false.
7. Цикл while выполняет тело цикла пока условие истинно. Цикл until выполняет тело цикла пока условие ложно.

Выводы

Я научилась писать более сложные командный файлы с использованием логических управляющих конструкций и циклов.