Операционные системы

Лабораторная работа №11

Матюшкин Денис Владимирович (НПИбд-02-21)

Содержание

1 Цель работы

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

2 Ход работы

1. Используя команды getopts grep, напишем командный файл, который анализирует командную строку с ключами:
   – -iinputfile — прочитать данные из указанного файла;
   – -ooutputfile — вывести данные в указанный файл;
   – -pшаблон — 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 (рис. 2.1). Проверим написанный командный файл (рис. 2.2).

```
#!/bin/bash

while getopts "i:o:p:C:n" opt
do
    case $opt in
        i)input=$OPTARG;;
        o)output=$OPTARG;;
        p)shablon=$OPTARG;;
        C)registr="";;
        n)number="";;
    esac
done

grep -n "$shablon" "$input" > "$output"
```

Рис. 2.1: Командный файл поиска файлов

```
[dvmatyushkin@dvmatyushkin ~]$ chmod +x lab11_1.sh
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_1.sh -i text.txt -o test.txt -p "sh" -C -n
[dvmatyushkin@dvmatyushkin ~]$ cat test.txt
1:lab1.sh
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_1.sh -i text.txt -o test.txt -p "tg" -C -n
[dvmatyushkin@dvmatyushkin ~]$ cat test.txt
5:zxc.tg
6:xty.tg
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_1.sh -i text.txt -o test.txt -p "" -C -n
[dvmatyushkin@dvmatyushkin ~]$ cat test.txt
1:lab1.sh
2:lab2.txt
3:lab3.cpp
4:abc
5:zxc.tg
6:xty.tg
7:stifell.io
8:nice
9:good
10:well
[dvmatyushkin@dvmatyushkin ~]$
```

Рис. 2.2: Проверка файла

2. Напишем 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 вызывает эту программу и, проанализировав с помощью команды $?, выдает сообщение о том, какое число было введено (рис. 2.3 и рис. 2.4). Проверим написанный командный файл (рис. 2.5).

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

int main(){
  int n;
  printf("Введите число: "); scanf("%d",&n);
  if (n > 0) printf("Число больше нуля\n", n);
  else if (n < 0) printf("Число меньше нуля\n", n);
  else printf("Число равно нулю\n", n);
  printf("Введеное число: "); exit(n);
  return n;
}
```

Рис. 2.3: Программа на Си

```
#!/bin/bash

gcc lab11.c -o lab11
./lab11
echo $?
```

Рис. 2.4: Командный файл

```
[dvmatyushkin@dvmatyushkin ~]$ emacs lab11.c
[dvmatyushkin@dvmatyushkin ~]$ emacs lab11_2.sh
[dvmatyushkin@dvmatyushkin ~]$ chmod +x lab11_2.sh
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_2.sh
Введите число: 5
Число больше нуля
Введеное число: 5
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_2.sh
Введите число: 0
Число равно нулю
Введеное число: 0
[dvmatyushkin@dvmatyushkin ~]$
```

Рис. 2.5: Проверка файла

3. Напишем 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 создается, передаётся в аргументы командной строки. Этот же командный файл удаляет все созданные им файлы (если они существуют) (рис. 2.6). Проверим написанный командный файл (рис. 2.7).

```
#!/bin/bash

while getopts "c:r" opt
do
    case $opt in
        c)n=$OPTARG;
        for i in $(seq 1 $n);
        do
            touch "$i.tmp";
        done;;
        r)for i in $(find -name "*.tmp");
         do
             rm $i;
          done;;
    esac
done
```

Рис. 2.6: Командный файл для создание файлов

```
[dvmatyushkin@dvmatyushkin ~]$ chmod +x lab11_3.sh
[dvmatyushkin@dvmatyushkin ~]$ ./lab11.sh -c 10
bash: ./lab11.sh: Нет такого файла или каталога
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_3.sh -c 10
[dvmatyushkin@dvmatyushkin ~]$ ls
 10.tmp   5.tmp   bin          lab11_2.sh~   lab11.c~     resources   themes      Изображени
 1.tmp    6.tmp   lab11        lab11_3.sh    lab11.cpp~   snap        work        Музыка
 2.tmp    7.tmp   lab11_1.sh   lab11_3.sh~   opt          test.txt    Видео       Общедоступ
 3.tmp    8.tmp   lab11_1.sh~  lab11_4.sh    output       text.txt    Документы   'Рабочий ст
 4.tmp    9.tmp   lab11_2.sh   lab11.c       outputfile   text.txt~   Загрузки    Шаблоны
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_3.sh -r
find: ‘./.local/share/Trash/files/monthly/play’: Отказано в доступе
[dvmatyushkin@dvmatyushkin ~]$ ls
 bin          lab11_2.sh    lab11_4.sh   opt          snap        themes      Загрузки
 lab11        lab11_2.sh~   lab11.c      output       test.txt    work        Изображения
 lab11_1.sh   lab11_3.sh    lab11.c~     outputfile   text.txt    Видео       Музыка
 lab11_1.sh~  lab11_3.sh~   lab11.cpp~   resources    text.txt~   Документы   Общедоступные
[dvmatyushkin@dvmatyushkin ~]$
```

Рис. 2.7: Проверка файла

4. Напишем командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицируем его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (используем команду find) (рис. 2.8). Проверим написанный командный файл (рис. 2.9 и рис. 2.10).

```
#!/bin/bash

while getopts ":p:" opt
do
    case $opt in
        p)dir=$OPTARG;;
    esac
done

find $dir -maxdepth 1 -ctime -7 -type f -print0 | xargs -0 tar rfv test.tar
```

Рис. 2.8: Командный файл архиватор

```
[dvmatyushkin@dvmatyushkin ~]$ emacs lab11_4.sh
[dvmatyushkin@dvmatyushkin ~]$ chmod +x lab11_4.sh
[dvmatyushkin@dvmatyushkin ~]$ ./lab11_4.sh
./.vboxclient-clipboard.pid
./.vboxclient-seamless.pid
./.vboxclient-draganddrop.pid
./.vboxclient-display-svga-x11.pid
./.bash_history
./outputfile
./output
./lab11_1.sh~
./lab11_1.sh
./lab11.cpp~
./lab11_2.sh~
./lab11_2.sh
./lab11_4.sh
./text.txt~
./text.txt
./test.txt
./lab11.c~
./lab11.c
./lab11
./lab11_3.sh~
./lab11_3.sh
[dvmatyushkin@dvmatyushkin ~]$
```

Рис. 2.9: Проверка файла

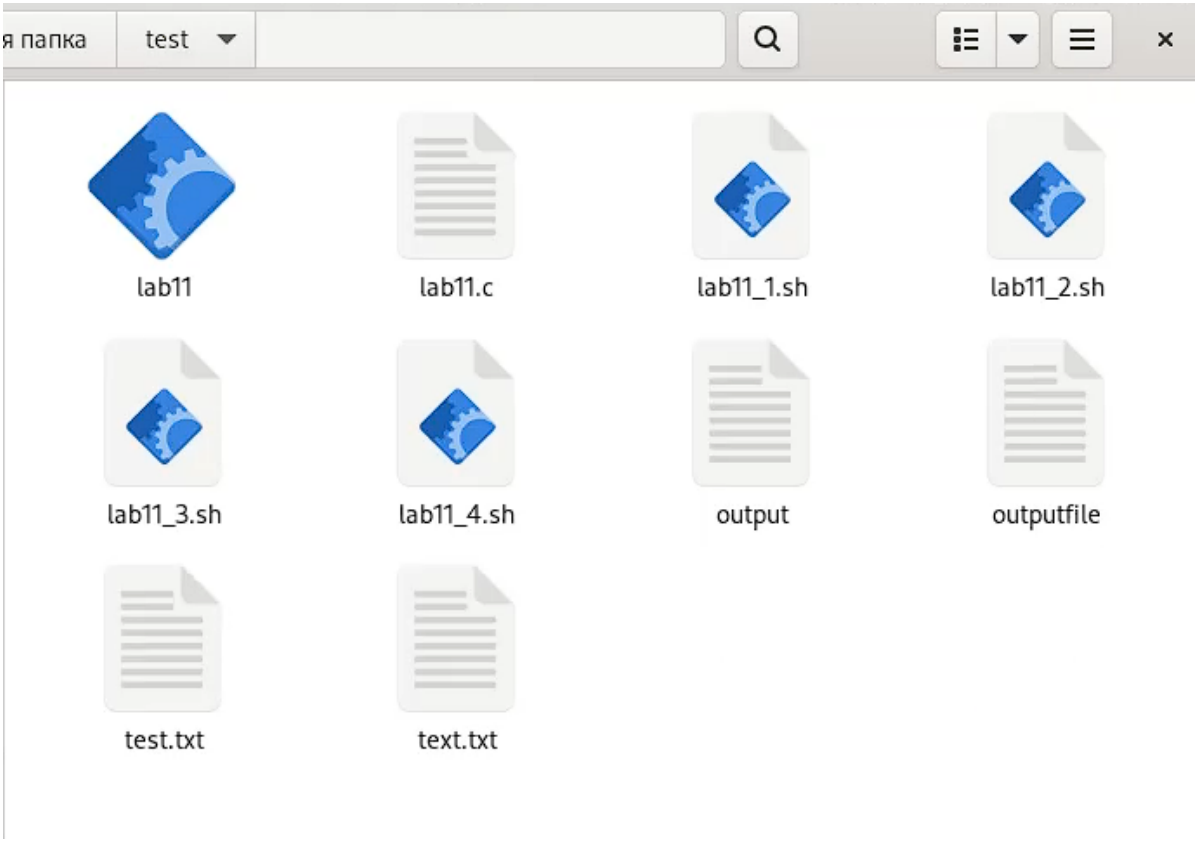

Рис. 2.10: Проверка файла

3 Контрольные вопросы

1. Каково предназначение команды getopts?

Команда getopts используется для разбора параметров и проверки опций на допустимость. Осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных.

2. Какое отношение метасимволы имеют к генерации имён файлов?

Метасимволы отвечают за параметры выдачи файлов, а следовательно могут вносить иной смысл, нежели прямое значение как символа. Например:

• * — соответствует произвольной, в том числе и пустой строке;

• ? — соответствует любому одному символу;

• [c1-c1] — соответствует любому символу, лексикографически на ходящемуся между символами c1 и c2.

• echo * — выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды ls;

• ls *.c — выведет все файлы с последними двумя символами, равными .c.*

• *echo prog.? — выдаст все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются prog. .*

• *[a-z]* — соответствует произвольному имени файла в текущем каталоге, начинающемуся с любой строчной буквы латинского алфавита.

3. Какие операторы управления действиями вы знаете?

Ответ: for, break , while, until, case ,continue, if, else.

4. Какие операторы используются для прерывания цикла?

Ответ: break

5. Для чего нужны команды false и true?

Это логические значения (0 и 1 соответственно). Можно использовать как более понятный вариант.

6. Ответ: условие существование файла man$s$/i.$s

7. Объясните различия между конструкциями while и until.

while - проверка условия затем выполнение тела цикла.

until - выполнение тела цикла затем проверка условия.

4 Вывод

- В ходе этой лабораторной работы мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.