Отчет по лабораторной работе №11

Павлова Варвара Юрьевна НПМбд-02-21

Москва, 2022

Цель работы

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

Ход работы

Первый скрипт

Написание

Используя команды getopts grep, пишу 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. [-@fig:001]) (рис. [-@fig:002])

Проверка работы

Добавляю право на выполнение файла и проверяю его работу. (рис. [-@fig:003])

Второй скрипт

Написание

Пишу 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. (рис. [-@fig:004])

```
#!/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 "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
```

Рис. 1: написание скрипта

```
                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
```

Рис. 2: написание скрипта

```
[vypavlova@fedora ~]$ chmod +x 1.sh
[vypavlova@fedora ~]$ ./1.sh -i 11.txt -C -n
not found
[vypavlova@fedora ~]$ ./1.sh -o 22.txt -p file -C -n
file not found
```

Рис. 3: проверка первого файла

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
  int a;
  printf("Input the number: \n");
  scanf ("%d", &a);
  if (a<0)
    exit(0);
  if (a>0)
    exit(1);
  if (a==0)
    exit(2);
  return 0;
}
```

Рис. 4: написание 2.c

Написание

Командный файл вызывает эту программу и, проанализировав с помощью команды $?, выдает сообщение о том, какое число было введено. (рис. [-@fig:005])

```
#!/bin/bash
gcc 2.c -o g
./g
code=$?
case $code in
    0) echo "<0";;
    1) echo ">0";;
    2) echo "=0";;
esac
```

Рис. 5: написание 2.sh

Проверка работы

Добавляю право на выполнение файла и проверяю его работу. (рис. [-@fig:006])

```
[vypavlova@fedora ~]$ chmod +x 2.sh
[vypavlova@fedora ~]$ ./2.sh
Input the number:
0
=0
[vypavlova@fedora ~]$ ./2.sh
Input the number:
7
>0
[vypavlova@fedora ~]$ ./2.sh
Input the number:
-8
<0
```

Рис. 6: проверка второго файла

Третий скрипт

Написание

Пишу 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 командный файл должен уметь удалять все созданные им файлы (если они существуют). (рис. [-@fig:007])

```
#!/bin/bash
opt=$1;
format=$2;
number=$3;
function g ()
{
    for (( i=1; i<=$number; i++)) do
        file=$(echo $format | tr '#' "$i")
        if (( $opt == "-r" ))
        then
            rm -f $file
        elif (($opt == "-c"))
        then
            touch $file
        fi
    done
}
g
```

Рис. 7: написание скрипта

Проверка работы

Добавляю право на выполнение файла и проверяю его работу. (рис. [-@fig:008])

```
[vypavlova@fedora ~]$ chmod +x 3.sh
[vypavlova@fedora ~]$ ./3.sh -c et#.txt 3
[vypavlova@fedora ~]$ ls
11.txt  2.c    3.sh        bin         g           Public      Videos
1.sh    2.c~   3.sh~       Desktop     Music       reports     work
1.sh~   2.sh   3.txt       Documents   my_os       second.sh~
1.txt   2.sh~  backup      Downloads   opsystemlab Templates
22.txt  2.txt  backup.sh~  fourth.sh~  Pictures    third.sh~
[vypavlova@fedora ~]$ ./3.sh -r et#.txt 3
[vypavlova@fedora ~]$ ls
11.txt  2.c    3.sh        bin         fourth.sh~  opsystemlab  second.sh~  work
1.sh    2.c~   3.sh~       Desktop     g           Pictures     Templates
1.sh~   2.sh   backup      Documents   Music       Public       third.sh~
22.txt  2.sh~  backup.sh~  Downloads   my_os       reports      Videos
```

Рис. 8: проверка третьего файла

Четвертый скрипт

Написание

Пишу командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицирую его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использую команду find). (рис. [-@fig:009])

Проверка работы

Добавляю право на выполнение файла и проверяю его работу. (рис. [-@fig:009])

```
#!/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. 9: написание скрипта

```
#!/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. 10: проверка четвертого файла

Выводы

Выполняя данную лабораторную работу я изучила основы программирования в оболочке ОС UNIX/Linux и научилась писать небольшие командные файлы.