

Отчёт по лабораторной работе №9

Программирование в командном процессе ОС UNIX.

Ветвления и циклы

Архипов Юрий Денисович

Цель работы

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

Ход работы

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

- `-i`inputfile – прочитать данные из указанного файла;
- `-o`outputfile – вывести данные в указанный файл;
- `-r`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 `-r`.

а) Создаём файл и заходим в редактор `emacs`, после меняем права доступа.

Рис.1

```
[root@yurallarhipov ~]# touch lab9.sh
[root@yurallarhipov ~]# emacs lab9.sh
[root@yurallarhipov ~]# chmod +x lab9.sh
```

b) Пишем скрипт.

```
#!/bin/bash

i_=""
o_=""
p_=""
C_=0
n_=0

while getopts "i:p:p:Cn" opt
do
    case $opt in
        i) i_="$OPTARG";;
        o) o_="$OPTARG";;
        p) p_="$OPTARG";;
        C) C_=1;;
        n) n_=1;;
    esac
done
if (($C_ + $n_ == 2))
then
    grep -i -n "$p_" "$i_" > "$o_"
elif (($C_ + $n_ == 0))
then
    grep "$p_" "$p_" > "$o_"
elif (($C_ == 1))
then
    grep -n "$p_" "$i_" > "$o_"
fi
```

Рис.2 **fi**

c) Создание и заполнение .txt файлов.

```
[root@yurallarhipov ~]# touch fromfile.txt tofile.txt
[root@yurallarhipov ~]# emacs fromfile.txt
[root@yurallarhipov ~]# cat fromfile.txt
arhipov
ARHIPOV
ARhiPOV
```

Рис.3

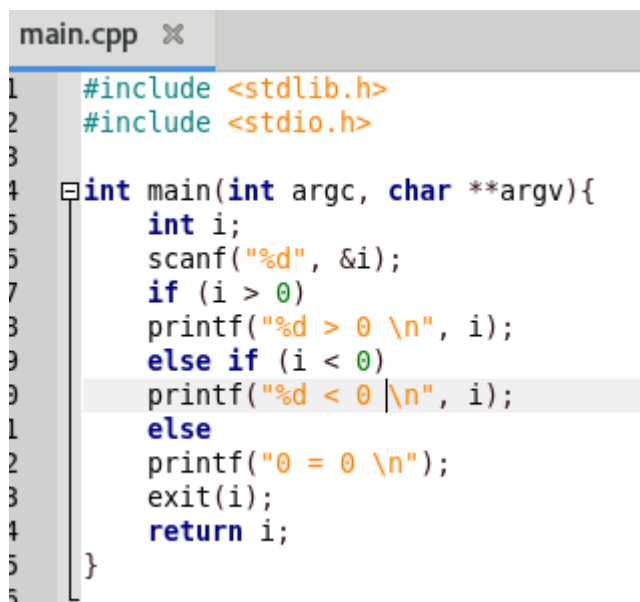
d) Запускаем скрипт и проверяем работу.

Рис.4

```
[root@yurallarhipov ~]# ./lab9.sh -i fromfile.txt -o tofile.txt -p arhipov
[root@yurallarhipov ~]# cat tofile.txt
arhipov
[root@yurallarhipov ~]# ./lab9.sh -i fromfile.txt -o tofile.txt -p arhipov -C
[root@yurallarhipov ~]# cat tofile.txt
arhipov
ARHIPOV
ARhiPOV
[root@yurallarhipov ~]# ./lab9.sh -i fromfile.txt -o tofile.txt -p arhipov -C -n
[root@yurallarhipov ~]# cat tofile.txt
1:arhipov
2:ARHIPOV
3:ARhiPOV
[root@yurallarhipov ~]#
```

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 о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.

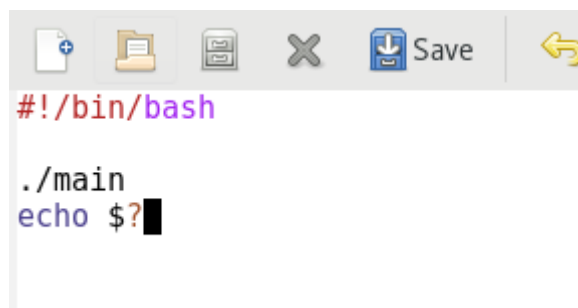
а) Создаём файл `.cpp`, запускаем `geany`, пишем программу.



```
main.cpp x
1  #include <stdlib.h>
2  #include <stdio.h>
3
4  int main(int argc, char **argv){
5      int i;
6      scanf("%d", &i);
7      if (i > 0)
8          printf("%d > 0 \n", i);
9      else if (i < 0)
10         printf("%d < 0 \n", i);
11     else
12         printf("0 = 0 \n");
13     exit(i);
14     return i;
15 }
```

Рис.5

б) Создаём файл `.sh`, запускаем `emacs`, пишем скрипт.

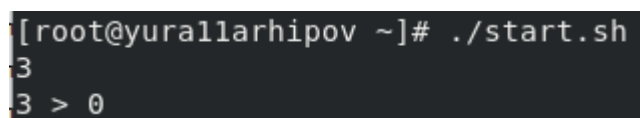


```
#!/bin/bash

./main
echo $?
```

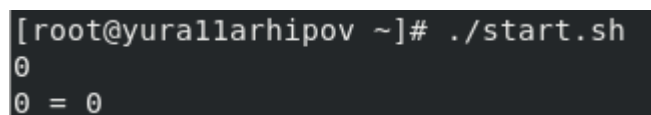
Рис.6

в) Запускаем скрипт, проверяем работу программ.



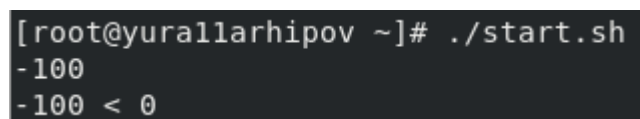
```
[root@yurallarhipov ~]# ./start.sh
3
3 > 0
```

Рис.7



```
[root@yurallarhipov ~]# ./start.sh
0
0 = 0
```

Рис.8



```
[root@yurallarhipov ~]# ./start.sh
-100
-100 < 0
```

Рис.9

3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 да N. Число файлов, которые необходимо создать, передаётся в аргументы командной строки.

Этот же командный файл должен уметь удалять все созданные им файлы.

а) Создаём файл .sh, запускаем его в редакторе emacs, после меняем права доступа.

Рис.10

```
[root@yurallarhipov ~]# touch 123.sh
[root@yurallarhipov ~]# emacs 123.sh
[root@yurallarhipov ~]# chmod +x 123.sh
```

б) Пишем скрипт.

```
#!/bin/bash
r_=0
n_=0
name=1
while getopts "r:n:" opt
do
    case $opt in
        r)r_="$OPTARG";;
        n)n_="$OPTARG";;
        esac
    done
    if [ "$r_" -eq '0' ]
    then
        while (($name!=($n_+1)))
        do
            for i in $name
            do
                touch $i.tmp
            done
            ((name+=1))
        done
    else
        name=1
        while (($name!=($n_+1)))
        do
            for i in $name
            do
                rm $i.tmp
            done
            ((name+=1))
        done
    fi
```

Рис.11

с) Запускаем скрипт, смотрим результат.

```
[root@yurallarhipov yurallarhipov]# ./123.sh -r 0 -n 4
[root@yurallarhipov yurallarhipov]# ls
123.sh  4.tmp  file.sh  may  Видео  Общедоступные
123.sh~ abcl   file.txt  play  Документы  'Рабочий стол'
1.tmp   conf.txt fs       reports  Загрузки  Шаблоны
2.tmp   dev    hello.cpp text.txt  Изображения
3.tmp   feathers lab005  work     Музыка

[root@yurallarhipov yurallarhipov]# ./123.sh -r 1 -n 4
[root@yurallarhipov yurallarhipov]# ls
123.sh  dev  fs  play  Видео  Музыка
123.sh~ feathers  hello.cpp  reports  Документы  Общедоступные
abcl   file.sh  lab005  text.txt  Загрузки  'Рабочий стол'
conf.txt  file.txt  may  work  Изображения  Шаблоны
```

Рис.12

4. 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).

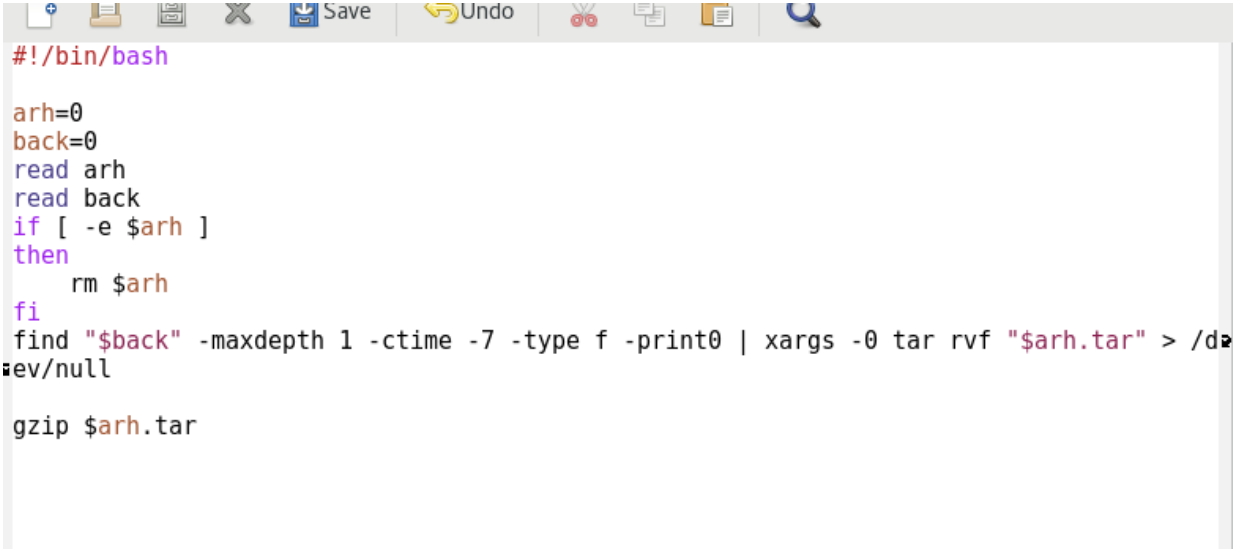
а) Создаём файл, запускает emacs и меняем права доступа.

```
[root@yurallarhipov yurallarhipov]# touch 444.sh
[root@yurallarhipov yurallarhipov]# chmod +x 444.sh
[root@yurallarhipov yurallarhipov]# emacs 444.sh
```

Рис.13

б) Пишем скрипт.

Рис.14



```
#!/bin/bash

arh=0
back=0
read arh
read back
if [ -e $arh ]
then
    rm $arh
fi
find "$back" -maxdepth 1 -ctime -7 -type f -print0 | xargs -0 tar rvf "$arh.tar" > /dev/null

gzip $arh.tar
```

Рис.15

```
[root@yurallarhipov yurallarhipov]# ./444.sh
yura
/home/yurallarhipov
tar: Удаляется начальный '/' из имен объектов
tar: Удаляются начальные '/' из целей жестких ссылок
[root@yurallarhipov yurallarhipov]# ls
123.sh  conf.txt  fs      reports  Документы  'Рабочий стол'
123.sh~ dev      hello.cpp  text.txt  Загрузки   Шаблоны
444.sh  feathers  lab005    work     Изображения
444.sh~ file.sh   may      yura.tar.gz  Музыка
```

Рис.16 [root@yurallarhipov yurallarhipov]# less yura.tar.gz

```
-rw-rw-r-- yurallarhipov/yurallarhipov 3065 2021-11-20 13:29  
file.txt  
-rw-rw-r-- yurallarhipov/yurallarhipov 527 2021-11-20 13:32  
conf.txt  
-rw-rw-r-- yurallarhipov/yurallarhipov 472 2021-11-20 20:10  
text.txt  
-rw-rw-r-- yurallarhipov/yurallarhipov 0 2021-11-20 20:13  
hello.cpp  
-rw----- yurallarhipov/yurallarhipov 2889 2021-11-20 21:51  
.viminfo  
-rwxrwxr-x yurallarhipov/yurallarhipov 363 2021-11-21 22:24  
123.sh~  
-rwxrwxr-x yurallarhipov/yurallarhipov 376 2021-11-21 22:33  
123.sh  
-rwxrwxr-x yurallarhipov/yurallarhipov 0 2021-11-21 22:12  
file.sh  
-rwxr-xr-x root/root 188 2021-11-21 23:23  
444.sh  
-rwxr-xr-x root/root 0 2021-11-21 23:19  
444.sh~
```

Рис.17 (END)

Вывод

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