

Презентация по лабораторной работе №11

Подъярова Ксения Витальевна

Российский Университет Дружбы Народов

Цель работы

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

Выполнение лабораторной работы

Задание 1

- 1) Используя команды `getopts` `grep`, написала командный файл, который анализирует командную строку с ключами:
 - `-iinputfile` — прочитать данные из указанного файла;
 - `-ooutputfile` — вывести данные в указанный файл;
 - `-р`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 `-р`. Для данной задачи я создала файл `progra1.sh` (рис. 1) и написала соответствующий скрипт (рис. 2) , (рис. 3)

```
kypodjhyarova@dk6n61 ~ $ touch progra1.sh
kypodjhyarova@dk6n61 ~ $ enacs &
[1] 6251
kypodjhyarova@dk6n61 ~ $ touch a1.txt a2.txt
kypodjhyarova@dk6n61 ~ $ chmod +x progra1.sh
kypodjhyarova@dk6n61 ~ $ cat a1.txt
kypodjhyarova@dk6n61 ~ $ enacs &
[2] 8637
kypodjhyarova@dk6n61 ~ $ cat a1.txt
hello
goodbye goodbye
```

```
#!/bin/bash
iflag=0; oflag=0; pflag=0; Cflag=0; nflag=0;
while getopts i:op:Cn 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 "Даблон не найден"
    exit
fi
if ((iflag==0))
then
    echo "Входящий файл не найден"
    exit
fi
if ((oflag==0))
then
    echo "Исходящий файл не найден"
    exit
fi
```

Figure 2: Скрипт

```
if ((nflag==0))
then
    if ((iflag==0))
    then
        grep $pval $ival > $oval
    else
        grep -i $pval $ival > $oval
    fi
else
    if ((iflag==0))
    then
        grep -n $pval $ival > $oval
    else
        grep -i -n $pval $ival > $oval
    fi
fi
```

Figure 3: Скрипт

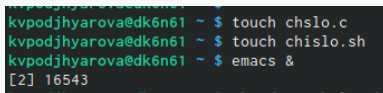
- 2) Проверила работу написанного скрипта, используя различные опции (например команду `./progra1.sh -i a1.txt -o a2.txt -C -n`), предварительно добавив право на исполнение файла (`chmod +x progra1.sh`) и создав 2 файла, которые необходимы для выполнения программы (`a1.txt`, `a2.txt`). Скрипт работает корректно.(рис. 4)

```
kvpodjhyarova@dk6n61 ~ $ ./progra1.sh -i a1.txt -o a2.txt -p hello -n
kvpodjhyarova@dk6n61 ~ $ cat a2.txt
1:hello
3:hello123
4:hello hello hello
kvpodjhyarova@dk6n61 ~ $ ./progra1.sh -i a1.txt -o a2.txt -p hello -C -n
kvpodjhyarova@dk6n61 ~ $ cat a2.txt
1:hello
3:hello123
4:hello hello hello
kvpodjhyarova@dk6n61 ~ $ ./progra1.sh -o a2.txt -p hello -C -n
Входящий файл не найден
```

Figure 4: Проверка работы

Задание 2

2. 1) Написала 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 должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено. Для данной задачи я создала 2 файла: `chslo.c` `chislo.sh` (рис. 5) и написала соответствующие скрипты (рис. 6) (рис. 7)



```
kvpodjhyarova@dk6n61 ~ $ touch chslo.c
kvpodjhyarova@dk6n61 ~ $ touch chislo.sh
kvpodjhyarova@dk6n61 ~ $ emacs &
[2] 16543
```

Figure 5: Создание файла


```
#!/bin/bash
gcc chslo.c -o chslo
./chslo
code=$?
case $code in
  0) echo "Число меньше 0";;
  1) echo "Число больше 0";;
  2) echo "Число равно 0";;
esac
```

j:*** chislo.sh All L10 (

Figure 6: Скрипт

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

j:--- chslo.c All L11

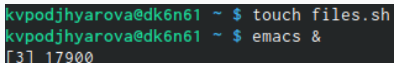
Figure 7: Скрипт

- 2) Проверила работу написанных скриптов (команда `./chislo.sh`), предварительно добавив право на исполнение файла (`chmod +x chislo.sh`). Скрипты работают корректно.(рис. 8)

```
kvpodjhyarova@dk6n61 ~ $ chmod +x chislo.sh
kvpodjhyarova@dk6n61 ~ $ ./chislo.sh
Введите число
0
Число равно 0
kvpodjhyarova@dk6n61 ~ $ ./chislo.sh
Введите число
9
Число больше 0
kvpodjhyarova@dk6n61 ~ $ ./chislo.sh
Введите число
-25
Число меньше 0
kvpodjhyarova@dk6n61 ~ $
```

Figure 8: Проверка работы

3. 1) Написала командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передается в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют). Для данной задачи я создала файл files.sh (рис. 9) и написала соответствующий скрипт (рис. 10)



```
kvpodjhyarova@dk6n61 ~ $ touch files.sh
kvpodjhyarova@dk6n61 ~ $ emacs &
[3] 17900
```

Figure 9: Создание файла

```
File Edit Options Buffers Tools Sn-Script Help
#!/bin/bash
opt=$1;
format=$2;
number=$3;
function Files()
{
    for ((i=1; i<=$number; i++)) do
        file=$(echo $format | tr '#' "$i")
        if [ $opt == "-r" ]
        then
            rm -f $file
        elif [ $opt == "-c" ]
        then
            touch $file
        fi
    done
}
Files
```

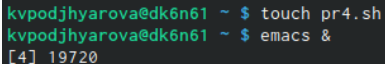
Figure 10: Скрипт

- 2) Далее я проверила работу написанного скрипта (./files.sh), предварительно добавив право на исполнение файла. Сначала я создала три файла, удовлетворяющих условию задач, а потом удалила их. Скрипт работает корректно (рис. 11)

```
root@kyanovskidm1:~# chmod +x files.sh
root@kyanovskidm1:~# ls
al.txt  abi1  backup  chilo.sh  code3.sh  files.sh  format.sh  games  lab07.sh  no-intro  public  test.txt  Документы  Оперативные
al.txt~ architecture_PC  backup.sh  chilo.sh~  code2.sh  far2.docx  far2.docx  lab  lab07.sh  no-intro  public.html  tmp  Журналы  "Рабочий стол"
al.txt~  architecture_PC  bin  chilo.c  code2.sh~  features  far2.docx  far  lab07.sh  no-intro  public.html  tmp  Журналы  "Рабочий стол"
root@kyanovskidm1:~# ./files.sh -r abc.txt 2
root@kyanovskidm1:~# ls
al.txt  abi1  backup  chilo.sh  code3.sh  files.sh  format.sh  games  lab07.sh  no-intro  public  test.txt  Документы  Оперативные
al.txt~ architecture_PC  backup.sh  chilo.sh~  code2.sh  far2.docx  far2.docx  lab  lab07.sh  no-intro  public.html  tmp  Журналы  "Рабочий стол"
al.txt~  architecture_PC  bin  chilo.c  code2.sh~  features  far2.docx  far  lab07.sh  no-intro  public.html  tmp  Журналы  "Рабочий стол"
root@kyanovskidm1:~# ./files.sh -r abc.txt 3
root@kyanovskidm1:~# ls
al.txt  abi1  backup  chilo.sh  code3.sh  files.sh  format.sh  games  lab07.sh  no-intro  public  test.txt  Документы  Оперативные
al.txt~ architecture_PC  backup.sh  chilo.sh~  code2.sh  far2.docx  far2.docx  lab  lab07.sh  no-intro  public.html  tmp  Журналы  "Рабочий стол"
al.txt~  architecture_PC  bin  chilo.c  code2.sh~  features  far2.docx  far  lab07.sh  no-intro  public.html  tmp  Журналы  "Рабочий стол"
root@kyanovskidm1:~#
```

Figure 11: Проверка работы

4. 1) Написала командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицировала его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду `find`). Для данной задачи я содала файл `pr4.sh` (рис. 12) и написала соответствующий скрипт (рис. 13)

A terminal window with a dark background and light green text. The prompt is 'kvpodjhyarova@dk6n61 ~'. The first command is '\$ touch pr4.sh' and the second is '\$ emacs &'. The output of the second command is '[4] 19720'.

```
kvpodjhyarova@dk6n61 ~ $ touch pr4.sh
kvpodjhyarova@dk6n61 ~ $ emacs &
[4] 19720
```

Figure 12: Создание файла

```
#!/bin/bash
files=$(find ./ -maxdepth 1 -mtime -7)
listing=""
for file in "$files"; do
    files=$(echo "$file" | cut -c 3-)
    listing="$listing $file"
done
dir=$(basename $(pwd))
tar -cvf $dir.tar $listing
```

Figure 13: Скрипт

- 2) Далее я проверила работу написанного скрипта, предварительно добавив право на исполнение файла и создав отдельный каталог с несколькими файлами ((рис. 11)

```
kvpodjhyarova@dk6n61 ~ $ chmod +x pr4.sh
[3]- Завершён      emacs
kvpodjhyarova@dk6n61 ~ $ mkdir Catalog1
kvpodjhyarova@dk6n61 ~ $ cd Catalog1
kvpodjhyarova@dk6n61 ~/Catalog1 $ ./pr4.sh
./
tar: ./Catalog1.tar: файл является архивом; не сброшен
kvpodjhyarova@dk6n61 ~/Catalog1 $ tar -tf Catalog1.tar
./
kvpodjhyarova@dk6n61 ~/Catalog1 $ ./pr4.sh
bash: ./pr4.sh: Нет такого файла или каталога
kvpodjhyarova@dk6n61 ~/Catalog1 $ ./pr4.sh
./
tar: ./Catalog1.tar: файл является архивом; не сброшен
tar: ./Catalog1.tar: файл является архивом; не сброшен
kvpodjhyarova@dk6n61 ~/Catalog1 $ tar -tf Catalog1.tar
./
kvpodjhyarova@dk6n61 ~/Catalog1 $
```

Figure 14: Проверка работы

Выводы

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