Отчёт по лабораторной работе №11

Дисциплина “Операционные системы”

Батова Ирина Сергеевна, НММбд-01-22

20 апреля 2023

Российский университет дружбы народов, Москва, Россия

Вводная часть

Цель работы

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

Основная часть

Программа 1

- В данном скрипте мы сначала вводим соответствующие опциям переменные и присваиваем этим переменным 0. Далее программа просматривает командную строку на наличие опций и присваивает 1 тем опциям (переменным), которые есть в командной строке. После этого мы используем команду if для проверки наличия различных опций. Сначала проверяем есть ли шаблон для поиска (слово, которое ищем), а потом есть ли файл, в котором будет искаться это слово. При невыполнении хотя бы одного из условий программа выводит ошибку. Далее мы перебираем различное сочетание опций и выводим соответствующие строки в файл.

Программа 1, архивация файла

```
#!/bin/bash

iflag=0; oflag=0; pflag=0; Cflag=0; nflag=0;
while getopts i:o:p:Cn opt
do case $opt in
       i) iflag=1; ival=$OPTARG;;
       o) oflag=1; oval=$OPTARG;;
       p) pflag=1; pval=$OPTARG;;
       C) Cflag=1;;
       n) nflag=1;;
       *) echo illegal option $opt
   esac
done

if (($pflag==0))
then echo "Не найден шаблон для поиска"
else
    if (($iflag==0))
    then echo "Не найден файл"
    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 > $oval
                  fi
             else if (($nflag==0))
                  then grep -i $pval $ival >$oval
                  else grep -i -n $pval $ival > $oval
                  fi

             fi
        fi
    fi
fi
```

Проверка программы 1

- Далее добавляем право на выполнение файла командой ’chmod +x *.sh’, создаем файл с текстом (test1.txt) и пустой файл, в который выводятся данные (test2.txt) и выполняем скрипт командой ‘./file21.sh (аргументы)’. Для проверки корректности выполнения просматриваем содержимое файла командой cat

Проверка программы 1

```
[isbatova@fedora ~]$ chmod +x *.sh
[isbatova@fedora ~]$ ./file21.sh -i test1.txt -o test2.txt -p Markdown -C -n
[isbatova@fedora ~]$ cat test2.txt
1:Markdown — язык разметки, используемый для создания форматированного текста.
3:Markdown содержит базовые элементы, которые можно найти почти в любом README.md.
4:В целом, Markdown используется для быстрого форматирования статьи для перевода в PDF.
[isbatova@fedora ~]$ cat test1.txt
Markdown — язык разметки, используемый для создания форматированного текста.
Почему так?
Markdown содержит базовые элементы, которые можно найти почти в любом README.md.
В целом, Markdown используется для быстрого форматирования статьи для перевода в PDF.
```

Программа 2 на языке Си

- В данном скрипте мы запрашиваем у пользователя число, затем программа считывает его и определяет, число больше нуля, равно нулю или меньше нуля.

Программа 2 на языке Си

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

int main()
{
  printf("Введите число\n");
  int a;
  scanf ("%b", &a);
  if (a<0) exit (0);
  if (a>0) exit (1);
  if (a==0) exit (2);
  return 0;
}
```

Программа 2

- В данном скрипте мы сначала компилируем файл с программой на языке Си в объектный файл. Далее вводим команды запуска этого объектного файла, затем программа анализирует, какое число получилось на выходе. После этого с помощью команды “case” выводим соответствующее сообщение.

```
#!/bin/bash
gcc file22.c -o file22
./file22
code=$?
case $code in
    0) echo "Введенное Вами число меньше 0";;
    1) echo "Введенное Вами число больше 0";;
    2) echo "Введенное Вами число равно 0"";;
esac
```

Проверка программы 2

- Далее добавляем право на выполнение файла командой ’chmod +x *.sh’ и выполняем скрипт командой ‘./file22.sh’. Для проверки корректности выполнения я ввела несколько чисел из разных диапазонов

```
[isbatova@fedora ~]$ chmod +x *.sh
[isbatova@fedora ~]$ ./file22.sh
Введите число
-2
Введенное Вами число меньше 0
[isbatova@fedora ~]$ ./file22.sh
Введите число
5
Введенное Вами число больше 0
[isbatova@fedora ~]$ ./file22.sh
Введите число
0
Введенное Вами число равно 0
```

Программа 3

- В данном скрипте мы вводим переменные для опций, которые будут введены пользователям - опция -r или -c (remove - удалить или create - создать), формат файла и количество файлов, которые нужно создать. После этого задаем функцию Func, которая будет удалять и создавать нужное количество файлов нужного формата в зависимости от аргументов, и запускаем ее внутри программы.

Программа 3

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

Проверка программы 3

- Далее добавляем право на выполнение файла командой ’chmod +x *.sh’ и выполняем скрипт командой ‘./file23.sh (аргументы)’. Для проверки корректности выполнения просматриваем содержимое каталога командой ls и после создания, и после удаления

Проверка программы 3

```
[isbatova@fedora ~]$ chmod +x *.sh
[isbatova@fedora ~]$ ./file23.sh -c lala#.txt 4
[isbatova@fedora ~]$ ls
 backup       file23.sh~    lala4.txt                      test2.txt
 file1.sh~    file2.sh~     os                             work
 file21.sh    file3.sh~     pandoc-3.0                     Видео
 file21.sh~   file4.sh~     pandoc-3.0-linux-amd64.tar.gz  Документы
 file22       forlab11      pandoc-crossref                Загрузки
 file22.c     lab07.sh~     pandoc-crossref.1              Изображения
 file22.c~    labpractice   pandoc-crossref-Linux.tar.xz   Музыка
 file22.sh    lala1.txt     stage2                         Общедоступные
 file22.sh~   lala2.txt     stage3.txt                    'Рабочий стол'
 file23.sh    lala3.txt     test1.txt                      Шаблоны
[isbatova@fedora ~]$ ./file23.sh -r lala#.txt 4
[isbatova@fedora ~]$ ls
 backup       file22.sh~    labpractice                    stage3.txt    Музыка
 file1.sh~    file23.sh     os                             test1.txt     Общедоступные
 file21.sh    file23.sh~    pandoc-3.0                     test2.txt    'Рабочий стол'
 file21.sh~   file2.sh~     pandoc-3.0-linux-amd64.tar.gz  work          Шаблоны
 file22       file3.sh~     pandoc-crossref                Видео
 file22.c     file4.sh~     pandoc-crossref.1              Документы
 file22.c~    forlab11      pandoc-crossref-Linux.tar.xz   Загрузки
 file22.sh    lab07.sh~     stage2                         Изображения
```

Программа 4

- В данном скрипте сначала с помощью команды find находим файлы, которые были изменены меньше недели назад и создаем переменную для списка файлов, которые будем архивировать. Далее с помощью цикла for анализируем каждый файл по времени создания и в зависимости от этого добавляем его в “список файлов” или нет. После окончания цикла архивируем все файлы, содержащиеся в списке.

Программа 4

```
#!/bin/bash
f=$(find ./ -maxdepth 1 -mtime -7)
list=""
for file in "$f" ; do
    file=$(echo "$file" | cut -c 3-)
    list="$list $file"

done
catalog=$(basename $(pwd))
tar -cvf $catalog.tar $list
```

Проверка программы 4

- Далее добавляем право на выполнение файла командой ’chmod +x *.sh’, переходим в специально созданный каталог ‘forlab11’, в который помещены файлы различное давности, и выполняем скрипт командой ‘sudo ~/file24.sh’. Для проверки корректности выполнения просматриваем содержимое каталога командой ls и видим, что был создан архив файлов

Проверка программы 4

```
[isbatova@fedora ~]$ chmod +x *.sh
[isbatova@fedora ~]$ cd ~/forlab11
[isbatova@fedora forlab11]$ ls
exp.doc  exp.pdf  file1.sh  file2.sh  file3.sh  file4.sh  lab07.sh  stage2  stage3.txt
[isbatova@fedora forlab11]$ sudo ~/file24.sh
file2.sh
file1.sh
exp.pdf
exp.doc
file4.sh
file3.sh
stage3.txt
[isbatova@fedora forlab11]$ ls
exp.doc  file1.sh  file3.sh  forlab11.tar  stage2
exp.pdf  file2.sh  file4.sh  lab07.sh      stage3.txt
```

Вывод

Вывод

В данной лабораторной работе мной были изучены основы программирования в оболочке ОС UNIX. Помимо этого, я научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.