

Отчёт по лабораторной работе № 12

Перелыгин Сергей Викторович

Содержание

| | |
|----------------------------------|----|
| 1 Цель работы | 4 |
| 2 Выполнение лабораторной работы | 5 |
| 3 Ответы на контрольные вопросы | 13 |
| 4 Выводы | 17 |
| 5 Библиография | 18 |

Список иллюстраций

| | |
|---------------------------------------|----|
| 2.1 Скрипт 1 | 6 |
| 2.2 Проверка скрипта 1 | 7 |
| 2.3 c1.c | 8 |
| 2.4 c1.sh | 8 |
| 2.5 Проверка скриптов | 9 |
| 2.6 ex3.sh | 10 |
| 2.7 Проверка скрипта ex3.sh | 11 |
| 2.8 ex4.sh | 12 |
| 2.9 Скрипт 4 | 12 |

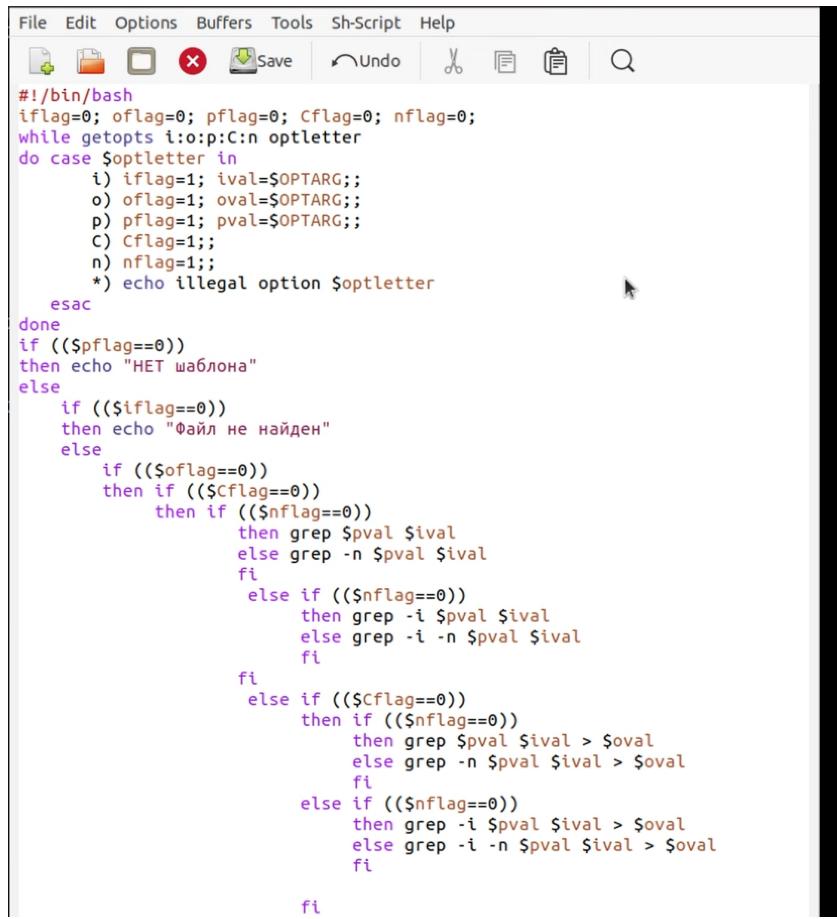
1 Цель работы

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

2 Выполнение лабораторной работы

1. Используя команды getopt grep, написал командный файл, который анализирует командную строку с ключами:

- -iinputfile - прочитать данные из указанного файла;
- -ooutputfile - вывести данные в указанный файл;
- -ршаблон - указать шаблон для поиска;
- -С - различать большие и малые буквы;
- -п - выдавать номера строк, а затем ищет в указанном файле нужные строки, определяемые ключом -р. Для данной задачи я создал файл ex1.sh(Рисунки 1) и написал соответствующие скрипты.



```
#!/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 "НЕТ шаблона"
else
    if ((${iflag==0}))
    then echo "Файл не найден"
    else
        if ((${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
```

Рис. 2.1: Скрипт 1

Далее я проверил работу написанного скрипта, используя различные опции (например, команда «./ex1.sh - I a1.txt-o a2.txt-p UNIX -C-n»), предварительно добавив право на исполнение файла (команда «chmod +x ex1.sh») и создав 2 файла, которые необходимы для выполнения программы: a1.txt и a2.txt(Рисунок 2)

```

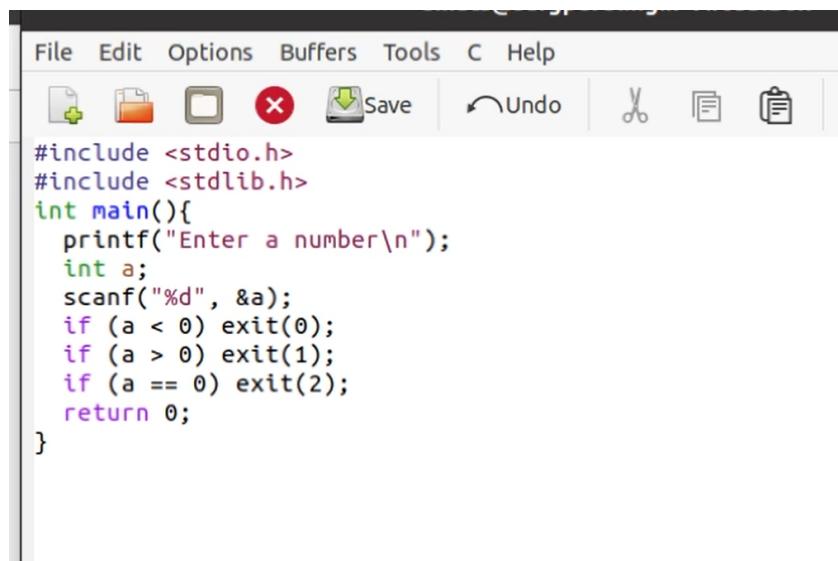
п
sergperelihgin@sergperelihgin-VirtualBox: $ cat a1.txt
Изучить основы программирования в оболочке ОС UNIX.
Научится писать более сложные командные файлы.
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -i a1.txt -o a2.txt -p UNIX -C -
п
sergperelihgin@sergperelihgin-VirtualBox: $ 
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -o a2.txt -p UNIX -C -n
Файл не найден
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -i a1.txt -o a2.txt -p UNIX -n
sergperelihgin@sergperelihgin-VirtualBox: $ cat a1.txt
Изучить основы программирования в оболочке ОС UNIX.
Hello world
Научится писать более сложные командные файлы.
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -i a1.txt -o a2.txt -p UNIX -C -
п
sergperelihgin@sergperelihgin-VirtualBox: $ cat a2.txt
Изучить основы программирования в оболочке ОС UNIX.
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -i a1.txt -o a2.txt -p UNIX -n
sergperelihgin@sergperelihgin-VirtualBox: $ cat a2.txt
1:Изучить основы программирования в оболочке ОС UNIX.
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -o a2.txt -p UNIX -C -n
Файл не найден
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -o a2.txt -p -C -n
Файл не найден
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -i a1.txt -p -C -n
grep: a1.txt: invalid context length argument
sergperelihgin@sergperelihgin-VirtualBox: $ ./ex1.sh -i a1.txt -C -n
НЕТ шаблона
sergperelihgin@sergperelihgin-VirtualBox: $ ■

```

Рис. 2.2: Проверка скрипта 1

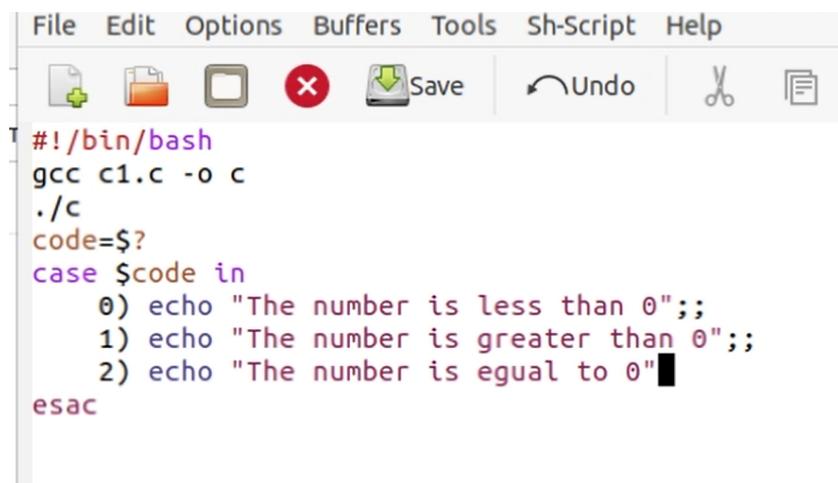
Скрипт работает корректно.

2. Написал 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 создал 2 файла: c1.c (Рисунок 3) и c1.sh(Рисунок 4)и написал соответствующие скрипты.



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

Рис. 2.3: c1.c



```
#!/bin/bash
gcc c1.c -o c
./c
code=$?
case $code in
    0) echo "The number is less than 0";;
    1) echo "The number is greater than 0";;
    2) echo "The number is equal to 0";;
esac
```

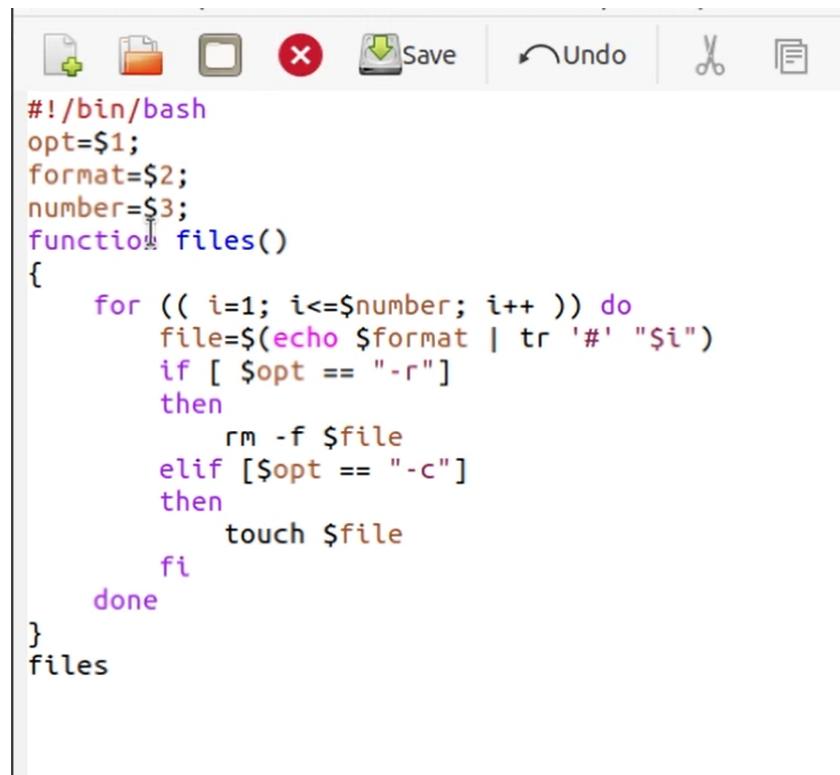
Рис. 2.4: c1.sh

Далее я проверил работу написанных скриптов (команда «./c1.sh»), предварительно добавив право на исполнение файла (команда «chmod +x c1.sh») (Рисунок 5). Скрипты работают корректно.

```
sergperelihgin@sergperelihgin-VirtualBox:~$ emacs &
[1] 17128
sergperelihgin@sergperelihgin-VirtualBox:~$ 
[1]+  Done                  emacs
sergperelihgin@sergperelihgin-VirtualBox:~$ chmod +x c1.sh
sergperelihgin@sergperelihgin-VirtualBox:~$ ./c1.sh
Enter a number
1
The number is greater than 0
sergperelihgin@sergperelihgin-VirtualBox:~$ ./c1.sh
Enter a number
0
The number is equal to 0
sergperelihgin@sergperelihgin-VirtualBox:~$ ./c1.sh
Enter a number
-3
The number is less than 0
sergperelihgin@sergperelihgin-VirtualBox:~$ 
```

Рис. 2.5: Проверка скриптов

3. Написал командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N. Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют). Для данной задачи я создал файл: ex3.sh(Рисунок 6) и написал соответствующий скрипт.



```
#!/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
```

Рис. 2.6: ex3.sh

Далее я проверил работу написанного скрипта, предварительно добавив право на исполнение файла (команда «chmod +x ex3.sh»). Сначала я создал три файла (команда «./ex3.sh -c abc#.txt 3»), удовлетворяющие условию задачи, а потом удалил их (команда «./ex3.sh -r abc#.txt 3»)(Рисунок7)

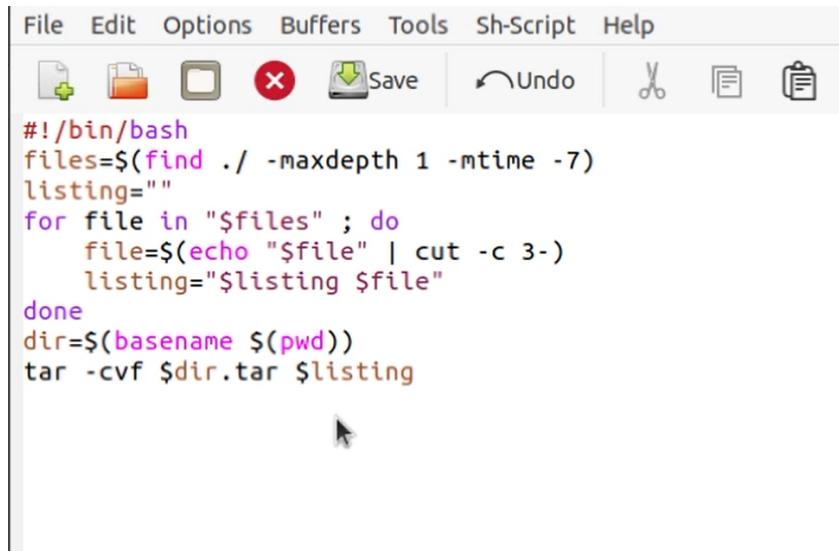
```

sergperelihgin@sergperelihgin-VirtualBox:~$ emacs &
[1] 17388
sergperelihgin@sergperelihgin-VirtualBox:~$ chmod +x ex3.sh
[1]+ Done                  emacs
sergperelihgin@sergperelihgin-VirtualBox:~$ 
sergperelihgin@sergperelihgin-VirtualBox:~$ chmod +x ex3.sh
sergperelihgin@sergperelihgin-VirtualBox:~$ ./ex3.sh -c abc#.txt 3
sergperelihgin@sergperelihgin-VirtualBox:~$ ls
a1.txt          Desktop      my_os
a2.txt          Documents    pandoc-crossref
abc1.txt        Downloads    Pictures
abc2.txt        equipment   play
abc3.txt        ex1.sh       Public
academic-laboratory-report-template ex1.sh~    reports
academic-presentation-markdown-template ex2.sh~    ski.plases
australia        ex3.sh       snap
backup          ex3.sh~    Templates
backup.sh~      ex4.sh       test
c               feathers    Videos
c1.c           lab2        work
c1.c~          monthly     +x
c1.sh          morefunnew  лаба10
c1.sh~          Music       'лаба9 (1)'
Desktop         my_os      [ ]
Documents
sergperelihgin@sergperelihgin-VirtualBox:~$ ./ex3.sh -r abc#.txt 3
sergperelihgin@sergperelihgin-VirtualBox:~$ ls
a1.txt          Downloads    pandoc-crossref
a2.txt          equipment   Pictures
academic-laboratory-report-template ex1.sh     play
academic-presentation-markdown-template ex1.sh~    Public
australia        ex2.sh     reports
backup          ex3.sh     ski.plases
backup.sh~      ex3.sh~    snap
c               feathers    Templates
c1.c           lab2       Videos
c1.c~          monthly     work
c1.sh          morefunnew +x
c1.sh~          Music      лаба10
Desktop         my_os      [ ] 'лаба9 (1)'
Documents
sergperelihgin@sergperelihgin-VirtualBox:~$ █

```

Рис. 2.7: Проверка скрипта ex3.sh

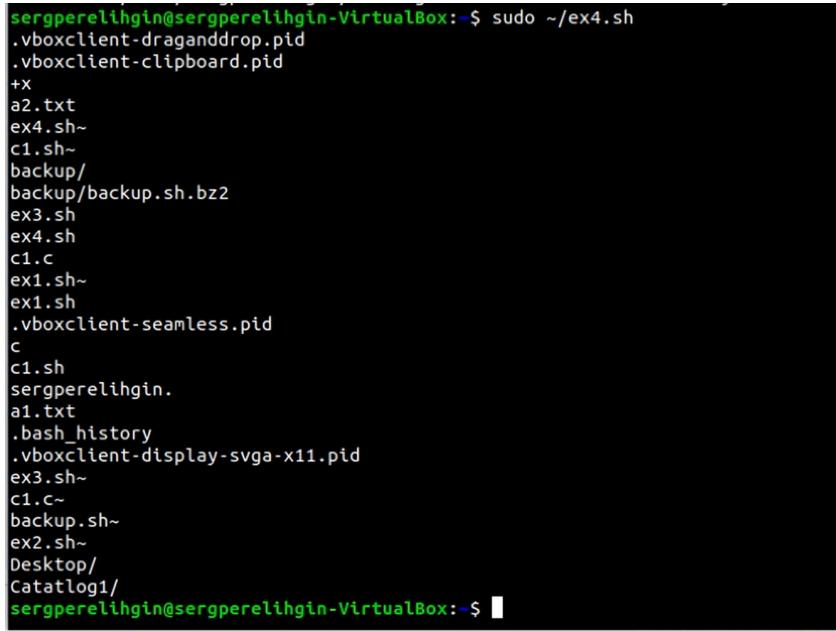
4. Написал командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировал его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find). Для данной задачи я создал файл: ex4.sh(Рисунок 8) и написал соответствующий скрипт.



```
#!/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. 2.8: ex4.sh

Далее я проверил работу написанного скрипта, предварительно добавив право на исполнение файла (команда «chmod +x ex4.sh») и создав отдельный Catalog1 с несколькими файлами. Скрипт работает корректно.



```
sergperelihgin@sergperelihgin-VirtualBox:~$ sudo ~/ex4.sh
.vboxclient-draganddrop.pid
.vboxclient-clipboard.pid
+x
a2.txt
ex4.sh~
c1.sh~
backup/
backup/backup.sh.bz2
ex3.sh
ex4.sh
c1.c
ex1.sh~
ex1.sh
.vboxclient-seamless.pid
c
c1.sh
sergperelihgin.
a1.txt
.bash_history
.vboxclient-display-svga-x11.pid
ex3.sh~
c1.c~
backup.sh~
ex2.sh~
Desktop/
Catalog1/
sergperelihgin@sergperelihgin-VirtualBox:~$ █
```

Рис. 2.9: Скрипт 4

3 Ответы на контрольные вопросы

- 1) Команда getopt осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных. Синтаксис команды следующий:

getopt option-string variable [arg...]

Флаги - это опции командной строки, обычно помеченные знаком минус; Например, для команды ls флагом может являться -F.

Строка опций option-string - это список возможных букв и чисел соответствующего флага. Если ожидается, что некоторый флаг будет сопровождаться некоторым аргументом, то за символом, обозначающим этот флаг, должно следовать двоеточие. Соответствующей переменной присваивается буква данной опции. Если команда getopt может распознать аргумент, то она возвращает истину. Принято включать getopt в цикл while и анализировать введённые данные с помощью оператора case.

Функция getopt включает две специальные переменные среды -OPTARG и OPTIND. Если ожидается дополнительное значение, то OPTARG устанавливается в значение этого аргумента.

Функция getopt также понимает переменные типа массив, следовательно, можно использовать её в функции не только для синтаксического анализа аргументов функций, но и для анализа введённых пользователем данных.

- 2) При перечислении имён файлов текущего каталога можно использовать следующие символы:

1. *-соответствует произвольной, в том числе и пустой строке;

2. ?-соответствует любому одинарному символу;
3. [c1-c2] - соответствует любому символу, лексикографически находящемуся между символами c1 и c2.

Например,

- 1.1. echo* - выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды ls;
- 1.2. ls*.c-выведет все файлы с последними двумя символами, совпадающими с.c.
- 1.3. echoprog.? -выведет все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются prog..
- 1.4. [a-z]*-соответствует произвольному имени файла в текущем каталоге, начинаящемуся с любой строчной буквы латинского алфавита.

- 3) Часто бывает необходимо обеспечить проведение каких-либо действий циклически и управление дальнейшими действиями в зависимости от результатов проверки некоторого условия. Для решения подобных задач язык программирования bash предоставляет возможность использовать такие управляющие конструкции, как for, case, if и while. С точки зрения командного процессора эти управляющие конструкции являются обычными командами и могут использоваться как при создании командных файлов, так и при работе в интерактивном режиме. Команды, реализующие подобные конструкции, по сути, являются операторами языка программирования bash. Поэтому при описании языка программирования bash термин оператор будет использоваться наравне с термином команда.

Команды ОСUNIX возвращают код завершения, значение которого может быть использовано для принятия решения о дальнейших действиях.

Команда test, например, создана специально для использования в командных файлах. Единственная функция этой команды заключается в выработке кода завершения.

4) Два несложных способа позволяют вам прерывать циклы в оболочке bash.

Команда break завершает выполнение цикла, а команда continue завершает данную итерацию блока операторов.

Команда break полезна для завершения цикла while в ситуациях, когда условие перестаёт быть правильным.

Команда continue используется в ситуациях, когда больше нет необходимости выполнять блок операторов, но вы можете захотеть продолжить проверять данный блок на других условных выражениях.

5) Следующие две команды OCUNIX используются только совместно с управляющими конструкциями языка программирования bash: это команда true, которая всегда возвращает код завершения, равный нулю(т.е.истина), и команда false, которая всегда возвращает код завершения,неравный нулю(т.е.ложь).Примеры бесконечных циклов:

```
while true
```

```
do echo hello andy
```

```
done
```

```
until false
```

```
do echo hello mike
```

```
done
```

6) Стока iftest-fmans/i.sïðîâåð,,åò, ïóùåñòåóåòëèôàéëmans/i.s и является ли этот файл обычным файлом.Если данный файл является каталогом,то команда вернет нулевое значение(ложь).

7) Выполнение оператора цикла while сводится к тому,что сначала выполняется последовательность команд(операторов),которую задаёт список-команд в строке,содержащей служебное слово while,а затем,если последняя выполненная

команда из этой последовательности команд возвращает нулевой код завершения(истина),выполняется последовательность команд(операторов),которую задаёт список-команд в строке,содержащей служебное слово do,после чего осуществляется безусловный переход на начало оператора цикла while. Выход из цикла будет осуществлён тогда,когда последняя выполненная команда из последовательности команд (операторов),которую задаёт список-команд в строке,содержащей служебное слово while, возвратит ненулевой код завершения(ложь).

При замене в операторе цикла while служебного слова while на until условие,при выполнении которого осуществляется выход из цикла,меняется на противоположное.В остальном оператор цикла while и оператор цикла until идентичны.

4 Выводы

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

5 Библиография

- Кулябов Д.С. Операционные системы: лабораторные работы: учебное пособие / Д.С. Кулябов, М.Н. Геворкян, А.В. Королькова, А.В. Демидова. — М. : Изд-во РУДН, 2016. — 117 с. — ISBN 978-5-209-07626-1 : 139.13; То же [Электронный ресурс]. — URL: <http://lib.rudn.ru/MegaPro2/Download/MObject/6118>.
- Робачевский А.М. Операционная система UNIX [текст] : Учебное пособие / А.М. Робачевский, С.А. Немлюгин, О.Л. Стесик. — 2-е изд., перераб. и доп. — СПб. : БХВ-Петербург, 2005, 2010. — 656 с. : ил. — ISBN 5-94157-538-6 : 164.56. (ET 60)
- Таненбаум Эндрю. Современные операционные системы [Текст] / Э. Таненбаум. — 2-е изд. — СПб. : Питер, 2006. — 1038 с. : ил. — (Классика Computer Science). — ISBN 5-318-00299-4 : 446.05. (ET 50)
- Ван Стеен М., Эндрю Таненбаум Распределенные системы. Принципы и парадигмы [Текст] / Э. Таненбаум, в.М. Стеен. — СПб. : Питер, 2003. — 877 с. : ил. — (Классика Computer science). — ISBN 5-272-00053-6 : 377.52. (ET 50)
- Сафонов, В.О. Основы современных операционных систем : учебное пособие / В.О. Сафонов. — Москва : Интернет-Университет Информационных Технологий, 2011. — 584 с. — (Основы информационных технологий). — ISBN 978-5-9963-0495-0 ; То же [Электронный ресурс]. — URL: <http://biblioclub.ru/index.php?page=book&id=233210>.
- Немет Эви. UNIX — руководство системного администратора [Текст] / Э. Немет, Г. Снайдер, С. Сибасс; Э.Немет, Г.Снайдер, С.Сибасс, Х.Р.Трент. — 3-е изд. — СПб. : Питер, 2004. — 925 с. : ил. — (Для профессионалов). — ISBN

0-13-020601-6. — ISBN 5-318-00754-6 : 280.00. (ET 30)

- Бек Л. Введение в системное программирование [Текст] / Л. Бек; Пер. с англ. Н.А.Богомолова, В.М.Вязовского и С.Е.Морковина; Под ред. Л.Н.Королева. — М. : Мир, 1988. — 448 с. : ил. — ISBN 5-03-000011-9 : 2.60. (ET 3)
- Дьяконов Владимир Юрьевич. Системное программирование [Текст] : Учебное пособие для втузов / В.Ю. Дьяконов, В.А. Китов, И.А. Калинчев; Под ред. А.Л.Горелика. — М. : Высшая школа, 1990. — 221 с. : ил. — ISBN 5-06-000732-4 : 0.55.