

Отчет по лабораторной работе №11

Операционные системы

Ничипорова Елена Дмитриевна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	13
	Список литературы	14

Список иллюстраций

2.1	создание файла	6
2.2	скрипт №1	7
2.3	предоставление прав доступа	7
2.4	проверка работы программы	8
2.5	создание файлов	8
2.6	работа с файлом chislo.c	9
2.7	работа с файлом chislo.sh	9
2.8	Проверка скрипта №2	10
2.9	создание файла	10
2.10	скрипт №3	11
2.11	Проверка скрипта №3	11
2.12	создание файлов	12
2.13	скрипт №4	12
2.14	создание каталога и предоставление прав доступа	12
2.15	проверка скрипта №4	12

Список таблиц

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 `getopts` `grep`, написала 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. Для данной задачи создала файл `prog1.sh` (рис. 2.1) и написала следующие алгоритмы (рис. 2.2)

```
ednichiporova@dk6n64 ~ $ touch prog1.sh
ednichiporova@dk6n64 ~ $ emacs &
```

Рис. 2.1: создание файла

```
#!/bin/bash
iflag=0; oflag=0; pflag=0; Cflag=0; nflag=0;
while getopts i:o:p: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 "шаблон не найден"
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. 2.2: скрипт №1

- проверила работу написанного скрипта, используя различные опции, предварительно добавив права на выполнение(рис. 2.3) и создав два файла, которые необходимы для выполнения программы. Скрипт работает корректно(рис. 2.4)

```
ednichiporova@dk6n64 ~ $ touch a1.txt a2.txt
ednichiporova@dk6n64 ~ $ chmod +x prog1.sh
```

Рис. 2.3: предоставление прав доступа

```

ednichiporova@dk6n64 ~ $ cat a1.txt
water abc
abc
prog1
water water
ednichiporova@dk6n64 ~ $ ./prog1.sh -i a1.txt -o a2.txt -p water -n
ednichiporova@dk6n64 ~ $ cat a2.sh
cat: a2.sh: Нет такого файла или каталога
ednichiporova@dk6n64 ~ $ cat a2.txt
1:water abc
4:water water
ednichiporova@dk6n64 ~ $ ./prog1.sh -i a1.txt -o a2.txt -p water -n
ednichiporova@dk6n64 ~ $ cat a2.txt
1:water abc
4:water water
ednichiporova@dk6n64 ~ $ ./prog1.sh -i a1.txt -C -n
шаблон не найден
ednichiporova@dk6n64 ~ $ ./prog1.sh -i a2.txt -p water -C -n
1:1:water abc

```

Рис. 2.4: проверка работы программы

2. 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 (рис. 2.5) и написала соответствующие скрипты (рис. 2.6) (рис. 2.7)

```

ednichiporova@dk6n64 ~ $ touch chislo.c
ednichiporova@dk6n64 ~ $ touch chislo.sh

```

Рис. 2.5: создание файлов


```

#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;
}

```

Рис. 2.6: работа с файлом chislo.c

```

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

```

Рис. 2.7: работа с файлом chislo.sh

- Проверила работу написанных скриптов. Скрипты работают корректно(рис. 2.8)

```

ednichiporova@dk6n64 ~ $ chmod +x chislo.sh
ednichiporova@dk6n64 ~ $ ./chislo.sh
введите число
5
число больше нуля
ednichiporova@dk6n64 ~ $ ./chislo.sh
введите число
0
число равно нулю
ednichiporova@dk6n64 ~ $ ./chislo.sh
введите число
-2
число меньше нуля
ednichiporova@dk6n64 ~ $

```

Рис. 2.8: Проверка скрипта №2

3. 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, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют). Для данной задачи я создала новый файл (рис. 2.9) и написала соответствующий скрипт (рис. 2.10)

```

ednichiporova@dk6n64 ~ $ touch files.sh
ednichiporova@dk6n64 ~ $ emacs &

```

Рис. 2.9: создание файла

```
#!/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. 2.10: скрипт №3

- Проверила работу написанного скрипта, предварительно добавив право на исполнение(рис. 2.11)

```
ednichiporova@dk6n64 ~ $ chmod +x files.sh
ednichiporova@dk6n64 ~ $ ls
-
1.txt
'2022-04-20 15:56:54.mkv'
a1.txt
a2.txt
abc
abcl
addition.txt
Architecture_PC
australia
backup
backup.sh
chislo
chislo.c
chislo.sh
conf.txt
example1.txt
example2.txt
example3.txt
example4.txt
file1.doc
file2.doc
file.pdf
files.sh
feathers
file.txt
games
GNUstep
lab07.sh
lab07.sh~
math
may
monthly
my_os
play
prog1.sh
prog1.sh~
prog2.sh
prog2.sh~

ednichiporova@dk6n64 ~ $ ./files.sh -r abc#.txt 3
./files.sh: строка 9: [: отсутствует символ «]»
./files.sh: строка 9: [: отсутствует символ «]»
./files.sh: строка 9: [: отсутствует символ «]»
ednichiporova@dk6n64 ~ $ ./files.sh -r abc#.txt 3
ednichiporova@dk6n64 ~ $ ls
-
1.txt
'2022-04-20 15:56:54.mkv'
a1.txt
a2.txt
abc
abcl
addition.txt
Architecture_PC
australia
backup
backup.sh
chislo
chislo.c
chislo.sh
conf.txt
example1.txt
example2.txt
example3.txt
example4.txt
file1.doc
file2.doc
file.pdf
files.sh
feathers
file.txt
games
GNUstep
lab07.sh
lab07.sh~
math
may
monthly
my_os
play
prog1.sh
prog1.sh~
prog2.sh
prog2.sh~

ednichiporova@dk6n64 ~ $ ./files.sh -c abc#.txt 3
ednichiporova@dk6n64 ~ $ ls
-
1.txt
'2022-04-20 15:56:54.mkv'
a1.txt
a2.txt
abc
abcl
addition.txt
Architecture_PC
australia
backup
backup.sh
chislo
chislo.c
chislo.sh
conf.txt
example1.txt
example2.txt
example3.txt
example4.txt
file1.doc
file2.doc
file.pdf
files.sh
feathers
file.txt
games
GNUstep
lab07.sh
lab07.sh~
math
may
monthly
my_os
play
prog1.sh
prog1.sh~
prog2.sh
prog2.sh~
```

Рис. 2.11: Проверка скрипта №3

4. 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 программы я создала

новый файл (рис. 2.12) и написала соответствующий скрипт(рис. 2.13)

```
ednichiporova@dk6n64 ~ $ touch prog.sh
ednichiporova@dk6n64 ~ $ emacs &
```

Рис. 2.12: создание файлов

```
#!/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. 2.13: скрипт №4

- Проверила работу скрипта, предварительно добавив право на исполнение и создала новый каталог с несколькими файлами (рис. 2.14). Скрипт работает корректно (рис. 2.15)

```
ednichiporova@dk6n64 ~ $ chmod +x prog.sh
ednichiporova@dk6n64 ~ $ mkdir catalog1
ednichiporova@dk6n64 ~ $ cd ~/catalog1
```

Рис. 2.14: создание каталога и предоставление прав доступа

```
ednichiporova@dk6n64 ~/catalog1 $ ls -l
итого 2
-rw-r--r-- 1 ednichiporova studsci 32 мая 19 14:26 a1.txt
-rw-r--r-- 1 ednichiporova studsci 26 мая 19 14:29 a2.txt
ednichiporova@dk6n64 ~/catalog1 $ ~/prog.sh
a1.txt
a2.txt
ednichiporova@dk6n64 ~/catalog1 $ tar -tf catalog1.tar
a1.txt
a2.txt
ednichiporova@dk6n64 ~/catalog1 $
```

Рис. 2.15: проверка скрипта №4

3 Выводы

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

Список литературы