

# Лабораторная работа №12

---

Бондарь Алексей Олегович НПМбд-01-20<sup>1</sup>

10 мая, 2021, Москва, Россия

<sup>1</sup>RUDN University, Moscow, Russian Federation

## Цель работы

---

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

## Задачи

---

1. Познакомиться с логическими управляющими конструкцией и циклов.
2. В ходе работы написать 4 командных файла.
3. Выполнить отчет.

## Выполнение лабораторной работы

---

- 1) Используя команды `getopts` `grep`, написал 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 создал файл `prog1.sh` и написал соответствующие скрипты. (рис. 1)

# Выполнение лабораторной работы

```
#!/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
```



Далее я проверил работу написанного скрипта, используя различные опции (например, команда «./prog.sh -i a1.txt -o a2.txt -p capital -C -n»), предварительно добавив право на исполнение файла (команда «chmod +x prog1.sh») и создав 2 файла, которые необходимы для выполнения программы: a1.txt и a2.txt. Скрипт работает корректно. (рис. 2):

## Выполнение лабораторной работы

```
aabondarj@dk8n74 ~ $ chmod +x prog1.sh
aabondarj@dk8n74 ~ $ cat a1.txt
Moscow is the capital of Russia
Paris is the Capital of France
Simple text
Rome is not the CAPITAL of Canada
aabondarj@dk8n74 ~ $ ./prog1.sh -i a1.txt -o a2.txt -p capital -C -n
aabondarj@dk8n74 ~ $ cat a2.txt
1:Moscow is the capital of Russia
2:Paris is the Capital of France
4:Rome is not the CAPITAL of Canada
aabondarj@dk8n74 ~ $ ./prog1.sh -i a1.txt -o a2.txt -p capital -n
aabondarj@dk8n74 ~ $ cat a2.txt
1:Moscow is the capital of Russia
aabondarj@dk8n74 ~ $ ./prog1.sh -i a1.txt -C -n
Шаблон не найден
aabondarj@dk8n74 ~ $ ./prog1.sh -o a2.txt -p capital -C -n
Файл не найден
aabondarj@dk8n74 ~ $ █
```

Figure 2: Проверка работы скрипта

- 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: `chislo.c` и `chislo.sh` и написала соответствующие скрипты. (рис. 3) (рис. 4)

```
#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;
}
```

Figure 3: Файл

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

Figure 4: Файл

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

```
aabondarj@dk6n64 ~ $ ./chislo.sh
Введите число
0
Число равно 0
aabondarj@dk6n64 ~ $ ./chislo.sh
Введите число
5
Число больше 0
aabondarj@dk6n64 ~ $ ./chislo.sh
Введите число
-5
Число меньше 0
```

Figure 5: Проверка работы скриптов

- 3) Написал командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (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 создал файл files.sh и написала соответствующий скрипт. (рис. 6)



```
#!/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 6: Третий скрипт

Далее я проверил работу написанного скрипта (команда «./files.sh»), предварительно добавив право на исполнение файла (команда «chmod +x files.sh»). Сначала я создал три файла (команда «./files.sh -c abc#.txt 3»), удовлетворяющие условию задачи, а потом удалил их (команда «./files.sh -r abc#.txt 3»). (рис. 7) , (рис. 8) , (рис. 9)

## Выполнение лабораторной работы

```
aabondarj@dk6n64 ~ $ chmod +x files.sh
aabondarj@dk6n64 ~ $ ls
'2021-05-14 14-28-02.mkv'    lab07
a1.txt                      lab07.asm
a2.txt                      '#lab10.sh#'
ARA.cpp                     lab10.sh
asdfg                       lab2
asdfg.asm                   lab2.asm
backup                       lab.asm
backup.sh                   labor
backup.sh~                  newdir
chislo                      prog1.sh
chislo.c                    prog1.sh~
chislo.c~                   prog2.sh
chislo.sh                   prog2.sh~
chislo.sh~                  '#progl.s.sh#'
```

Figure 7: Проверка работы скрипта

```
aabondarj@dk6n64 ~ $ ./files.sh -c abc#.txt 3
aabondarj@dk6n64 ~ $ ls
'2021-05-14 14-28-02.mkv'  lab06
a1.txt                    lab06.asm
a2.txt                    lab07
abc1.txt                  lab07.asm
abc2.txt                  '#lab10.sh#'
abc3.txt                  lab10.sh
ARA.cpp                   lab2
asdfg                     lab2.asm
asdfg.asm                 lab.asm
backup                     labor
backup.sh                 newdir
```

Figure 8: Проверка работы скрипта

```
aabondarj@dk6n64 ~ $ ./files.sh -r abc#.txt 3
aabondarj@dk6n64 ~ $ ls
'2021-05-14 14-28-02.mkv'    lab07
a1.txt                      lab07.asm
a2.txt                      '#lab10.sh#'
ARA.cpp                     lab10.sh
asdfg                       lab2
asdfg.asm                   lab2.asm
backup                      lab.asm
backup.sh                   labor
backup.sh~                  newdir
chislo                      prog1.sh
chislo.c                    prog1.sh~
chislo.c~                   prog2.sh
chislo.sh                   prog2.sh~
```

Figure 9: Проверка работы скрипта

- 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 `prog4.sh` и написал соответствующий скрипт. (рис. 10)

```
#!/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
```

Figure 10: Четвертый скрипт

Далее я проверил работу написанного скрипта (команды «./prog4.sh» и «tar -tf catalog.tar»), предварительно добавив право на исполнение файла (команда «chmod +x prog4.sh») и создав отдельный catalog с несколькими файлами. Скрипт работает корректно.(рис. 11) , (рис. 12)



```
aabondarj@dk6n64 ~ $ chmod +x *.sh
aabondarj@dk6n64 ~ $ mkdir lab12
aabondarj@dk6n64 ~ $ cd lab12
aabondarj@dk6n64 ~/lab12 $ ls -l
итого 0
aabondarj@dk6n64 ~/lab12 $ ls -l
итого 13
-rw-r--r-- 1 aabondarj studsci 109 мая 28 14:28 a1.txt
-rw-r--r-- 1 aabondarj studsci 32 мая 28 14:32 a2.txt
-rwxr-xr-x 1 aabondarj studsci 8072 мая 28 14:45 chislo
-rw-r--r-- 1 aabondarj studsci 196 мая 28 14:44 chislo.c
-rwxr-xr-x 1 aabondarj studsci 193 мая 28 14:42 chislo.sh
-rwxr-xr-x 1 aabondarj studsci 208 мая 28 15:11 prog4.sh
```

Figure 11: Проверка работы скрипта

```
aabondarj@dk6n64 ~/lab12 $ ./prog4.sh
a1.txt
chislo
chislo.c
chislo.sh
prog4.sh
a2.txt
files.sh
aabondarj@dk6n64 ~/lab12 $ tar -tf lab12.tar
a1.txt
chislo
chislo.c
chislo.sh
prog4.sh
a2.txt
files.sh
```

Figure 12: Проверка работы скрипта

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.