

Лабораторная работа № 11

**Программирование в командном процессоре ОС UNIX. Ветвления и
циклы**

Тарутина Кристина Олеговна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	8
	Список литературы	9

Список иллюстраций

2.1	программа 1	6
2.2	программа 2	7
2.3	программа 3	7

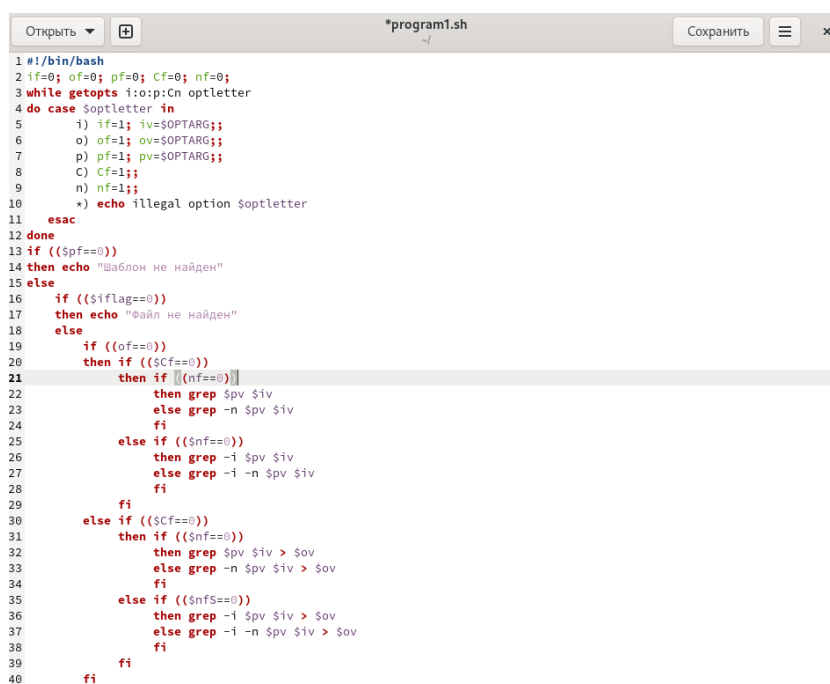
Список таблиц

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

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 (рис. 2.1).

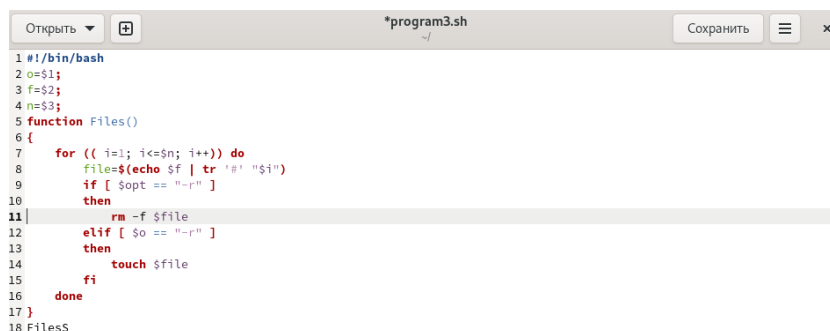


```
1 #!/bin/bash
2 if=0; of=0; pf=0; Cf=0; nf=0;
3 while getopts i:o:p:Cn optletter
4 do case $optletter in
5     i) if=1; iv=$OPTARG;;
6     o) of=1; ov=$OPTARG;;
7     p) pf=1; pv=$OPTARG;;
8     C) Cf=1;;
9     n) nf=1;;
10    *) echo illegal option $optletter
11    esac
12 done
13 if (($pf==0))
14 then echo "Шаблон не найден"
15 else
16     if (($iflag==0))
17     then echo "Файл не найден"
18     else
19         if (($of==0))
20         then if (($Cf==0))
21             then if (($nf==0))
22                 then grep $pv $iv
23                 else grep -n $pv $iv
24                 fi
25             else if (($nf==0))
26                 then grep -i $pv $iv
27                 else grep -i -n $pv $iv
28                 fi
29             fi
30         else if (($Cf==0))
31             then if (($nf==0))
32                 then grep $pv $iv > $ov
33                 else grep -n $pv $iv > $ov
34                 fi
35             else if (($nf==0))
36                 then grep -i $pv $iv > $ov
37                 else grep -i -n $pv $iv > $ov
38                 fi
39             fi
40     fi
```

Рис. 2.1: программа 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.).

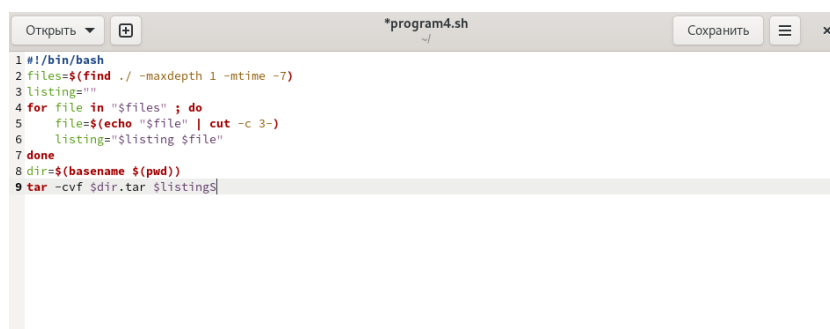
Число файлов, которые необходимо создать, передаётся в аргументы командной строки (рис. 2.2).



```
1 #!/bin/bash
2 o=$1;
3 f=$2;
4 n=$3;
5 function Files()
6 {
7     for (( i=1; i<=n; i++)) do
8         file=$(echo $f | tr 'a' " $i")
9         if [ $(opt == "-r") ]
10            then
11                rm -f $file
12            elif [ $o == "-r" ]
13                then
14                    touch $file
15            fi
16        done
17    }
18 Files
```

Рис. 2.2: программа 2

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)(рис. 2.3).



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

Рис. 2.3: программа 3

3 Выводы

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

Список литературы