

# Лабораторная работа №11

## Дисциплина: Операционные системы

Маслова Анастасия Сергеевна

### Содержание

Цель работы .....	1
Задание .....	1
Выполнение лабораторной работы .....	2
Выводы .....	7
Список литературы .....	7

### Цель работы

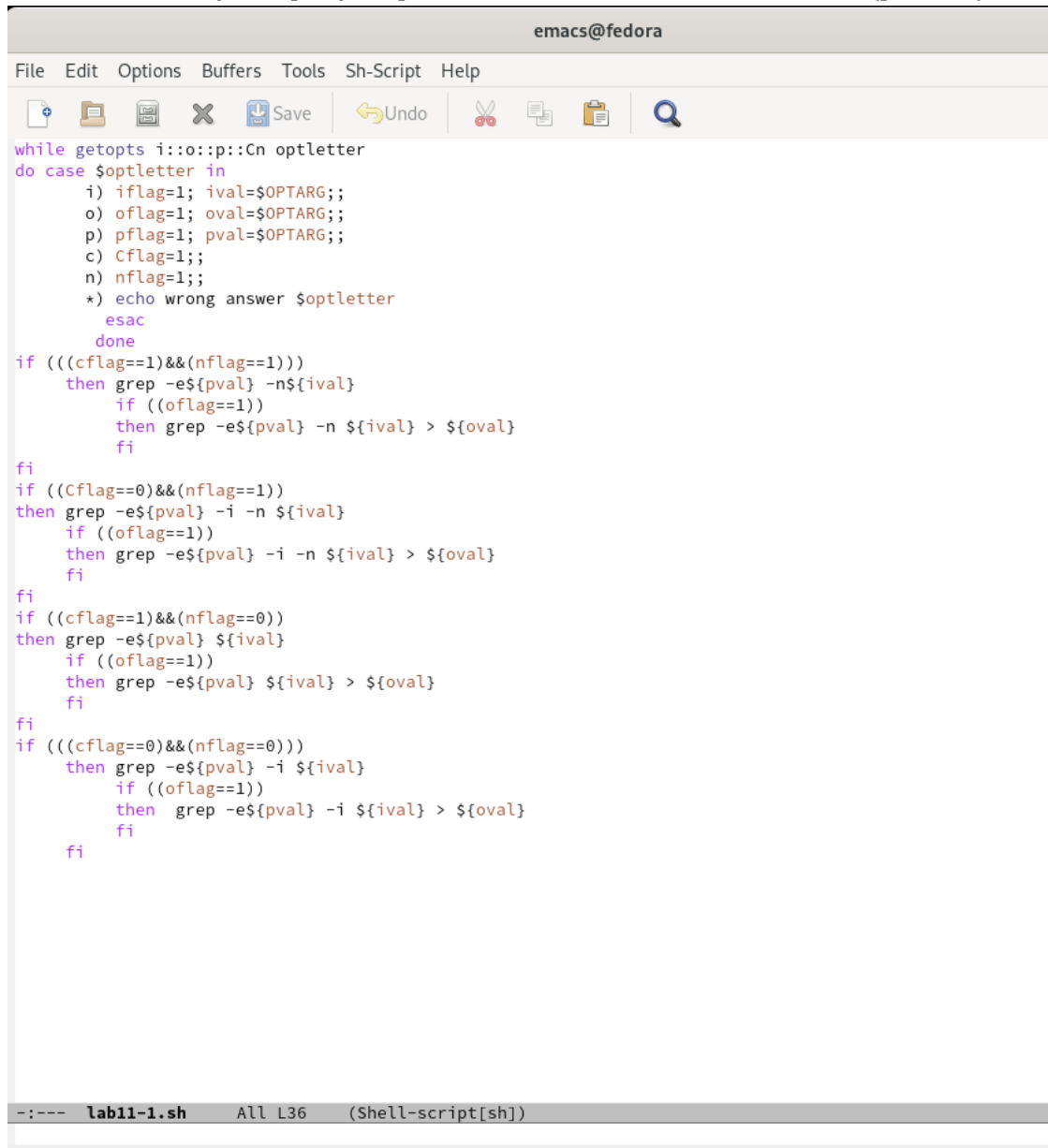
Изучить основы программирования в оболочке ОС UNIX. Научится писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

### Задание

1. Используя команды `getopts` `grep`, написать 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`.
2. Написать 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.
3. Написать 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).

## Выполнение лабораторной работы

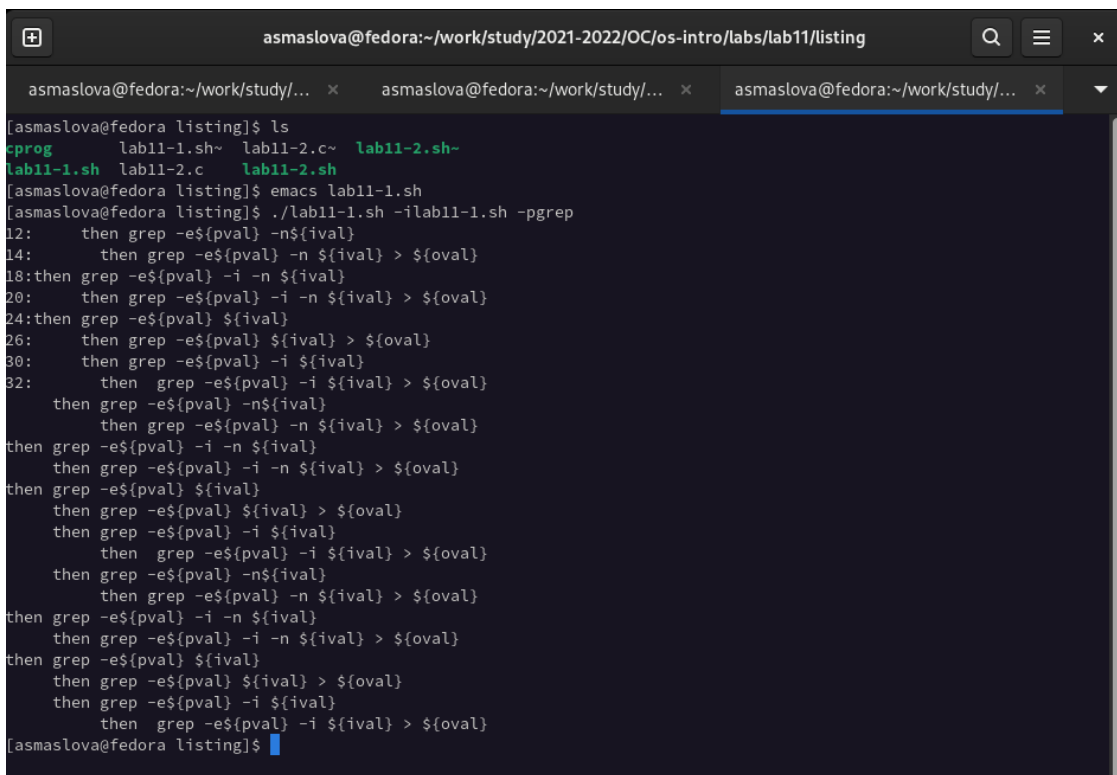
1. Используя команды getopt и grep, я написала командный файл, анализирующий командную строку с приведенными в задании ключами (рис.1-2).



```
emacs@fedora
File Edit Options Buffers Tools Sh-Script Help
Save Undo
while getopt 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 wrong answer $optletter
  esac
done
if ((Cflag==1)&&(nflag==1)))
then grep -e${pval} -n${ival}
  if ((oflag==1))
  then grep -e${pval} -n ${ival} > ${oval}
  fi
fi
if ((Cflag==0)&&(nflag==1))
then grep -e${pval} -i -n ${ival}
  if ((oflag==1))
  then grep -e${pval} -i -n ${ival} > ${oval}
  fi
fi
if ((Cflag==1)&&(nflag==0))
then grep -e${pval} ${ival}
  if ((oflag==1))
  then grep -e${pval} ${ival} > ${oval}
  fi
fi
if ((Cflag==0)&&(nflag==0))
then grep -e${pval} -i ${ival}
  if ((oflag==1))
  then grep -e${pval} -i ${ival} > ${oval}
  fi
fi
fi

-:--- lab11-1.sh All L36 (Shell-script[sh])
```

рис.1 Код программы 1



```

[asmaslova@fedora listing]$ ls
cprog      lab11-1.sh~  lab11-2.c~  lab11-2.sh~
lab11-1.sh  lab11-2.c   lab11-2.sh
[asmaslova@fedora listing]$ emacs lab11-1.sh
[asmaslova@fedora listing]$ ./lab11-1.sh -ilab11-1.sh -pgrep
12:      then grep -e${pval} -n${ival}
14:      then grep -e${pval} -n ${ival} > ${oval}
18:then grep -e${pval} -i -n ${ival}
20:      then grep -e${pval} -i -n ${ival} > ${oval}
24:then grep -e${pval} ${ival}
26:      then grep -e${pval} ${ival} > ${oval}
30:      then grep -e${pval} -i ${ival}
32:      then grep -e${pval} -i ${ival} > ${oval}
      then grep -e${pval} -n${ival}
      then grep -e${pval} -n ${ival} > ${oval}
then grep -e${pval} -i -n ${ival}
      then grep -e${pval} -i -n ${ival} > ${oval}
then grep -e${pval} ${ival}
      then grep -e${pval} ${ival} > ${oval}
      then grep -e${pval} -i ${ival}
      then grep -e${pval} -i ${ival} > ${oval}
      then grep -e${pval} -n${ival}
      then grep -e${pval} -n ${ival} > ${oval}
then grep -e${pval} -i -n ${ival}
      then grep -e${pval} -i -n ${ival} > ${oval}
then grep -e${pval} ${ival}
      then grep -e${pval} ${ival} > ${oval}
      then grep -e${pval} -i ${ival}
      then grep -e${pval} -i ${ival} > ${oval}
[asmaslova@fedora listing]$

```

рис.2 Пример работы программы

2. На языке С я написала программу, определяющую, больше, меньше или равно нулю введенное пользователем число (рис.3-5).

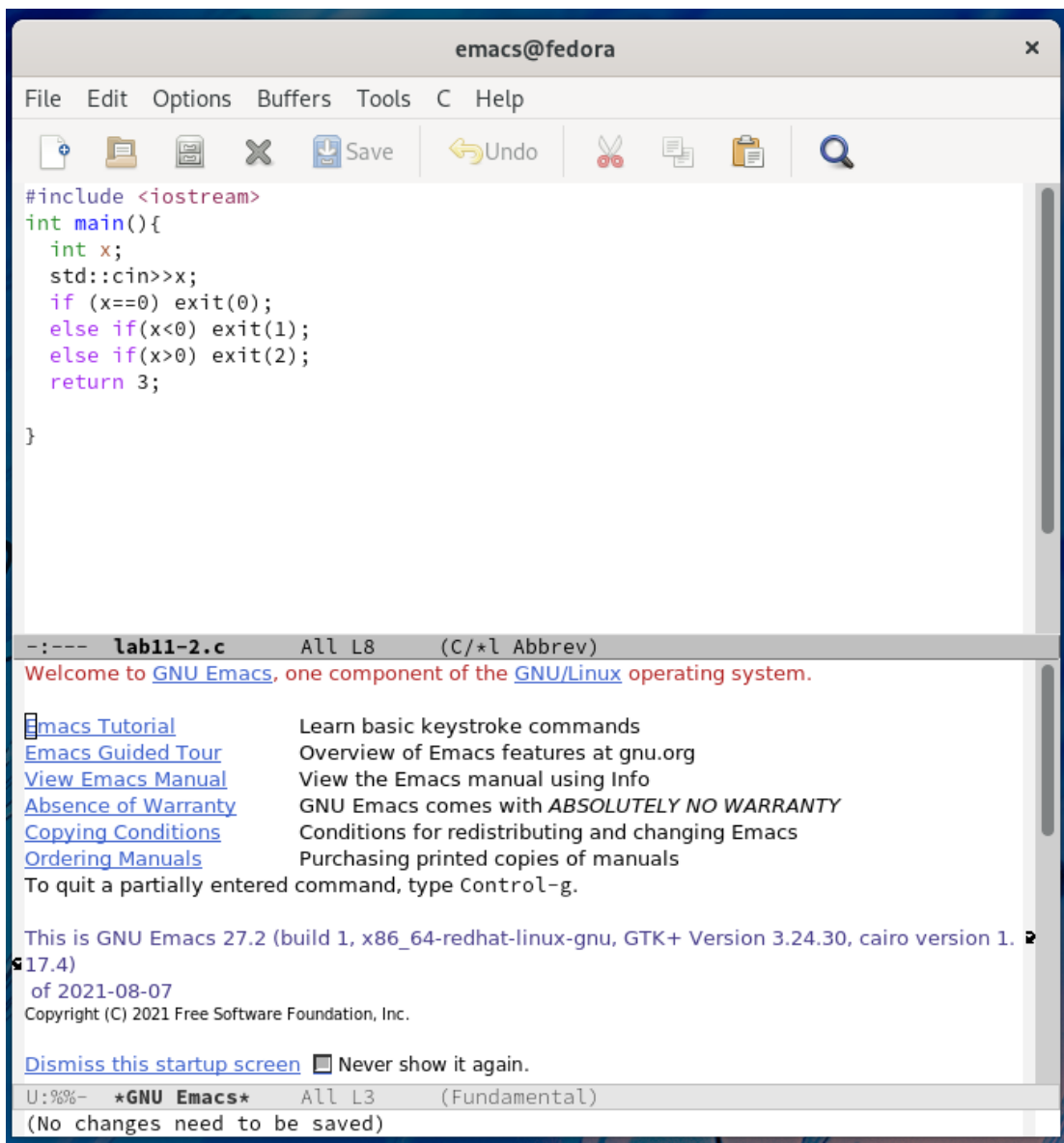


рис.3 Код программы 2.1

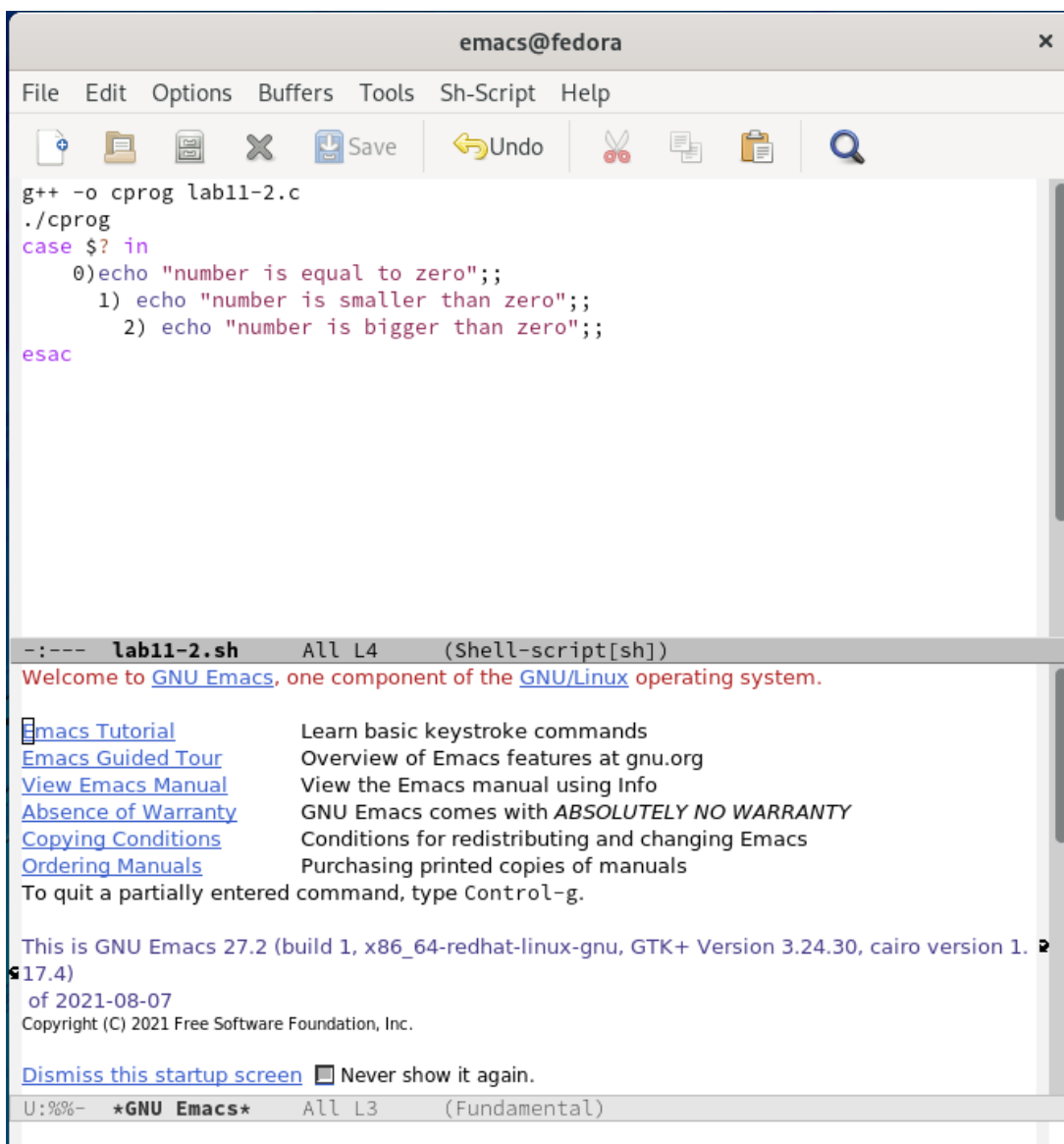
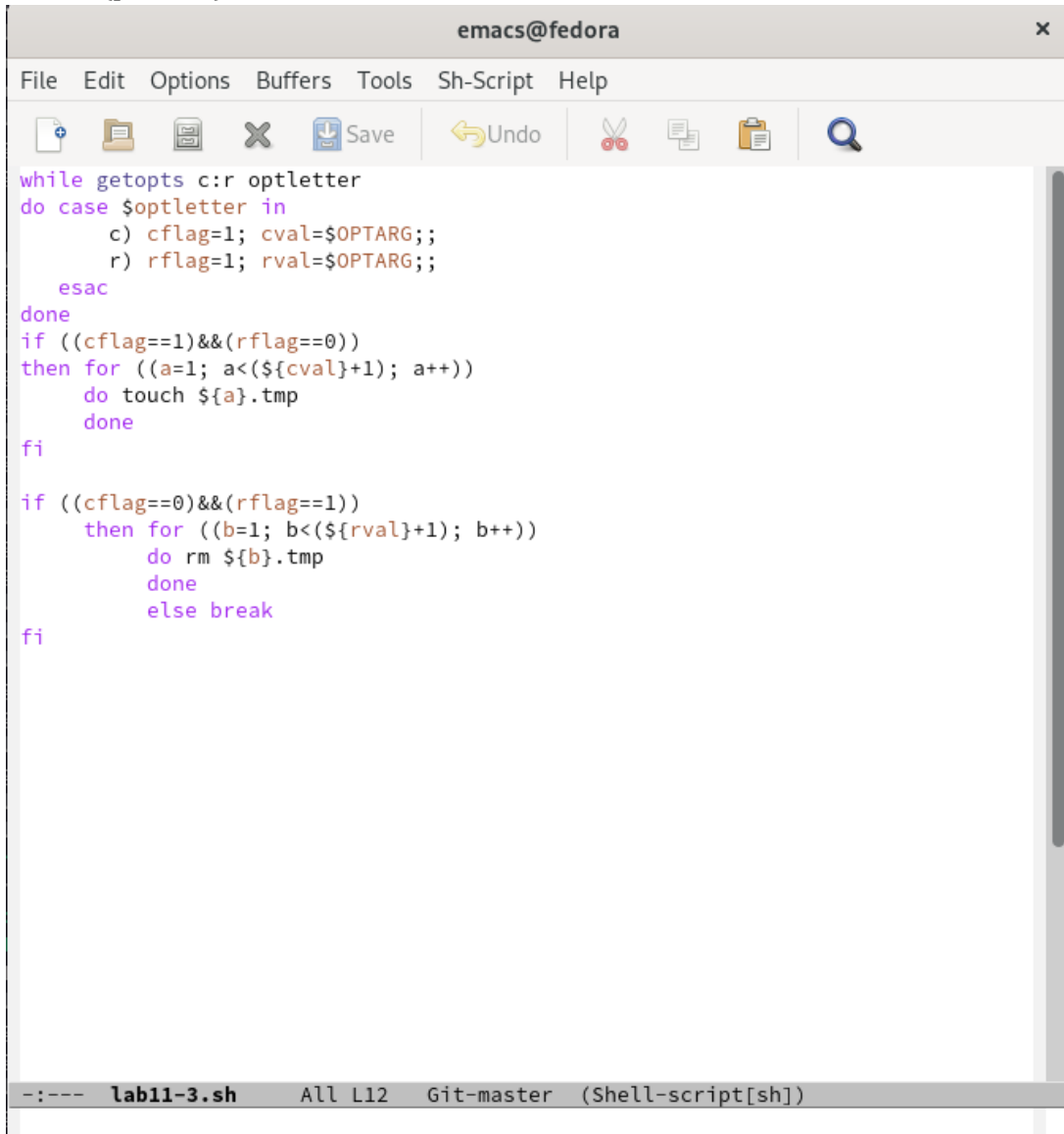


рис.4 Код программы 2.2

```
[asmaslova@fedora listing]$ emacs lab11-2.c &
[2] 7920
[asmaslova@fedora listing]$ ./lab11-2.sh
5
number is bigger than zero
[asmaslova@fedora listing]$ 0
bash: 0: command not found...
[asmaslova@fedora listing]$ ./lab11-2.sh
0
number is equal to zero
[asmaslova@fedora listing]$ ./lab11-2.sh
-1513
number is smaller than zero
[asmaslova@fedora listing]$
```

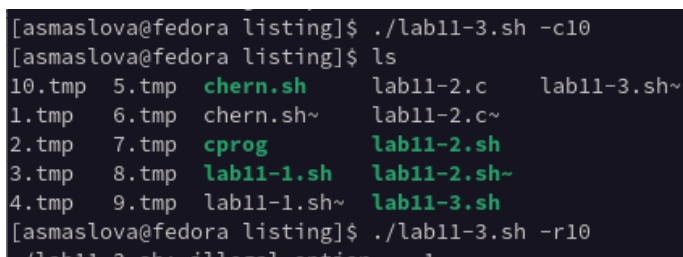
рис.5 Пример работы программы

3. Далее я написала командный файл, создающий указанное пользователем количество файлов, пронумерованных последовательно от 1 до указанного числа (рис.6-7).



```
while getopts c:r optletter
do case $optletter in
    c) cflag=1; cval=$OPTARG;;
    r) rflag=1; rval=$OPTARG;;
    esac
done
if ((cflag==1)&&(rflag==0))
then for ((a=1; a<({cval}+1); a++))
do touch ${a}.tmp
done
fi
if ((cflag==0)&&(rflag==1))
then for ((b=1; b<({rval}+1); b++))
do rm ${b}.tmp
done
else break
fi
```

рис.6 Код программы 3



```
[asmaslova@fedora listing]$ ./lab11-3.sh -c10
[asmaslova@fedora listing]$ ls
10.tmp  5.tmp  chern.sh  lab11-2.c  lab11-3.sh~
1.tmp   6.tmp  chern.sh~ lab11-2.c~
2.tmp   7.tmp  cprog     lab11-2.sh
3.tmp   8.tmp  lab11-1.sh lab11-2.sh~
4.tmp   9.tmp  lab11-1.sh~ lab11-3.sh
[asmaslova@fedora listing]$ ./lab11-3.sh -r10
(lab11-3.sh: illegal option -r)
```

рис.7 Пример работы программы

## Выводы

В результате выполнения лабораторной работе я получила навыки написания более сложных командных файлов с использованием логических управляющих конструкций и циклов.

## Список литературы