

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ «РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ»

Факультет физико-математических и естественных наук Кафедра информационных технологий

ОТЧЕТ

по лабораторной работе 11

ТЕМА «Программирование в командном процессоре ОС UNIX. Ветвления и циклы»

по дисциплине» по дисциплине «Операционные системы»

Выполнил/ла:

Студент/ка группы НПИбд-02-21

Студенческий билет No 1032205421

Студент/ка: Стелина Петрити

Цель работы

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

Последовательность выполнения работы

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами:

– `-iinputfile` — прочитать данные из указанного файла;

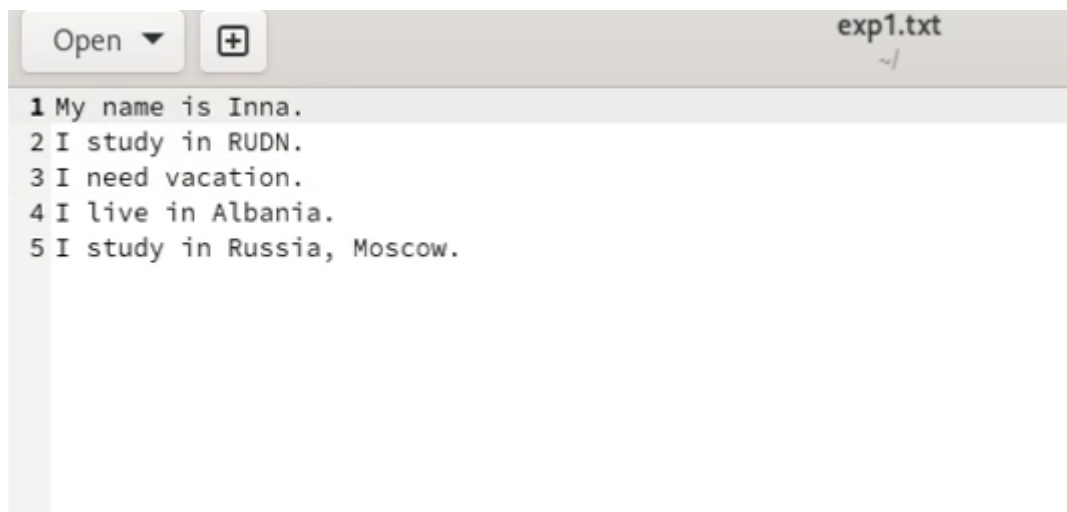
– `-ooutputfile` — вывести данные в указанный файл;

– `-rшаблон` — указать шаблон для поиска;

– `-C` — различать большие и малые буквы;

– `-n` — выдавать номера строк.

а затем ищет в указанном файле нужные строки, определяемые ключом `-r`



```
Open ▾ + exp1.txt
~/
1 My name is Inna.
2 I study in RUDN.
3 I need vacation.
4 I live in Albania.
5 I study in Russia, Moscow.
```

1.1. данные exp1.txt

```
[inna@fedora ~]$ cat exp2.txt
[inna@fedora ~]$ cat exp1.txt
My name is Inna.
I study in RUDN.
I need vacation.
I live in Albania.
I study in Russia, Moscow.
[inna@fedora ~]$ cat exp3.sh
#!/bin/bash

while getopts "i:o:p:C:n" opt
do
    case $opt in
        i) input=$OPTARG;;
        o) output=$OPTARG;;
        p) shablon=$OPTARG;;
        C) diff="";;
        n) num="";;

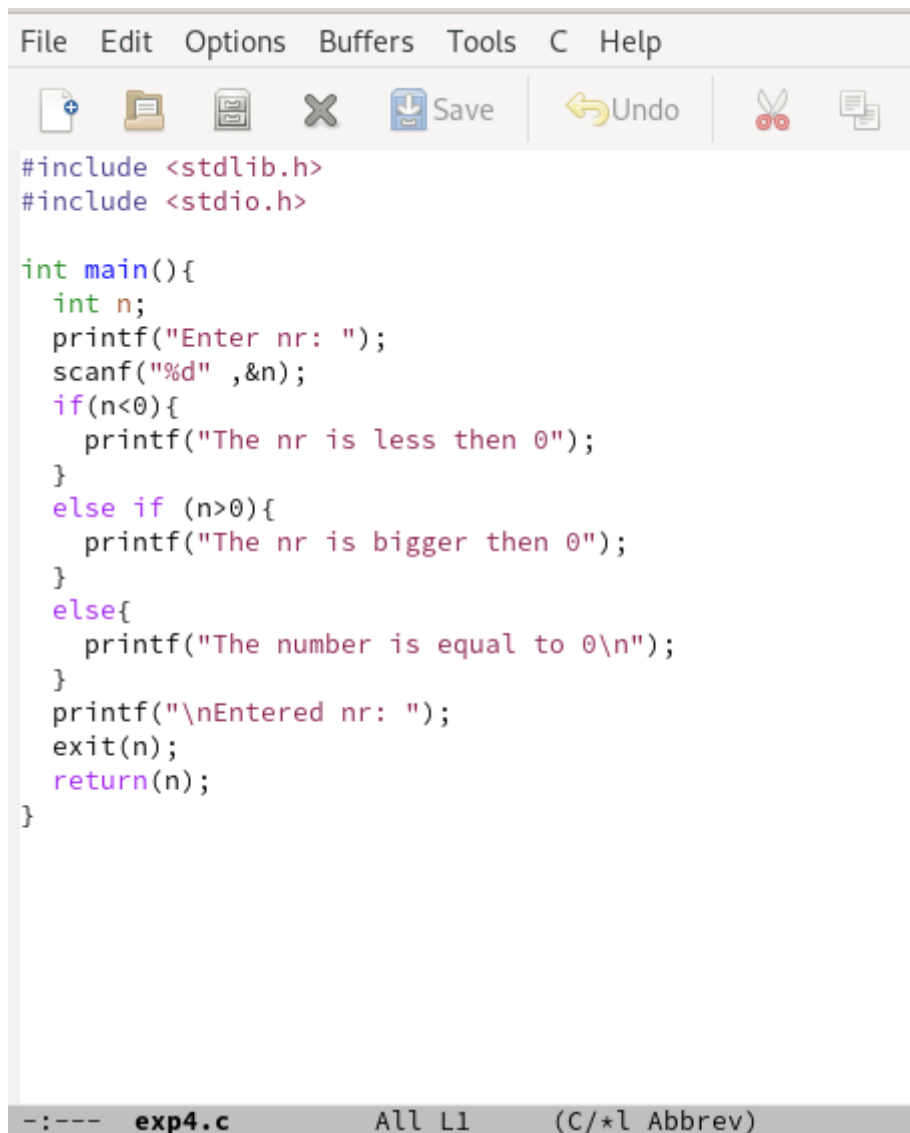
        esac
    done
    grep -n "$shablon" "$input" > "$output"
[inna@fedora ~]$
```

1.2. анализирует командную строку с ключами

```
[inna@fedora ~]$ chmod +x exp3.sh
[inna@fedora ~]$ ./exp3.sh -i exp1.txt -o exp2.txt -p "study" -C -n
[inna@fedora ~]$ cat exp2.txt
2:I study in RUDN.
5:I study in Russia, Moscow.
[inna@fedora ~]$ ./exp3.sh -i exp1.txt -o exp2.txt -p "Albania" -C -n
[inna@fedora ~]$ cat exp2.txt
4:I live in Albania.
[inna@fedora ~]$ ./exp3.sh -i exp1.txt -o exp2.txt -p "RUDN" -C -n
[inna@fedora ~]$ cat exp2.txt
2:I study in RUDN.
[inna@fedora ~]$ ./exp3.sh -i exp1.txt -o exp2.txt -p "" -C -n
[inna@fedora ~]$ cat exp2.txt
1:My name is Inna.
2:I study in RUDN.
3:I need vacation.
4:I live in Albania.
5:I study in Russia, Moscow.
```

1.3. проверка

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.



The image shows a screenshot of a C code editor window. The window has a menu bar with 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'C', and 'Help'. Below the menu bar is a toolbar with icons for opening a file, saving, closing, and a 'Save' button, followed by an 'Undo' button and icons for cutting and pasting. The main area of the window contains C code for a program that checks if a number is less than, greater than, or equal to zero. The code is as follows:

```
#include <stdlib.h>
#include <stdio.h>

int main(){
    int n;
    printf("Enter nr: ");
    scanf("%d" ,&n);
    if(n<0){
        printf("The nr is less then 0");
    }
    else if (n>0){
        printf("The nr is bigger then 0");
    }
    else{
        printf("The number is equal to 0\n");
    }
    printf("\nEntered nr: ");
    exit(n);
    return(n);
}
```

At the bottom of the window, there is a status bar that reads: `--:--- exp4.c All L1 (C/*l Abbrev)`

2.1.code exp4.c



```
File Edit Options Buffers Tools
+ [new file] [open file] [save] [close] Save
#!/bin/bash
gcc exp4.c -o exp4
./exp4
echo $?
```

--- exp5.sh All L1

2.2.code exp5.sh

```
[inna@fedora ~]$ emacs
[inna@fedora ~]$ chmod +x exp5.sh
[inna@fedora ~]$ ./exp5.sh
Enter nr: 3
The nr is bigger then 0
Entered nr: 3
[inna@fedora ~]$ ./exp5.sh
Enter nr: -1
The nr is less then 0
```

2.3. проверка

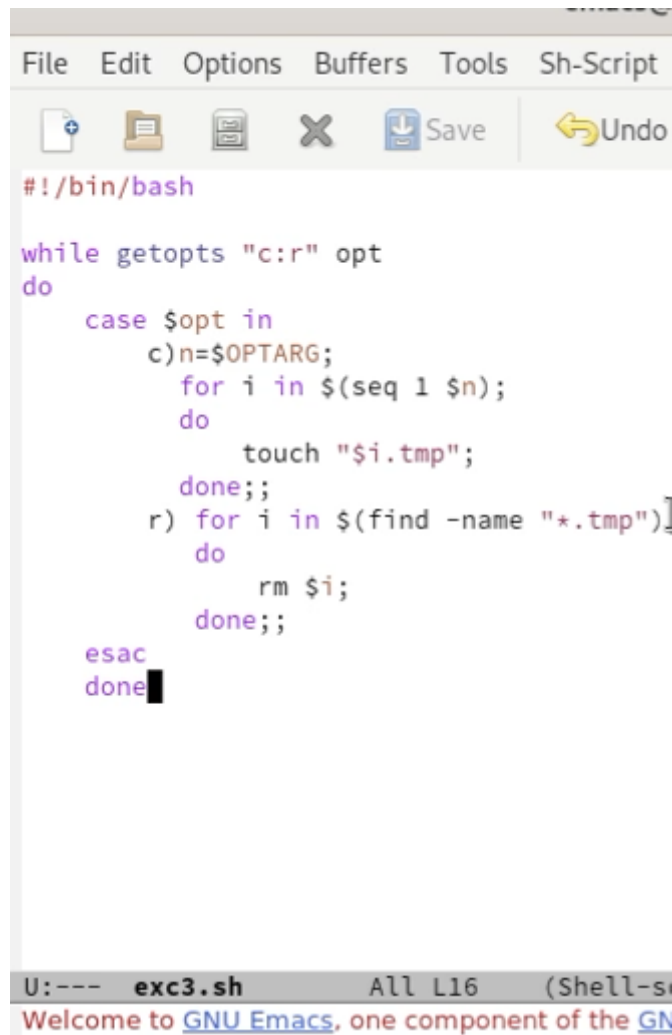
```
[inna@fedora ~]$ ./exp5.sh
Enter nr: 0
The number is equal to 0
Entered nr: 0
```

2.4. проверка

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).



```
#!/bin/bash

while getopts "c:r" opt
do
    case $opt in
        c)n=$OPTARG;
        for i in $(seq 1 $n);
        do
            touch "$i.tmp";
        done;;
        r) for i in $(find -name "*.tmp");
        do
            rm $i;
        done;;
    esac
done
```

3.1. code exc3.sh



```
inna@fedora ~]$ ls
123.sh  conf.txt      exp1.txt  feathers    '##lab07.sh#~'  lab4.pdf  pandoc-crossref-Linux.tar.xz  ski.plases
123.tar Desktop      exp2.txt  file.txt    '##lab07.sh#'   letters   parentdir                    Templates
456.sh  dir1          exp3.sh   'gets Linux' lab07.sh        may       Pictures                     text.txt
abc1.sh 'dmesg | grep -i "What are we looking"' exp4      hello.cpp    lab10.sh       memos     play                         Videos
archive Downloads    exp4.c    home        lab10.sh-      misk      point3                      work
backup  exc3.sh       exp5.sh   '##lab07.sh###' lab10.tar      monthly  Public                      world
inna@fedora ~]$ chmod +x exc3.sh
inna@fedora ~]$ ./exc3.sh -c 10
inna@fedora ~]$ ls
10.tmp  5.tmp  backup  exp1.txt  file.txt  lab07.sh  memos  point3  world
123.sh  6.tmp  conf.txt exp2.txt  'gets Linux' lab10.sh  misk  Public  World
123.tar 7.tmp  Desktop exp3.sh   hello.cpp  lab10.sh- monthly reports ski.plases
1.tmp   8.tmp  dir1     exp4      home       lab10.tar Music   reports Templates
2.tmp   9.tmp  'dmesg | grep -i "What are we looking"' exp4.c    '##lab07.sh###' lab4.md  pandoc-crossref-Linux.tar.xz text.txt
3.tmp   abc1.sh Documents exp5.sh   '##lab07.sh#~' lab4.pdf parentdir Pictures Videos
456.sh  abc1.tar Downloads exc3.sh   '##lab07.sh#~' letters  play     work
4.tmp   archive exc3.sh  feathers  '##lab07.sh#'   may      work
inna@fedora ~]$
```

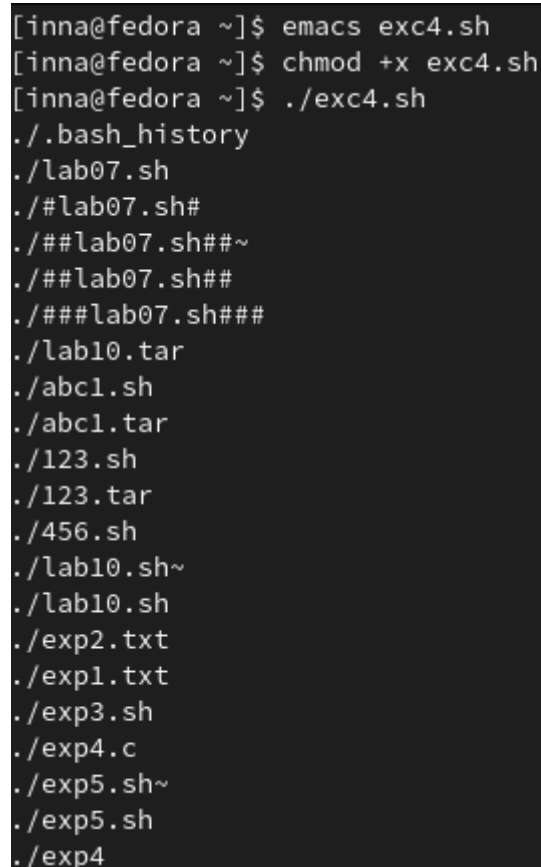
3.2. проверка

4. Написать командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find).



```
#!/bin/bash
while getopts "p" opt
do
    case $opt in
        p)dir=$OPTARG;;
    esac
done
find $dir -maxdepth 1 -ctime -7 -type f -print0 | xargs -0 tar rfv exp1.tar
```

4.1.code exc4.sh



```
[inna@fedora ~]$ emacs exc4.sh
[inna@fedora ~]$ chmod +x exc4.sh
[inna@fedora ~]$ ./exc4.sh
./.bash_history
./lab07.sh
./#lab07.sh#
./##lab07.sh##~
./##lab07.sh##
./###lab07.sh###
./lab10.tar
./abc1.sh
./abc1.tar
./123.sh
./123.tar
./456.sh
./lab10.sh~
./lab10.sh
./exp2.txt
./exp1.txt
./exp3.sh
./exp4.c
./exp5.sh~
./exp5.sh
./exp4
```

4.2.проверка

Выводы

В этой лаборатории я научилась писать сложные командные файлы, используя логические управляющие конструкции и циклы. Я изучила основы программирования в оболочке.

Контрольные вопросы

1. Каково предназначение команды `getopts`?

`getopts` - это очень удобная утилита `bash script`, которая помогает вам удобно и изящно обрабатывать передачу флагов и аргументов чистым, стандартизированным способом и чтобы разработчик вручную обрабатывал флаги, передаваемые скрипту, это обеспечивает удобный способ обработки этого при повторении цикла `while`.

2. Какое отношение метасимволы имеют к генерации имён файлов?

Оболочка может генерировать имена файлов, соответствующие именам существующих файлов.

3. Какие операторы управления действиями вы знаете?

`bash`, `for`, `case`, `if`, `while` и `until`.

4. Какие операторы используются для прерывания цикла?

`break` - для завершения цикла `while` в ситуациях, когда условие перестаёт быть правильным.

`continue` - используется когда больше нет необходимости выполнять блок операторов.

5. Для чего нужны команды `false` и `true`?

`true`- возвращает нулевое значение выхода.

`false`- возвращает ненулевое значение выхода.

6. Что означает строка `if test -f man$/i.$s`, встречающаяся в командном файле?

является ли аргумент `man$/i.$s` обычным файлом и возвращает логическое значение

7. Объясните различия между конструкциями `while` и `until`.

Цикл `while` выполняет тело цикла пока условие истинно.

Цикл `until` выполняет тело цикла пока условие ложно.