

## **Отчёт по лабораторной работе №12**

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

**Дисциплина:** Операционные системы

**Студент:** Оразгелдиева Огулнур

**Группа:** НПИбд-02-20

**Студ. номер:** 1032205431

2021, Москва

---

### **Лабораторная работа №12**

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

##### **Цель:**

- изучить основы программирования в оболочке ОС UNIX.
  - научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов
- 

##### **Задачи:**

1. Ознакомиться с теоретическим материалом
  2. Написать командный файл, используя команды `getopts` `grep`
  3. Написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю
  4. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N. Число файлов, которые необходимо создать, передаётся в аргументы командной строки.
  5. Написать командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории
- 

##### **Теоретические сведения [1]**

Весьма необходимой при программировании является команда `getopts`, которая осуществляет синтаксический анализ командной строки, выделяя флаги, и

используется для объявления переменных. Синтаксис команды следующий: *getopts option-string variable [arg ...]*

Функция *getopts* включает две специальные переменные среды — *OPTARG* и *OPTIND*. Если ожидается дополнительное значение, то *OPTARG* устанавливается в значение этого аргумента (будет равно *file\_in.txt* для опции *i* и *file\_out.doc* для опции *o*). *OPTIND* является числовым индексом на упомянутый аргумент. Функция *getopts* также понимает переменные типа массив, следовательно, можно использовать её в функции не только для синтаксического анализа аргументов функций, но и для анализа введённых пользователем данных.

В обобщённой форме оператор цикла *for* выглядит следующим образом:

*for имя [in список-значений]*

*do список-команд*

*done*

При каждом следующем выполнении оператора цикла *for* переменная *имя* принимает следующее значение из списка значений, задаваемых списком значений. Вообще говоря, список-значений является необязательным. При его отсутствии оператор цикла *for* выполняется для всех позиционных параметров или, иначе говоря, аргументов. Таким образом, оператор *for i* эквивалентен оператору *for i in \$\**. Выполнение оператора цикла *for* завершается, когда список значений будет исчерпан.

В обобщённой форме условный оператор *if* выглядит следующим образом:

*if список-команд*

*then список-команд*

*{elif список-команд then список-команд}*

*[else список-команд]*

*fi*

Выполнение условного оператора *if* сводится к тому, что сначала выполняется последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово *if*. Затем, если последняя выполненная команда из этой последовательности команд возвращает нулевой код завершения (истина), то будет выполнена последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово *then*. Фраза *elif* проверяется в том случае, когда предыдущая проверка была ложной. Строка, содержащая служебное слово *else*, является необязательной.

---

## Ход работы

1. Ознакомилась с теоретическим материалом из лабораторной работы 11.

Создала и открыла командный файл с помощью текстового редактора *vi*. (см. рис. 1)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-1.sh
```

*Рисунок 1. Создание командного файла*

Написала программу (см. рис. 2), используя `getopts` и цикл `if`, который анализирует командную строку с ключами:

*inputfile* — прочитать данные из указанного файла;

*outputfile* — вывести данные в указанный файл;

*ршаблон* — 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`.

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
#!/bin/bash  
while getopts o:i:p:Cn optletter  
do case $optletter in  
o) oflag=1; ooutputfile=$OPTARG;;  
i) iflag=1; iinputfile=$OPTARG;;  
p) pflag=1; pshablon=$OPTARG;;  
C) Cflag=1;;  
n) nflag=1;;  
*) echo Illegal option $optletter  
esac  
done  
if ((Cflag==1) && (nflag==1))  
then  
grep -i -n ${pshablon} ${iinputfile} > ${ooutputfile}  
elif ((Cflag==0) && (nflag==1))  
then  
grep -n ${pshablon} ${iinputfile} > ${ooutputfile}  
elif ((Cflag==1) && (nflag==0))  
then  
grep -i ${pshablon} ${iinputfile} > ${ooutputfile}  
elif ((Cflag==0) && (nflag==0))  
then  
grep ${pshablon} ${iinputfile} > ${ooutputfile}  
fi
```

Рисунок 2. Программа к заданию 1

**Пояснения:** сначала используем `getopts` и задаем нужные нам опции (o, i, p, C, n); затем используя цикл `if`, ищем внутри файла для чтения строки с шаблоном и записываем эту строку в файл для записи, используя команду `grep` (для этого изучила особенности этой команды в доп. источнике [2]).

Создаем файл для чтения (`iinputfile.txt`) и записываем в него некоторый текст с помощью текстового редактора `vi`. (см. рис. 3-4)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi iinputfile.txt
```

Рисунок 3. Создание файла чтения

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
operating systems  
Operating Systems  
hello world  
OPERATING SYSTEMS  
█
```

Рисунок 4. Текст для чтения

Создаём файл для записи с помощью команды touch. (см. рис. 5)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi iinputfile.txt  
[oorazgeldiyeva@oorazgeldiyeva ~]$ touch ooutputfile.txt
```

Рисунок 5. Создание файла записи

Предоставляем права на выполнение командного файла с помощью команды `chmod +x`. (см. рис. 6)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi iinputfile.txt  
[oorazgeldiyeva@oorazgeldiyeva ~]$ touch ooutputfile.txt  
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab12-1.sh
```

Рисунок 6. Права на выполнение

Выполняем командный файл, в качестве шаблона (слова для поиска) берем слово "systems". После выполнения смотрим содержимое файла записи с помощью `cat`. (см. рис. 7)

```
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab12-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab12-1.sh -i iinputfile.txt -o ooutputfile.  
txt -p systems -C -n  
[oorazgeldiyeva@oorazgeldiyeva ~]$ cat ooutputfile.txt  
1:operating systems  
2:Operating Systems  
4:OPERATING SYSTEMS
```

Рисунок 7. Выполнение командного файла

Как видно по рис. 7, в файл записались строки с указанным словом, при этом учитывались заглавные и строчные буквы (опция -C) и номера строк в исходном файле (опция -n).

2. Создаём и открываем файл расширения ".cpp", чтобы записать в него программу на языке C, 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. (см. рис. 8)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
#include <stdio.h>  
#include <stdlib.h>  
  
int main(){int n;  
printf("Enter number\n");  
scanf("%d",&n);  
if (n>0) printf("%d>0\n", n);  
if (n<0) printf("%d<0\n", n);  
if (n==0) printf("%d=0\n", n);  
exit(n);  
return 0;  
}  
~
```

Рисунок 8. Программа на C

**Пояснения:** в первых строчках пишем названия библиотек, которые нужны для выполнения программы (`#include...`); затем переходим к главной функции: запрос на ввод числа (`printf`), чтение этого числа при помощи `scanf`. Используем циклы `if` для проверки вводимого числа: больше/меньше/равно 0.

Теперь создаем командный файл с помощью редактора `vi`, который должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено. (см. рис. 9-10)

```
2:Operating Systems  
4:OPERATING SYSTEMS  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12.cpp  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-2.sh
```

Рисунок 9. Создание командного файла

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
#!/bin/bash  
g++ -o lab12 lab12.cpp  
./lab12  
echo $?  
~
```

Рисунок 10. Командный файл

Предоставляем командному файлу (lab12-2.sh) и файлу с программой на C (lab12.cpp) права на выполнение. (см. рис. 11)

```
4. CREATING SYSTEMS
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12.cpp
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-2.sh
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab12.cpp
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab12-2.sh
```

*Рисунок 11. Права на выполнение*

Выполняем командный файл. Вводим для примера число 9. (см. рис. 12)

```
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab12-2.sh
Enter number
9
9>0
9
```

*Рисунок 12. Выполнение программы*

Теперь для примера введем 0. (см. рис. 13)

```
9
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab12-2.sh
Enter number
0
0=0
0
```

*Рисунок 13. Выполнение программы*

Программа работает правильно: сначала выводится неравенство/равенство с 0, потом само введенное число.

3. Создаём командный файл для задания с помощью текстового редактора vi и напишем в нём программу, 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). (см. рис. 14)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
#!/bin/bash  
n=""  
a=""  
echo "Enter number of files"  
read n  
for ((i=1; i<=n; i++))  
do  
touch $i.tmp  
done  
echo "Files were created"  
echo "Files of catalog:"  
ls  
echo "Do you want to delete created files? Enter y/n"  
read a  
if (a=="y")  
then  
for ((i=1; i<=n; i++))  
do  
rm $i.tmp  
done  
echo "Files of catalog"  
ls  
fi
```

Рисунок 14. Программа к заданию 3

**Пояснения:** создаем переменные n (для вводимого числа - число создаваемых/удаляемых файлов) и a (для ответа). Чтение числа при помощи read; затем при помощи цикла for создаём файлы в текущем каталоге, используя touch, i.tmp (i - номер создаваемого файла). Далее выводим запрос на удаление, если ответ "y", то удаляем созданные файлы. (использовали if).

Выполняем командный файл. Создаем и потом удаляем 5 файлов. (см. рис. 15)



```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-3.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab12-3.sh  
Enter number of files  
5  
Files were created  
Files of catalog:  
004-lab_proc.pdf  file.txt      lab11-3.sh  lab12.tar      Загрузки  
1.tmp             iinputfile.txt lab11-4.sh  #lab.txt#      Изображения  
2.tmp             lab08         lab12      lab.txt        Музыка  
3.tmp             #lab10.sh#    lab12-1.sh ooutputfile.txt Общедоступные  
4.tmp             lab10.sh      lab12-2.sh text.txt       Рабочий стол  
5.tmp             lab10.sh~     lab12-3.sh work          Шаблоны  
backup           lab11-1.sh    lab12-4.sh Видео  
c.cpp            lab11-2.sh    lab12.cpp  Документы  
Do you want to delete created files? Enter y/n  
y  
Files of catalog  
004-lab_proc.pdf  lab10.sh      lab12-1.sh  lab.txt        Изображения  
backup           lab10.sh~     lab12-2.sh  ooutputfile.txt Музыка  
c.cpp            lab11-1.sh    lab12-3.sh  text.txt       Общедоступные  
file.txt         lab11-2.sh    lab12-4.sh  work          Рабочий стол  
iinputfile.txt   lab11-3.sh    lab12.cpp   Видео          Шаблоны  
lab08            lab11-4.sh    lab12.tar   Документы  
#lab10.sh#       lab12         #lab.txt#   Загрузки
```

Рисунок 15. Выполнение программы

Как видим, в домашнем каталоге мы создали 5 файлов, затем удалили их.

4. Создаём и открываем командный файл при помощи текстового редактора vi и составляем в нем программу (см. рис. 16), которая с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицируем его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (используем команду find).

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
#!/bin/bash  
catalog=""  
archive=""  
echo "Enter name of directory"  
read catalog  
cd $catalog  
echo "Name of archive"  
read archive  
find . -mtime -7 -type f -print0 | xargs -0 tar -cvzf ${archive}.tar  
echo "Archive was created"  
echo "Files of catalog:"  
ls
```

Рисунок 16. Программа к заданию 4

**Пояснения:** создаем переменные *catalog* и *archive*, которые будут содержать название задаваемого каталога и создаваемого архива соответственно. Читаем название

каталога и переходим в него, используя соответственно команды *read* и *cd*. С помощью команды *find*, конвейера и архиватора *tar* создаем архив и добавляем туда файлы данного каталога, которые были изменены неделю тому назад. (для этого изучила особенности команды *find* в доп. источнике [3]).

Предоставляем права на выполнение командного файла *lab12-4.sh* и выполняем программу. (см. рис. 17)

В домашнем каталоге и его подкаталогах ищем файлы и архивируем их с помощью архиватора *tar* в архив *lab12.tar*. (см. рис. 17-18)

```
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab12-4.sh
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab12-4.sh
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab12-4.sh
Enter name of directory

Name of archive
lab12
find: './.cache/gnome-control-center/backgrounds': Отказано в доступе
find: './.config/gnome-control-center': Отказано в доступе
././mozilla/extensions/{ec8030f7-c20a-464f-9b0e-13a3a9e97384}/.fedora-langpack-ins
tall
././mozilla/firefox/gqv6pxam.default-default/.parentlock
././mozilla/firefox/gqv6pxam.default-default/permissions.sqlite
././mozilla/firefox/gqv6pxam.default-default/cookies.sqlite
././mozilla/firefox/gqv6pxam.default-default/prefa...
```

Рисунок 17. Выполнение программы 4

```
Archive was created
Files of catalog:
004-lab_proc.pdf  lab10.sh  lab12-1.sh  lab.txt  Изображения
backup           lab10.sh~ lab12-2.sh  ooutputfile.txt  Музыка
c.cpp            lab11-1.sh lab12-3.sh  text.txt  Общедоступные
file.txt         lab11-2.sh lab12-4.sh  work      Рабочий стол
iinputfile.txt   lab11-3.sh lab12.cpp   Видео     Шаблоны
lab08            lab11-4.sh lab12.tar   Документы
#lab10.sh#       lab12     #lab.txt#   Загрузки
```

Рисунок 18. Выполнение программы 4

Как видим, в домашнем каталоге появился архив *lab12-4.tar*.

---

**Вывод:** на лабораторной работе изучила основы программирования в оболочке ОС UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов

---

## Библиография

[1] - РУДН, Операционные системы, Программирование в командном процессоре ОС UNIX. Ветвления и циклы

[2] - Руководство по команде *grep* ОС Unix/Linux

[3] - Команда *find* в ОС Unix/Linux