

Отчёт по лабораторной работе №13

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

Сергей Витальевич Павлюченков

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	13
5	Контрольные вопросы	14

Список иллюстраций

3.1	Создание файла	7
3.2	Код программы	8
3.3	Работа программы	8
3.4	Код программы на С	9
3.5	Код командного файла	9
3.6	Работа программы	10
3.7	Код программы	10
3.8	Запуск программы	11
3.9	Код программы	11
3.10	Запуск программы	12

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 Задание

1. Используя команды `getopts` `grep`, написать 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. а затем ищет в указанном файле нужные строки, определяемые ключом `-r`.
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 ∞ (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).
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`).

3 Выполнение лабораторной работы

Создаю директорию для работы и файл для первого задания

```
[svpavliuchenkov@fedora os]$ mkdir lab13  
[svpavliuchenkov@fedora os]$ cd lab13  
[svpavliuchenkov@fedora lab13]$ touch 1_lab13  
[svpavliuchenkov@fedora lab13]$
```

Рис. 3.1: Создание файла

Используя команду `getopts` `grep`, написал 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. а затем ищет в указанном файле нужные строки, определяемые ключом `-r`. и Записывает все в файл заданный `-o`.

```
mc [svpavliuchenkov@fedora]:~/work/os/lab13
1_lab13 [B---] 42 L:[ 1+20 21/ 21] *(330 / 330b)
#!/bin/bash
while getopts i:op: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 test $cflag
then
coption=-i
fi
if test $nflag
then
noption=-n
fi
grep $coption $noption $pval $ival > $oval
```

Рис. 3.2: Код программы

Для вида сначала вывожу содержимое файла, после чего запускаю программу и утверждаюсь в правильности ее работы. Код нашел все строки в которых встретилось слово aphs.

```
[svpavliuchenkov@fedora lab13]$ ./1_lab13 -i input.txt -o output.txt -p aphs -C -n
[svpavliuchenkov@fedora lab13]$ cat input
aphs is great
school burned
aphs was in Armenia
look he is from aphs[svpavliuchenkov@fedora lab13]$ cat output.txt
1:aphs is great
3:aphs was in Armenia
4:look he is from aphs
[svpavliuchenkov@fedora lab13]$
```

Рис. 3.3: Работа программы

Приступаю к выполнению 2-го задания. 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)`, пе-

передавая информацию в о коде завершения в оболочку.

```
mc [svpavliuchenkov@fedora]:~/work/os/lab13
2_lab13.c      [-----] 1 L:[ 1+13 14/ 14] *(185 / 185b
#include <stdio.h>
#include <stdlib.h>

int main() {
    int n;
    scanf("%d", &n);
    if (n > 0) {
        exit(1);
    }
    else if (n < 0) {
        exit(2);
    }
    else {
        exit(0);
    }
    return 0;
}
```

Рис. 3.4: Код программы на С

Написал командный файл который должен вызывать эту программу и, проанализировав с помощью команды \$?, выдавать сообщение о том, какое число было введено.

```
noname      [B---] 4 L:[ 1+ 8
#!/bin/bash

gcc -o 2_lab13.o 2_lab13.c
./2_lab13.o
case $? in
1) echo 'Number is positive';;
0) echo "Number is equal to 0";;
2) echo "Number is negative";;
esac
```

Рис. 3.5: Код командного файла

Запускаю код пару раз, все сработало правильно. тк $6 > 0$ и $0 = 0$.

```
[svpavliuchenkov@fedora lab13]$ ./noname
6
Number is positive
[svpavliuchenkov@fedora lab13]$ ./noname
0
Number is equal to 0
[svpavliuchenkov@fedora lab13]$
```

Рис. 3.6: Работа программы

Приступаю к 3 заданию. Написал 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. Этот же командный файл умеет удалять все созданные им файлы (если они существуют).

```
1 mc [svpavliuchenkov@fedora]:~/work/
mc [svpavliuchenkov@fedora]:~/work/os/
3_lab13 [B---] 4 L:[ 1+10 11/ 11] *(10
#!/bin/bash

for ((i = 1; i <= $1; i++))
do
if test -f "$i.tmp"
then
rm "$i.tmp"
else
touch "$i.tmp"
fi
done
```

Рис. 3.7: Код программы

Демонстрирую правильную работу программы. Сначала код создал файлы от 1

до 3, после чего удалил 1-3 и создал 4-5.

```
[svpavliuchenkov@fedora lab13]$ ./3_lab13 3
[svpavliuchenkov@fedora lab13]$ ls
1_lab13 1.tmp 2_lab13.c 2_lab13.o 2.tmp 3_lab13 3.tmp input input.txt noname output output.txt
[svpavliuchenkov@fedora lab13]$ ./3_lab13 5
[svpavliuchenkov@fedora lab13]$ ls
1_lab13 2_lab13.c 2_lab13.o 3_lab13 4.tmp 5.tmp input input.txt noname output output.txt
[svpavliuchenkov@fedora lab13]$
```

Рис. 3.8: Запуск программы

Приступаю к последнему заданию. 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 (использовал команду find).

```
1 mc [svpavliuchenkov@fedora]:~/work/os/lab13
mc [svpavliuchenkov@fedora]:~/work/os/lab13
4_lab13 [B---] 63 L:[ 1+ 3 4/ 5] *(83 / 84b) 0010 0x00A
#!/bin/bash
dir=$1
find $dir -type f -mtime -7 | tar -cvf "new-archie.tar" -T -
```

Рис. 3.9: Код программы

Проверяю, что программа запаковала только файлы которые были изменены менее недели назад.

```

svpavliuchenkov@fedora lab13]$ ./4_lab13 .
/1_lab13
/input
/output
/output.txt
/input.txt
/2_lab13.c
/noname
/2_lab13.o
/3_lab13
/4.tmp
/5.tmp
/4_lab13
svpavliuchenkov@fedora lab13]$ ./4_lab13ls -l
ash: ./4_lab13ls: Нет такого файла или каталога
svpavliuchenkov@fedora lab13]$ ls -l
total 84
-rwxr-xr-x. 1 svpavliuchenkov svpavliuchenkov 330 сен  4 17:24 1_lab13
-rwxr-xr-x. 1 svpavliuchenkov svpavliuchenkov 185 сен  4 17:53 2_lab13.c
-rwxr-xr-x. 1 svpavliuchenkov svpavliuchenkov 16720 сен  4 18:07 2_lab13.o
-rwxr-xr-x. 1 svpavliuchenkov svpavliuchenkov 108 сен  4 18:18 3_lab13
-rwxr-xr-x. 1 svpavliuchenkov svpavliuchenkov 84 сен  4 19:54 4_lab13
-rw-r--r--. 1 svpavliuchenkov svpavliuchenkov 0 сен  4 18:18 4.tmp
-rw-r--r--. 1 svpavliuchenkov svpavliuchenkov 0 сен  4 18:18 5.tmp
-rw-r--r--. 1 svpavliuchenkov svpavliuchenkov 68 сен  4 15:52 input
-rw-r--r--. 1 svpavliuchenkov svpavliuchenkov 68 сен  4 16:12 input.txt
-rw-r--r--. 1 svpavliuchenkov svpavliuchenkov 30720 сен  4 19:55 new-archieve.tar
-rwxr-xr-x. 1 svpavliuchenkov svpavliuchenkov 162 сен  4 18:07 noname
-rw-r--r--. 1 svpavliuchenkov svpavliuchenkov 0 сен  4 15:51 output
-rw-r--r--. 1 svpavliuchenkov svpavliuchenkov 61 сен  4 17:24 output.txt
svpavliuchenkov@fedora lab13]$

```

Рис. 3.10: Запуск программы

4 Выводы

Я попрактиковался в создании разных алгоритмов на `bash`. Поработал в `getopts`, `grep` и `find`.

5 Контрольные вопросы

1. Каково предназначение команды `getopts`? `getopts` осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных.
2. Какое отношение метасимволы имеют к генерации имён файлов? имя файла содержится в `$0` .
3. Какие операторы управления действиями вы знаете? `for`, `case`, `if` и `while`
4. Какие операторы используются для прерывания цикла? `break` и `continue`
5. Для чего нужны команды `false` и `true`? Для циклов и условных конструкций.
6. Что означает строка `if test -f mans/i.$s`, встречаемая в командном файле? Данная строка проверяет является ли `man$1/1.s` файлом.
7. Объясните различия между конструкциями `while` и `until`. При замене в операторе цикла `while` служебного слова `while` на `until` условие, при выполнении которого осуществляется выход из цикла, меняется на противоположное