

**Лабораторная работа
№11. Программирование в командном
процессоре ОС UNIX. Ветвления и циклы.**

Операционные системы

Кочарян Никита Робертович

Содержание

1	Цель работы	5
2	Задания	6
3	Выполнение лабораторной работы	7
4	Контрольные вопросы	10
5	Ответы на контрольные вопросы	11
6	Вывод	13

Список иллюстраций

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 Задания

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; `-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 `tag` 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

1. Используя команды `getopts` `grep`, пишу 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.

```
Открыть lab11.sh
1#!/bin/bash
2iflag=0; oflag=0; pflag=0; nflag=0;
3while getopts i:op:C:n optletter
4do case $optletter in
5  i) iflag=1; ival=$OPTARG;;
6  o) oflag=1; oval=$OPTARG;;
7  p) pflag=1; pval=$OPTARG;;
8  C) iflag=1;;
9  n) iflag=1;;
10 *) echo illegal option $optletter
11 ;;
12 esac
13 done
14 if ((iflag==0))
15 then echo "Шаблон не найден"
16 else
17   if ((oflag==0))
18   then echo "Файл не найден"
19   else
20     if ((pflag==0))
21     then if ((iflag==0))
22          then if ((oflag==0))
23              then grep $pval $ival > $oval
24              else grep -n $pval $ival > $oval
25              fi
26          else if ((iflag==1))
27              then grep -i $pval $ival > $oval
28              else grep -i -n $pval $ival > $oval
29              fi
30          fi
31      else if ((iflag==1))
32          then if ((oflag==0))
33              then grep $pval $ival > $oval
34              else grep -n $pval $ival > $oval
35              fi
36          else if ((iflag==1))
37              then grep -i $pval $ival > $oval
38              else grep -i -n $pval $ival > $oval
39              fi
40          fi
41      fi
42 fi
```

```
Открыть lab11.txt
1Two roads diverged in a yellow wood,
2And sorry I could not travel both
3And be one traveler, long I stood
4And looked down one as far as I could
5To where it bent in the undergrowth.
6
7Then took the other, as just as fair,
8And having perhaps the better claim,
9Because it was grassy and wanted wear;
10Though as for that the passing there
11Had worn them really about the same.
```

```
grep: lab11.txt: Нет такого файла или каталога
nrkocharyan@dk3n37 ~$ bash lab11.sh -ilab11.txt -olab1-1.txt -pice
nrkocharyan@dk3n37 ~$ cat ~/lab11.txt
Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth.

Then took the other, as just as fair,
And having perhaps the better claim,
Because it was grassy and wanted wear;
Though as for that the passing there
Had worn them really about the same.
```

```
nrkocharyan@dk3n37 ~$ ./lab11.sh -i ~/lab11.txt -o ~/lab11-1.txt -p it -C -n
nrkocharyan@dk3n37 ~$ cat ~/lab11-1.txt
To where it bent in the undergrowth.
Because it was grassy and wanted wear;
nrkocharyan@dk3n37 ~$ ./lab11.sh -i ~/lab.txt -o ~/lab11-1.txt -p it -n
grep: /afs/.dk.sci.pfu.edu.ru/home/n/r/nrkocharyan/lab.txt: Нет такого файла или каталога
nrkocharyan@dk3n37 ~$ ./lab11.sh -i ~/lab11.txt -o ~/lab11-1.txt -p it -n
nrkocharyan@dk3n37 ~$ cat ~/lab11.txt
Two roads diverged in a yellow wood,
And sorry I could not travel both
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth.

Then took the other, as just as fair,
And having perhaps the better claim,
Because it was grassy and wanted wear;
Though as for that the passing there
Had worn them really about the same.
nrkocharyan@dk3n37 ~$ cat ~/lab11-1.txt
5:To where it bent in the undergrowth.
9:Because it was grassy and wanted wear;
nrkocharyan@dk3n37 ~$ ./lab11.sh -i ~/lab.txt -C -n
Шаблон не найден
nrkocharyan@dk3n37 ~$ ./lab11.sh -o ~/lab.txt -p it -n
Файл не найден
nrkocharyan@dk3n37 ~$
```

2. Пишу 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 screenshot shows a development environment with two editors and a terminal. The left editor, titled 'prog.c', contains the following C code:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 int main()
4 {
5     printf("Введите число: ");
6     int a;
7     scanf("%d", &a);
8     if (a<0) exit(0);
9     if (a>0) exit(1);
10    if (a==0) exit(2);
11    return 0;
12 }
```

The right editor, titled 'prog.sh', contains the following shell script:

```
1 #!/bin/bash
2
3 gcc prog.c -o prog
4 ./prog
5 code=$?
6 case $code in
7     0) echo "Число меньше 0";;
8     1) echo "Число больше 0";;
9     2) echo "Число равно 0";;
10 esac
```

The terminal window shows the execution of these scripts:

```
nrkocharyan@dk3n37 ~ $ touch prog
nrkocharyan@dk3n37 ~ $ touch prog.sh
nrkocharyan@dk3n37 ~ $ gedit prog
nrkocharyan@dk3n37 ~ $ touch prog.c
nrkocharyan@dk3n37 ~ $ gedit prog.c
nrkocharyan@dk3n37 ~ $ gedit prog.sh
nrkocharyan@dk3n37 ~ $ chmod +x prog.sh
bash: chmod: команда не найдена
nrkocharyan@dk3n37 ~ $ chmod +x prog.sh
nrkocharyan@dk3n37 ~ $ ./prog.sh
Введите число: 3
Число больше 0
nrkocharyan@dk3n37 ~ $ ./prog.sh
Введите число: -9
Число меньше 0
nrkocharyan@dk3n37 ~ $ ./prog.sh
Введите число: 0
Число равно 0
nrkocharyan@dk3n37 ~ $
```

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).


```

Открыть  prog2.sh  Сохранить
1#!/bin/bash
2opt=$1;
3form=$2;
4num=$3;
5function Files()
6{
7    for ((i=1; i<=num; i++)) dot
8    file=$(echo $form | tr '#' "$i")
9    if [ #opt == "-r" ]
10    then
11        rm -f $file
12    elif [ opt == "-c" ]
13    then
14        touch $file
15    fi
16    done
17}
18Files

```

```

nrkocharyan@dk3n37 ~$ chmod +x prog2.sh
nrkocharyan@dk3n37 ~$ ./prog2.sh -c a#.txt 3
./prog2.sh: строка 7: синтаксическая ошибка рядом с неожиданным маркером «dot»
./prog2.sh: строка 7: `for ((i=1; i<=num; i++)) dot`
nrkocharyan@dk3n37 ~$ ./prog2.sh -c a#.txt 3
./prog2.sh: строка 9: [: отсутствует символ «]»
./prog2.sh: строка 9: [: отсутствует символ «]»
./prog2.sh: строка 9: [: отсутствует символ «]»
nrkocharyan@dk3n37 ~$ ./prog2.sh -c a#.txt 3
./prog2.sh: строка 8: [: отсутствует символ «]»
./prog2.sh: строка 8: [: отсутствует символ «]»
./prog2.sh: строка 8: [: отсутствует символ «]»
nrkocharyan@dk3n37 ~$ ./prog2.sh -c a#.txt 3
./prog2.sh: строка 9: [: отсутствует символ «]»
./prog2.sh: строка 9: [: отсутствует символ «]»
./prog2.sh: строка 9: [: отсутствует символ «]»
nrkocharyan@dk3n37 ~$ ./prog2.sh -c a#.txt 3
./prog2.sh: строка 9: [: отсутствует символ «]»
./prog2.sh: строка 9: [: отсутствует символ «]»
nrkocharyan@dk3n37 ~$ ./prog2.sh -c a#.txt 3
nrkocharyan@dk3n37 ~$ ls
backup  lab07.sh-  lab11.txt  prog.c      script2.sh-  tmp  Документы  Общедоступные
bin     lab11-1.txt lab11-1.txt prog.sh     script3.sh-  TP  Загрузки  'Поляков с++'
file.sh- lab11.sh     prog       public      script4.sh-  work  Изображения  'Рабочий стол'
GNUstep lab1-1.txt   prog2.sh   public_html  script.sh-   Видео  Музыка  Шаблоны
nrkocharyan@dk3n37 ~$ ./prog2.sh -r a#.txt3
nrkocharyan@dk3n37 ~$ ./prog2.sh -r a#.txt 2
nrkocharyan@dk3n37 ~$ ls
backup  lab07.sh-  lab11.txt  prog.c      script2.sh-  tmp  Документы  Общедоступные
bin     lab11-1.txt lab11-1.txt prog.sh     script3.sh-  TP  Загрузки  'Поляков с++'
file.sh- lab11.sh     prog       public      script4.sh-  work  Изображения  'Рабочий стол'
GNUstep lab1-1.txt   prog2.sh   public_html  script.sh-   Видео  Музыка  Шаблоны
nrkocharyan@dk3n37 ~$ ./prog2.sh -r a#.txt 3
nrkocharyan@dk3n37 ~$ ls
backup  lab07.sh-  lab11.txt  prog.c      script2.sh-  tmp  Документы  Общедоступные
bin     lab11-1.txt lab11-1.txt prog.sh     script3.sh-  TP  Загрузки  'Поляков с++'
file.sh- lab11.sh     prog       public      script4.sh-  work  Изображения  'Рабочий стол'
GNUstep lab1-1.txt   prog2.sh   public_html  script.sh-   Видео  Музыка  Шаблоны
nrkocharyan@dk3n37 ~$

```

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)

```

nrkocharyan@dk3n37 ~$ touch prog 3.sh
nrkocharyan@dk3n37 ~$ gedit prog3.sh
nrkocharyan@dk3n37 ~$ chmod +x prog3.sh
nrkocharyan@dk3n37 ~$ ls -l
итого 61
-rw-r--r-- 1 nrkocharyan studsci 0 anp 20 14:38 3.sh
drwxr-xr-x 2 nrkocharyan studsci 2048 map 16 13:33 backup
drwxr-xr-x 2 nrkocharyan studsci 2048 map 16 13:07 bin
-rwxr-xr-x 1 nrkocharyan studsci 0 anp 13 13:41 file.sh-
drwxr-xr-x 3 nrkocharyan studsci 2048 map 2 11:41 GNUstep
-rw-r--r-- 1 nrkocharyan studsci 0 anp 6 14:28 lab07.sh-
-rw-r--r-- 1 nrkocharyan studsci 80 anp 20 14:14 lab11-1.txt
-rwxr-xr-x 1 nrkocharyan studsci 1210 anp 20 14:11 lab11.sh
-rw-r--r-- 1 nrkocharyan studsci 0 anp 20 14:11 lab1-1.txt
-rw-r--r-- 1 nrkocharyan studsci 369 anp 20 14:03 lab11.txt
-rw-r--r-- 1 nrkocharyan studsci 0 anp 20 14:11 lab11-1.txt
-rwxr-xr-x 1 nrkocharyan studsci 15624 anp 20 14:38 prog
-rwxr-xr-x 1 nrkocharyan studsci 232 anp 20 14:38 prog2.sh
-rwxr-xr-x 1 nrkocharyan studsci 210 anp 20 14:42 prog3.sh
-rw-r--r-- 1 nrkocharyan studsci 189 anp 20 14:20 prog.c
-rwxr-xr-x 1 nrkocharyan studsci 181 anp 20 14:25 prog.sh
drwxr-xr-x 3 nrkocharyan root 2048 сен 2 2022 public
lrwxr-xr-x 1 nrkocharyan root 19 мар 3 00:05 public_html -> public/public_html
-rwxr-xr-x 1 nrkocharyan studsci 44 anp 13 13:37 script2.sh-
-rwxr-xr-x 1 nrkocharyan studsci 234 anp 13 12:49 script3.sh-
-rwxr-xr-x 1 nrkocharyan studsci 144 anp 13 13:49 script4.sh-
drwxr-xr-x 2 nrkocharyan studsci 83 anp 13 13:30 script.sh-
drwxr-xr-x 2 nrkocharyan studsci 2048 map 2 11:40 tmp
drwxr-xr-x 6 nrkocharyan studsci 2048 map 30 12:11 work
drwxr-xr-x 2 nrkocharyan studsci 2048 сен 15 2022 Видео
drwxr-xr-x 2 nrkocharyan studsci 2048 anp 20 10:40 Документы
drwxr-xr-x 3 nrkocharyan studsci 2048 сен 19 2022 Загрузки
drwxr-xr-x 2 nrkocharyan studsci 2048 сен 15 2022 Изображения
drwxr-xr-x 2 nrkocharyan studsci 2048 сен 15 2022 Музыка
drwxr-xr-x 2 nrkocharyan studsci 2048 сен 15 2022 Общедоступные
drwxr-xr-x 3 nrkocharyan studsci 2048 мар 23 10:47 'Поляков с++'
drwxr-xr-x 2 nrkocharyan studsci 2048 мар 16 12:34 'Рабочий стол'
drwxr-xr-x 2 nrkocharyan studsci 2048 сен 15 2022 Шаблоны
nrkocharyan@dk3n37 ~$ sudo ~/work/prog3.sh

```

```

Открыть  prog3.sh  Сохранить
1#!/bin/bash
2
3files=$(find ./ -maxdepth 1 -mtime -7)
4listing=""
5for file in $files; do
6    file=$(echo "$file" | cut -c 3-)
7    listing="$listing $file"
8done
9dir=$(basename $(pwd))
10tar -cvf $dir.tar $listing

```

```

Мы полагаем, что ваш системный администратор изложил вам основы безопасности. Как правило, всё сводится к трём следующим правилам:

1) Уважайте частную жизнь других.
2) Думайте, прежде что-то вводите.
3) С большой властью приходит большая ответственность.

Пароль:
Попробуйте ещё раз.
Пароль:
Попробуйте ещё раз.

```

4 Контрольные вопросы

1. Каково предназначение команды `getopts`?
2. Какое отношение метасимволы имеют к генерации имён файлов?
3. Какие операторы управления действиями вы знаете?
4. Какие операторы используются для прерывания цикла?
5. Для чего нужны команды `false` и `true`?
6. Что означает строка `if test -f mans/i.$s`, встреченная в командном файле?
7. Объясните различия между конструкциями `while` и `until`

5 Ответы на контрольные вопросы

1. Команда `getopts` является встроенной командой командной оболочки `bash`, предназначенной для разбора параметров сценариев. Она обрабатывает исключительно однобуквенные параметры как с аргументами, так и без них и этого вполне достаточно для передачи сценариям любых входных данных.
2. При генерации имен используют метасимволы: `*` произвольная (возможно пустая) последовательность символов; `?` один произвольный символ; `[...]` любой из символов, указанных в скобках перечислением и/или с указанием диапазона; `cat f*` выдаст все файлы каталога, начинающиеся с `"f"`; `cat f` выдаст все файлы, содержащие `"f"`; `cat program.?` выдаст файлы данного каталога с однобуквенными расширениями, скажем `"program.c"` и `"program.o"`, но не выдаст `"program.com"`; `cat [a-d]*` выдаст файлы, которые начинаются с `"a"`, `"b"`, `"c"`, `"d"`. Аналогичный эффект дадут и команды `"cat [abcd]"` и `"cat [bdac]"`.
3. Операторы `&&` и `&` являются управляющими операторами. Если в командной строке стоит `command1 && command2`, то `command2` выполняется в том, и только в том случае, если статус выхода из команды `command1` равен нулю, что говорит об успешном ее завершении. Аналогично, если командная строка имеет вид `command1 command2`, то команда `command2` выполняется тогда, и только тогда, когда статус выхода из команды `command1` отличен от нуля.
4. Оператор `break` завершает выполнение ближайшего включающего цикла

или условного оператора, в котором он отображается.

5. Команда `true` всегда возвращает ноль в качестве выходного статуса для индикации успеха. Команда `false` всегда возвращает не-ноль в качестве выходного статуса для индикации неудачи. Во всех управляющих конструкциях в качестве логического значения используется код возврата из программы, указанной в качестве условия. Код возврата 0 – истина, любое другое значение – ложь. Программа `true` – всегда завершается с кодом 0, `false` – всегда завершается с кодом 1.
6. Введенная строка означает условие существования файла `mans/i.$s`
7. Цикл `While` выполняется до тех пор, пока указанное в нем условие истинно. Когда указанное условие становится ложным - цикл завершается. Цикл `Until` выполняется до тех пор, пока указанное в нем условие ложно.

6 Вывод

В ходе выполнения данной лабораторной работы я научился программировать в командном процессоре ОС UNIX, узнал что такое ветвления и циклы.