

Программирование в командном процессоре ОС UNIX. Ветвления и циклы”

Бровкин Александр НБИбд-01-21¹

6 мая, 2022, Москва, Россия

¹Российский Университет Дружбы Народов

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

1. Используя команды `getopts` `grep`, написал 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 ищем в указанном файле нужные строки, определяемые ключом `-р`.

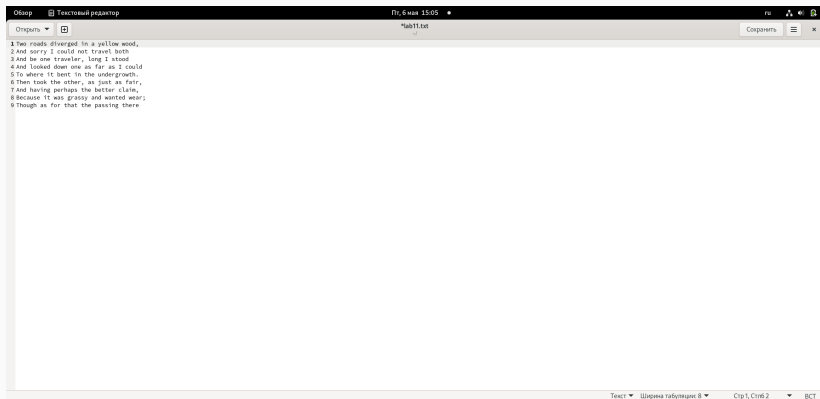


Figure 1: Вставил в файл любой текст из интернета

```
Обзор  Текстовый редактор  Пн, 6 мая 15:41  ru  Сохранить  x
Открыть  labT1.sh
1#!/bin/bash
2iflag=0 oflag=0 pflag=0 cflag=0 nflag=0
3while getopts :o:p:c:n optletter
4do case $optletter in
5  o) iflag=1 sval=$OPTARG;;
6  o) oflag=1 sval=$OPTARG;;
7  p) pflag=1 sval=$OPTARG;;
8  c) cflag=1;;
9  n) nflag=1;;
10 *) echo illegal option $optletter
11 esac
12 done
13 if (($iflag==0))
14 then echo "флагов не найдено"
15 else
16   if (($iflag==0))
17   then echo "опа не найдено"
18   else
19     if (($iflag==0))
20     then if (($cflag==0))
21       then if (($oflag==0))
22         then grep $sval $val
23         else grep -m $sval $val
24         fi
25       else if (($iflag==0))
26         then grep -s $sval $val
27         else grep -s -m $sval $val
28         fi
29       fi
30     else if (($cflag==0))
31     then if (($iflag==0))
32       then grep $sval $val > $val
33       else grep -m $sval $val > $val
34       fi
35     else if (($iflag==0))
36     then grep -s $sval $val > $val
37     else grep -s -m $sval $val > $val
38     fi
39   fi
40 fi
41 fi
42 fi
Сохранение файла в/home/abrown/labT1.sh...  sh  Ширина табуляции: 8  Стр 42, Стр 63  ВСТ
```

Figure 2: Пишу первый скрипт

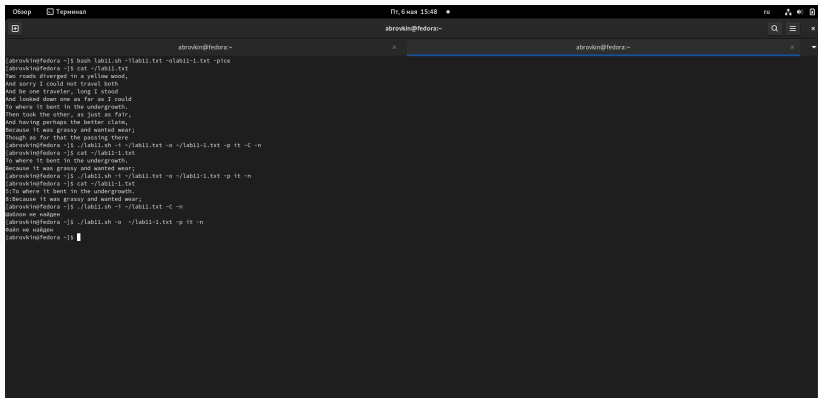


Figure 3: Проверяю в терминале

2. 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)`, 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.

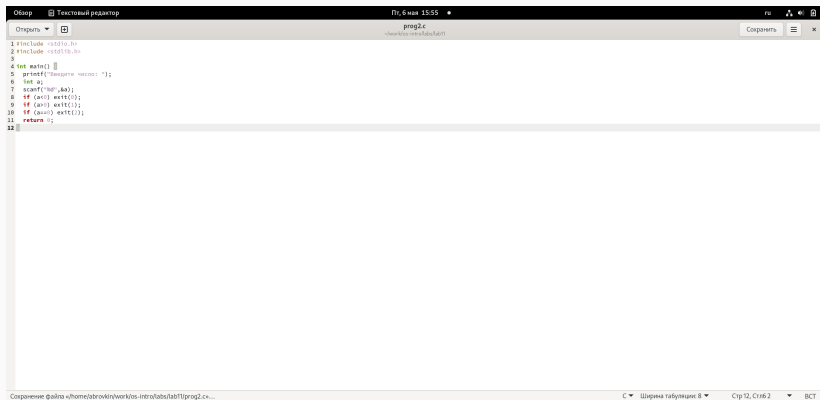


Figure 4: Пишу новый скрипт-на языке Си

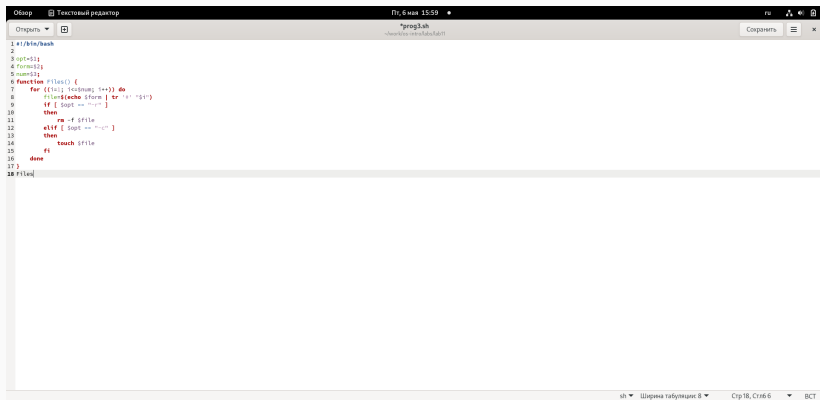

```
1 #!/bin/bash
2
3 gcc prog2.c -o prog2
4 ./prog2
5 codens1?
6 case $code in
7   0) echo "Число меньше 0" ;;
8   1) echo "Число больше 0" ;;
9   2) echo "Число равно 0" ;;
10 esac
```

Figure 5: Пишу еще один скрипт

```
abrovin@fedora:~$ touch prog2.c prog2.sh
abrovin@fedora:~$ chmod +x *.sh
abrovin@fedora:~$ ./prog2.sh
Введите число: 5
Число больше 0
abrovin@fedora:~$ ./prog2.sh
Введите число: -5
Число меньше 0
abrovin@fedora:~$ ./prog2.sh
Введите число: 0
Число равно 0
abrovin@fedora:~$
```

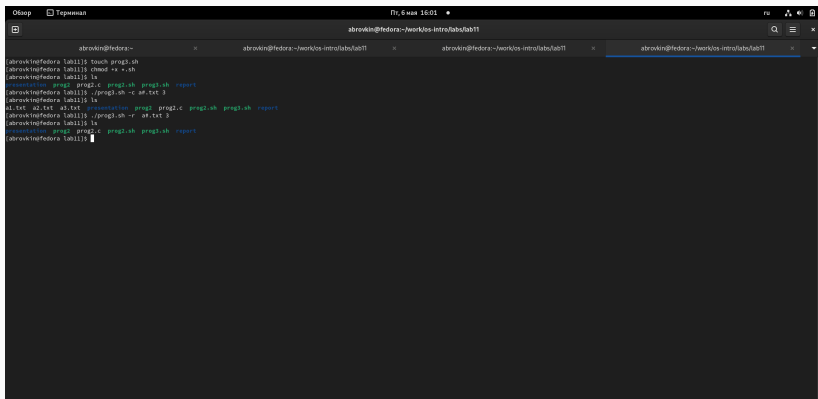
Figure 6: Проверяю все в терминале

3. Написал 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).



```
1 #!/bin/bash
2
3 opt=511
4 form=12
5 num=21
6 function files() {
7     for ((i=1; i<num; i++)) do
8         file=$(echo $form | tr '0' '1')
9         if [ $opt == "-y" ]
10             then
11                 rm -f $file
12             elif [ $opt == "-c" ]
13                 then
14                     touch $file
15             fi
16         done
17     }
18 files
```

Figure 7: Пишу новый скрипт



```
abrovin@fedora:~  
[abrovin@fedora lab11]$ touch prog3.sh  
[abrovin@fedora lab11]$ chmod +x *.sh  
[abrovin@fedora lab11]$ ls  
presentation prog2 prog2.c prog2.sh report  
[abrovin@fedora lab11]$ ./prog3.sh -c a1.txt 3  
[abrovin@fedora lab11]$ ls  
a1.txt a2.txt a3.txt presentation prog1 prog1.c prog2.sh report  
[abrovin@fedora lab11]$ ./prog3.sh -r a1.txt 3  
[abrovin@fedora lab11]$ ls  
presentation prog2 prog2.c prog2.sh report  
[abrovin@fedora lab11]$
```

Figure 8: Проверяю его в терминале

4. 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`).

The image shows a terminal window titled "Текстовый редактор" (Text Editor) with a file named "prog4.sh". The script contains the following commands:

```
1 #!/bin/bash
2
3 files=$(find ./ -maxdepth 1 -mtime -7)
4 listing=""
5 for file in $(ls "$files" | grep -v "\.tar$"); do
6     file=$(echo "$file" | cut -c 3-)
7     listing="$listing $file"
8 done
9 dir=$(basename $prog)
10 tar -cvf $dir.tar $listing
```

The terminal window has a status bar at the bottom showing the file path, shell type, window title, page number, and encoding.

Figure 9: Пишу новый скрипт

```
abrovkin@fedora:~$ touch prog4.sh
abrovkin@fedora:~$ chmod +x *.sh
abrovkin@fedora:~$ ls -l
-rwxr-xr-x 1 abrovkin abrovkin 48 апр 27 17:35 presentation
-rwxr-xr-x 1 abrovkin abrovkin 22822 мая 6 15:57 prog2
-rwxr-xr-x 1 abrovkin abrovkin 198 мая 6 15:55 prog2.c
-rwxr-xr-x 1 abrovkin abrovkin 193 мая 6 15:56 prog3.sh
-rwxr-xr-x 1 abrovkin abrovkin 225 мая 6 15:59 prog3.sh
-rwxr-xr-x 1 abrovkin abrovkin 210 мая 6 16:03 prog4.sh
abrovkin@fedora:~$ sudo -i
[sudo] пароль адм abrovkin:
prog2.c
prog2.sh
prog2
prog3.sh
prog4.sh
abrovkin@fedora:~$ tar -cf lab11.tar
prog2.c
prog2.sh
prog2
prog3.sh
prog4.sh
abrovkin@fedora:~$
```

Figure 10: Проверил его в терминале

Вывод:

Изучил основы программирования в оболочке ОС UNIX, научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.