

Лабораторная работа №11

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Заболотная Кристина

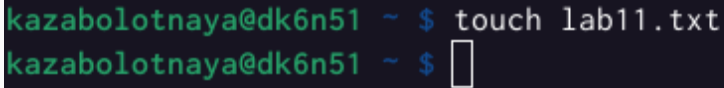
Российский университет дружбы народов, Москва, Россия

Информация

- Заболотная Кристина Александровна
- Студент группы НБИбд-01-22
- Российский 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`, написан 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`.

A terminal window with a dark background. The prompt is 'kazabolotnaya@dk6n51 ~ \$'. The first line shows the command 'touch lab11.txt' being entered. The second line shows the prompt again with a cursor, indicating the command has been executed.

```
kazabolotnaya@dk6n51 ~ $ touch lab11.txt
kazabolotnaya@dk6n51 ~ $
```

Рис. 1: создаем lab11.txt

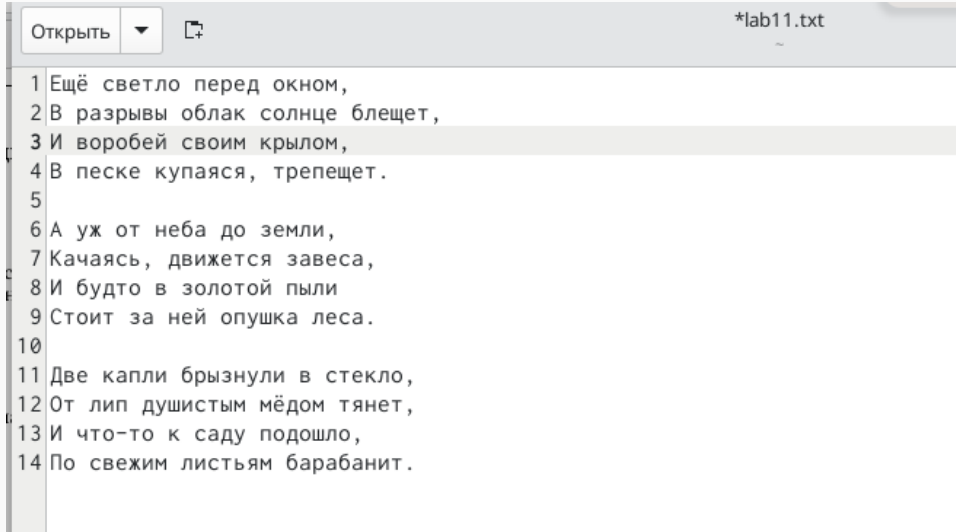


Рис. 2: текст из интернета

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

Рис. 3: первый скрипт

```
kazabolotnaya@dk6n51 ~ $ cat ~/lab11.txt
```

Ещё светло перед окном,
В разрывы облак солнце блещет,
И воробей своим крылом,
В песке купаяся, трепещет.

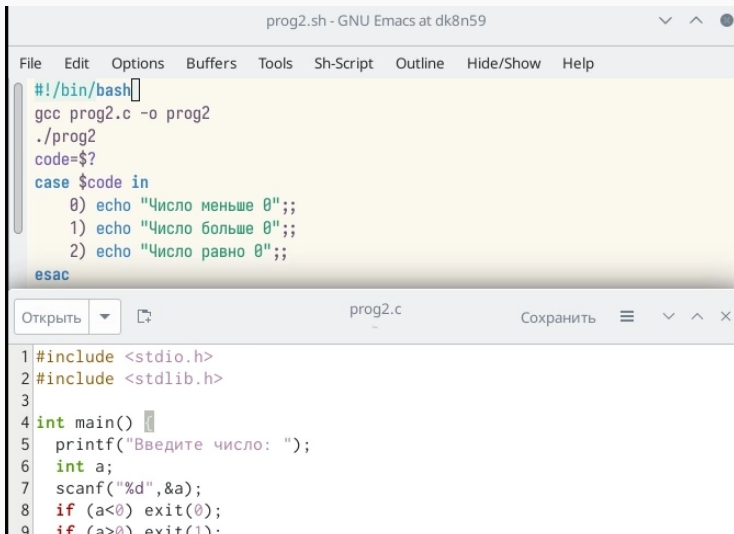
А уж от неба до земли,
Качаясь, движется завеса,
И будто в золотой пыли
Стоит за ней опушка леса.

Две капли брызнули в стекло,
От лип душистым мёдом тянет,
И что-то к саду подошло,
По свежим листьям барабанит.

```
kazabolotnaya@dk6n51 ~ $ █
```

Рис. 4: проверяем lab11.txt

2. Написана на языке Си программа, 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.



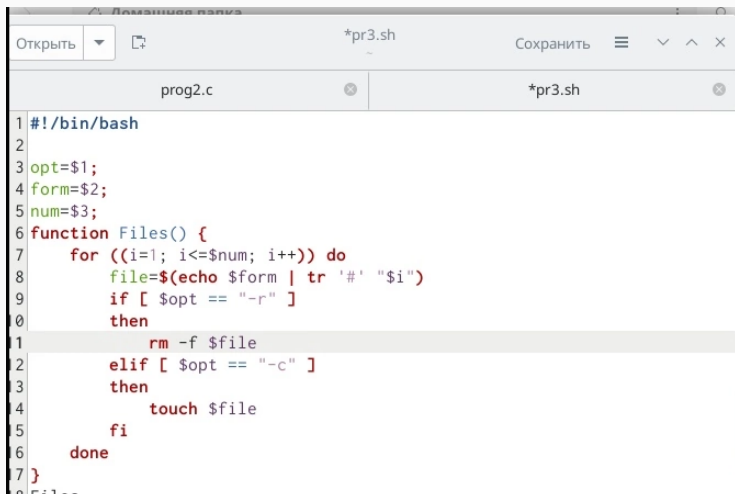
The screenshot shows a GNU Emacs editor window titled "prog2.sh - GNU Emacs at dk8n59". The window is split into two panes. The top pane displays a shell script for "prog2.sh" with the following content:

```
#!/bin/bash
gcc prog2.c -o prog2
./prog2
code=$?
case $code in
    0) echo "Число меньше 0";;
    1) echo "Число больше 0";;
    2) echo "Число равно 0";;
esac
```

The bottom pane displays the C source code for "prog2.c" with the following content:

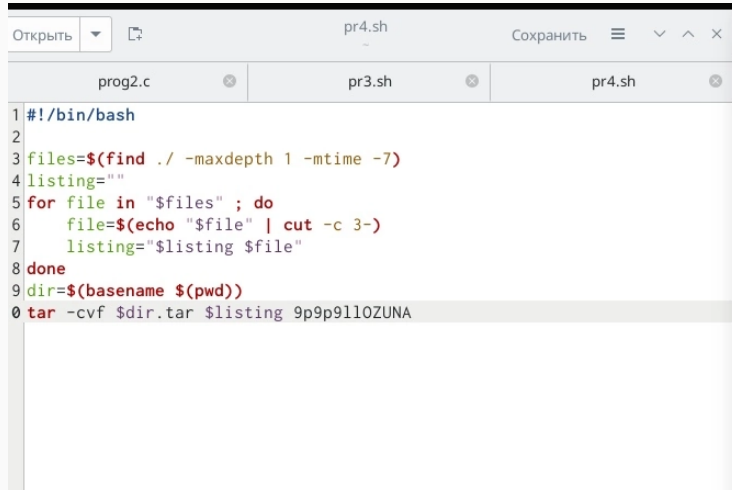
```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     printf("Введите число: ");
7     int a;
8     scanf("%d", &a);
9     if (a < 0) exit(0);
10    if (a > 0) exit(1);
11    }
```

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).



```
1 #!/bin/bash
2
3 opt=$1;
4 form=$2;
5 num=$3;
6 function Files() {
7     for ((i=1; i<=$num; i++)) do
8         file=$(echo $form | tr '#' "$i")
9         if [ $opt == "-r" ]
0         then
1             rm -f $file
2         elif [ $opt == "-c" ]
3         then
4             touch $file
5         fi
6     done
7 }
```

4. Написан 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`).



The screenshot shows a code editor window with three tabs: `prog2.c`, `pr3.sh`, and `pr4.sh`. The `pr4.sh` tab is active, displaying a shell script. The script uses `find` to locate files modified within the last 7 days and then uses `tar` to archive them. The script is as follows:

```
1 #!/bin/bash
2
3 files=$(find ./ -maxdepth 1 -mtime -7)
4 listing=""
5 for file in "$files" ; do
6     file=$(echo "$file" | cut -c 3-)
7     listing="$listing $file"
8 done
9 dir=$(basename $(pwd))
10 tar -cvf $dir.tar $listing 9p9p9l10ZUNA
```

В ходе выполнения данной лабораторной работы мы изучили основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.