

Операционные системы

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Альбина Тяпкова

17 апреля 2025

Российский университет дружбы народов, Москва, Россия

Цели и задачи работы

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

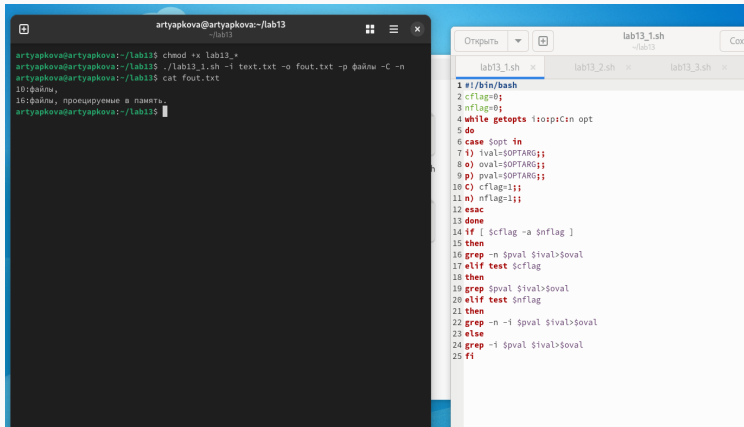
1 Выполнить 4 задания

Процесс выполнения лабораторной работы

1. Используя команды `getopts` `grep` напишем командный файл, который анализирует командную строку с ключами и выполним его: `-i inputfile` — прочитать данные из указанного файла; `-o outputfile` — вывести данные в указанный файл; `-p шаблон` — указать шаблон для поиска; `-C` — различать большие и малые буквы; `-n` — выдавать номера строк;

а затем ищет в указанном файле нужные строки

Выполнение работы



The image shows two overlapping windows from a Linux desktop environment. The background window is a terminal titled 'артыapkova@артыapkova:~/lab13'. It contains the following commands and output:

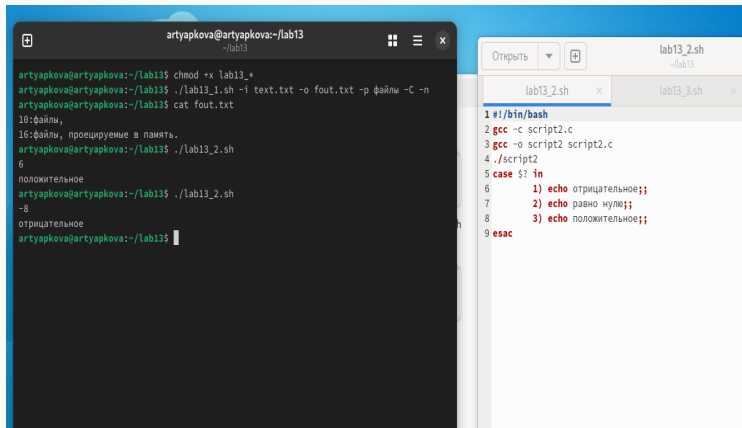
```
артыapkova@артыapkova:~/lab13$ chmod +x lab13_*
артыapkova@артыapkova:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C -n
артыapkova@артыapkova:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
артыapkova@артыapkova:~/lab13$
```

The foreground window is a code editor titled 'lab13_1.sh' with tabs for 'lab13_1.sh', 'lab13_2.sh', and 'lab13_3.sh'. It displays the source code of the program executed in the terminal:

```
1#!/bin/bash
2cflag=0;
3nflag=0;
4while getopts i:o:p:C:n opt
5do
6case $opt in
7i) ival=$OPTARG;;
8o) oval=$OPTARG;;
9p) pval=$OPTARG;;
10C) cflag=1;;
11n) nflag=1;;
12esac
13done
14if [ $cflag -a $nflag ]
15then
16grep -n $pval $ival>$oval
17elif test $cflag
18then
19grep $pval $ival>$oval
20elif test $nflag
21then
22grep -n -i $pval $ival>$oval
23else
24grep -i $pval $ival>$oval
25fi
```

Рис. 1: Задание 1

2. Напишем сначала на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю. Затем завершим программу при помощи функции `exit(n)`, передавая информацию о коде завершения в оболочку. Командный файл вызовет эту программу и, проанализировав с помощью команды `$?`, выдаст сообщение о том, какое число было введено



The image shows a terminal window on the left and a code editor on the right, both displaying shell script execution. The terminal window has a title bar 'artyapkova@artyapkova:~/lab13' and shows the following commands and output:

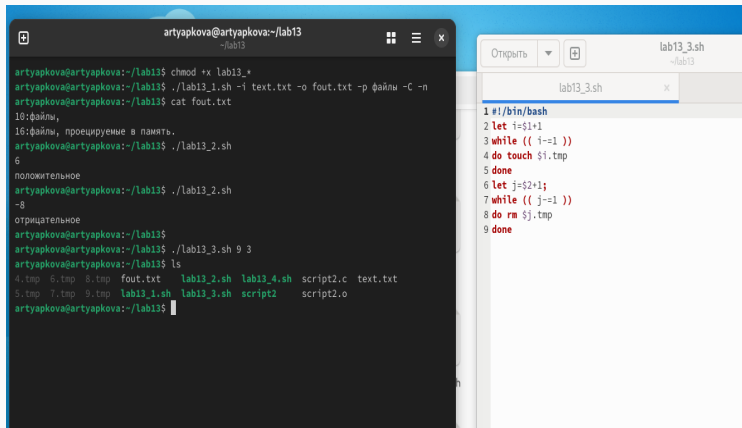
```
artyapkova@artyapkova:~/lab13$ chmod +x lab13_*
artyapkova@artyapkova:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C -n
artyapkova@artyapkova:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
artyapkova@artyapkova:~/lab13$ ./lab13_2.sh
6
положительное
artyapkova@artyapkova:~/lab13$ ./lab13_2.sh
-8
отрицательное
artyapkova@artyapkova:~/lab13$
```

The code editor on the right has a title bar 'lab13_2.sh' and shows the following script content:

```
1 #!/bin/bash
2 gcc -c script2.c
3 gcc -o script2 script2.c
4 ./script2
5 case $? in
6     1) echo отрицательное;;
7     2) echo равно нулю;;
8     3) echo положительное;;
9 esac
```

Рис. 2: Задание 2

3. Напишем командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N



The image shows a terminal window and a file editor side-by-side. The terminal window, titled 'artyapkova@artyapkova:~/lab13', shows the execution of a shell script 'lab13_1.sh' which creates a file 'fout.txt' and lists its contents. The file contains a list of files and directories. The terminal also shows the execution of 'lab13_2.sh' and 'lab13_3.sh'. The file editor, titled 'lab13_3.sh', shows the source code of the script, which is a bash script that creates a file 'fout.txt' and lists its contents.

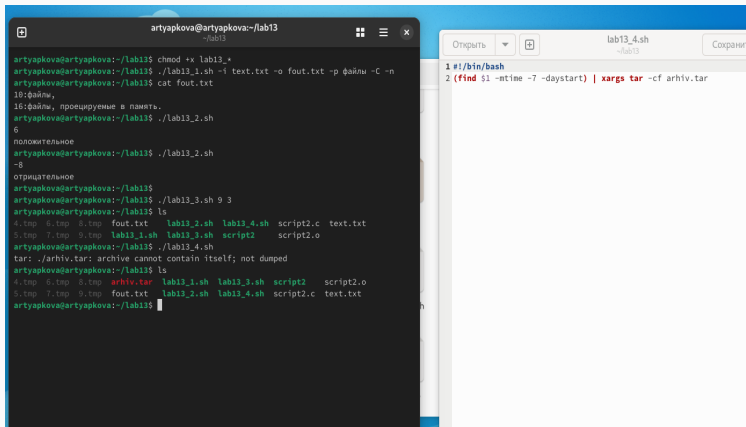
```
artyapkova@artyapkova:~/lab13$ chmod +x lab13_*
artyapkova@artyapkova:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C -n
artyapkova@artyapkova:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
artyapkova@artyapkova:~/lab13$ ./lab13_2.sh
6
положительное
artyapkova@artyapkova:~/lab13$ ./lab13_2.sh
-8
отрицательное
artyapkova@artyapkova:~/lab13$
artyapkova@artyapkova:~/lab13$ ./lab13_3.sh 9 3
artyapkova@artyapkova:~/lab13$ ls
4.tmp 6.tmp 8.tmp fout.txt lab13_2.sh lab13_4.sh script2.c text.txt
5.tmp 7.tmp 9.tmp lab13_1.sh lab13_3.sh script2 script2.o
artyapkova@artyapkova:~/lab13$
```

```
1 #!/bin/bash
2 let i=$1+1
3 while (( i--=1 ))
4 do touch $i.tmp
5 done
6 let j=$2+1;
7 while (( j--=1 ))
8 do rm $j.tmp
9 done
```

Рис. 3: Задание 3

4. Напишем командный файл, который 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.

Выполнение работы



The image shows a terminal window and a file manager window. The terminal window, titled 'artyapkova@artyapkova:~/lab13', displays the following commands and output:

```
artyapkova@artyapkova:~/lab13$ chmod +x lab13_*
artyapkova@artyapkova:~/lab13$ ./lab13_1.sh -i text.txt -o fout.txt -p файлы -C -n
artyapkova@artyapkova:~/lab13$ cat fout.txt
10:файлы,
16:файлы, проецируемые в память.
artyapkova@artyapkova:~/lab13$ ./lab13_2.sh
6
положительное
artyapkova@artyapkova:~/lab13$ ./lab13_2.sh
-8
отрицательное
artyapkova@artyapkova:~/lab13$
artyapkova@artyapkova:~/lab13$ ./lab13_3.sh 9 3
artyapkova@artyapkova:~/lab13$ ls
4.tmp 6.tmp 8.tmp  fout.txt  lab13_2.sh  lab13_4.sh  script2.c  text.txt
5.tmp 7.tmp 9.tmp  lab13_1.sh  lab13_3.sh  script2   script2.o
artyapkova@artyapkova:~/lab13$ ./lab13_4.sh
tar: ./arhiv.tar: archive cannot contain itself; not dumped
artyapkova@artyapkova:~/lab13$ ls
4.tmp 6.tmp 8.tmp  arhiv.tar  lab13_1.sh  lab13_3.sh  script2   script2.o
5.tmp 7.tmp 9.tmp  fout.txt  lab13_2.sh  lab13_4.sh  script2.c  text.txt
artyapkova@artyapkova:~/lab13$
```

The file manager window, titled 'lab13_4.sh', shows the following commands:

```
1 #!/bin/bash
2 (find $1 -mtime -7 -daystart) | xargs tar -cf arhiv.tar
```

Рис. 4: Задание 4

Выводы по проделанной работе

В данной работе мы изучили основы программирования в оболочке ОС UNIX и писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.