

Лабораторная работа № 11

Сильвен Макс Грегор Филс , НКАбд-03-22

21 Апрель 2023

Российский университет дружбы народов, Москва, Россия

- 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`, написать командный файл, который анализирует командную строку с ключами:
 - `-iinputfile` — прочитать данные из указанного файла;
 - `-ooutputfile` — вывести данные в указанный файл;
 - `-ршаблон` — указать шаблон для поиска;
 - `-C` — различать большие и малые буквы;
 - `-n` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-р`.

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.

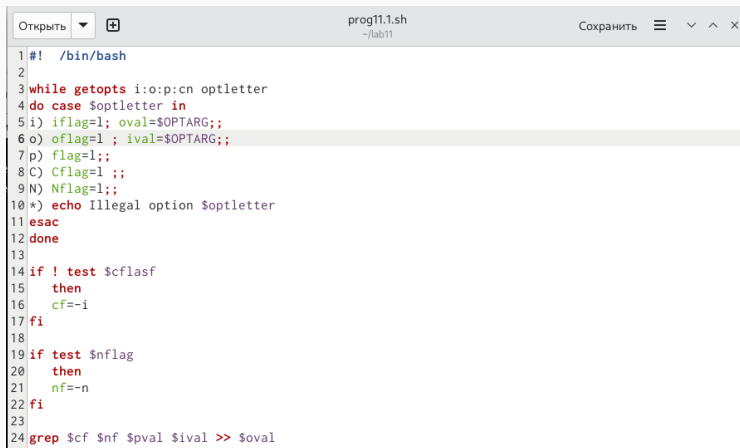
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).
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).

1. Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера. В операционных системах типа UNIX/Linux наиболее часто используются следующие реализации командных оболочек:
 - оболочка Борна (Bourne shell или sh) — стандартная командная оболочка UNIX/Linux, содержащая базовый, но при этом полный набор функций;
 - С-оболочка (или csh) — надстройка на оболочкой Борна, использующая С-подобный синтаксис команд с возможностью сохранения истории выполнения команд;
 - оболочка Корна (или ksh) — напоминает оболочку С, но операторы управления программой совместимы с операторами оболочки Борна;
 - BASH — сокращение от Bourne Again Shell (опять оболочка Борна), в основе своей совмещает свойства оболочек С и Корна (разработка компании Free Software

2. POSIX (Portable Operating System Interface for Computer Environments) — набор стандартов описания интерфейсов взаимодействия операционной системы и прикладных программ. Стандарты POSIX разработаны комитетом IEEE (Institute of Electrical and Electronics Engineers) для обеспечения совместимости различных UNIX/Linux-подобных операционных систем и переносимости прикладных программ на уровне исходного кода. POSIX-совместимые оболочки разработаны на базе оболочки Корна. Рассмотрим основные элементы программирования в оболочке `bash`. В других оболочках большинство команд будет совпадать с описанными ниже.

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;
 - `-ршаблон` — указать шаблон для поиска;
 - `-C` — различать большие и малые буквы;
 - `-n` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-р`. (рис. (fig:001?; fig:002?; fig:003?; fig:004?))

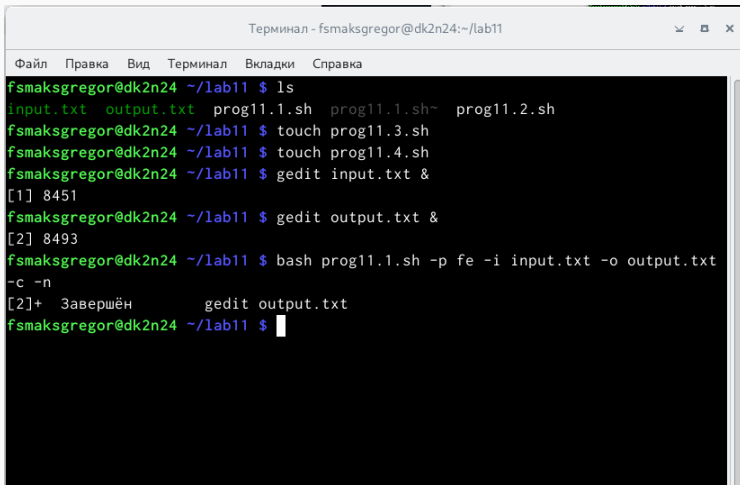
Выполнение лабораторной работы



```
1 #! /bin/bash
2
3 while getopts i:o:p:cn optletter
4 do case $optletter in
5 i) iflag=1; oval=$OPTARG;;
6 o) oflag=1 ; ival=$OPTARG;;
7 p) flag=1;;
8 C) Cflag=1 ;;
9 N) Nflag=1;;
10 *) echo Illegal option $optletter
11 esac
12 done
13
14 if ! test $cflag
15 then
16   cf=-i
17 fi
18
19 if test $nflag
20 then
21   nf=-n
22 fi
23
24 grep $cf $nf $pval $ival >> $oval
```

Рис. 1: Первая программа

Выполнение лабораторной работы



```
Терминал - fsmaksgregor@dk2n24:~/lab11
Файл  Правка  Вид  Терминал  Вкладки  Справка
fsmaksgregor@dk2n24 ~/lab11 $ ls
input.txt  output.txt  prog11.1.sh  prog11.1.sh~  prog11.2.sh
fsmaksgregor@dk2n24 ~/lab11 $ touch prog11.3.sh
fsmaksgregor@dk2n24 ~/lab11 $ touch prog11.4.sh
fsmaksgregor@dk2n24 ~/lab11 $ gedit input.txt &
[1] 8451
fsmaksgregor@dk2n24 ~/lab11 $ gedit output.txt &
[2] 8493
fsmaksgregor@dk2n24 ~/lab11 $ bash prog11.1.sh -p fe -i input.txt -o output.txt
-c -n
[2]+  Завершён          gedit output.txt
fsmaksgregor@dk2n24 ~/lab11 $
```

Рис. 2: Вызов программы в терминале

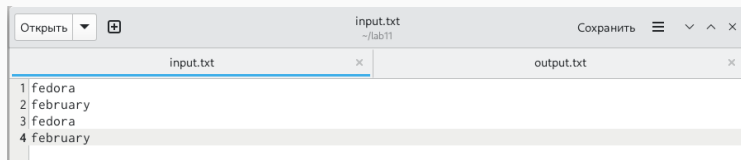


Рис. 3: Результат

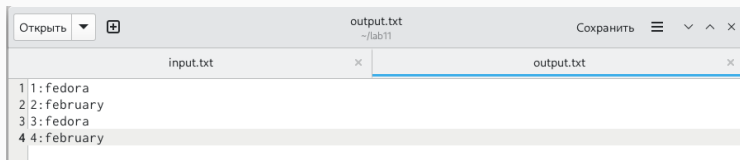
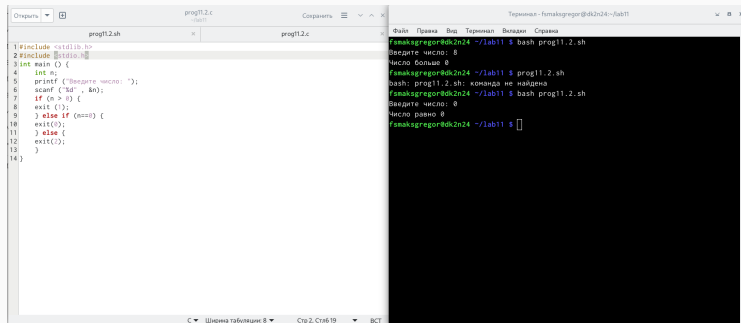


Рис. 4: Результат

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. (рис. (fig:005?; fig:006?; fig:007?))

Выполнение лабораторной работы



The image shows a code editor window with a C program named `prog11.2.c` and a terminal window showing its execution.

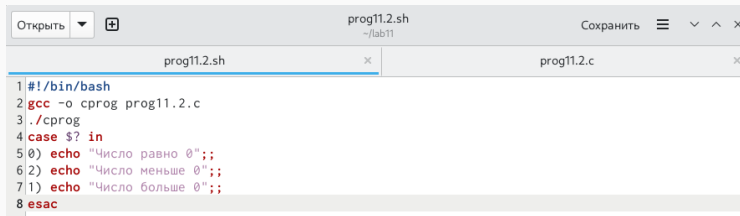
Code Editor (prog11.2.c):

```
1 #include <stdlib.h>
2 #include <stdio.h>
3 int main () {
4     int n;
5     printf ("Введите число: ");
6     scanf ("%d", &n);
7     if (n > 0) {
8         exit (1);
9     } else if (n == 0) {
10        exit (0);
11    } else {
12        exit (1);
13    }
14 }
```

Terminal:

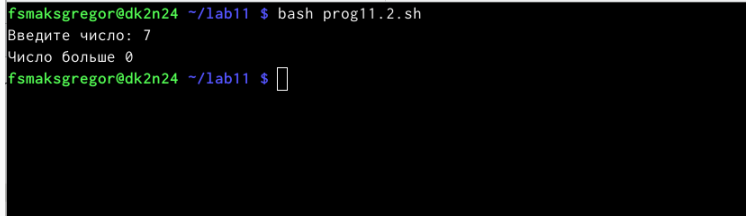
```
fsmaksgregor@dk2n24 ~/lab11 $ bash prog11.2.sh
Введите число: 8
Число больше 0
fsmaksgregor@dk2n24 ~/lab11 $ prog11.2.sh
bash: prog11.2.sh: команда не найдена
fsmaksgregor@dk2n24 ~/lab11 $ bash prog11.2.sh
Введите число: 0
Число равно 0
fsmaksgregor@dk2n24 ~/lab11 $
```

Рис. 5: Вторая программа



```
1 #!/bin/bash
2 gcc -o cprog prog11.2.c
3 ./cprog
4 case $? in
5 0) echo "Число равно 0";;
6 2) echo "Число меньше 0";;
7 1) echo "Число больше 0";;
8 esac
```

Рис. 6: Вторая программа



```
fsmaksgregor@dk2n24 ~/lab11 $ bash prog11.2.sh
Введите число: 7
Число больше 0
fsmaksgregor@dk2n24 ~/lab11 $
```

The image shows a terminal window with a black background and green text. The prompt is 'fsmaksgregor@dk2n24 ~/lab11 \$'. The user enters 'bash prog11.2.sh'. The program outputs 'Введите число: 7' (Enter number: 7) and 'Число больше 0' (Number is greater than 0). The prompt returns to 'fsmaksgregor@dk2n24 ~/lab11 \$' with a cursor.

Рис. 7: Результат

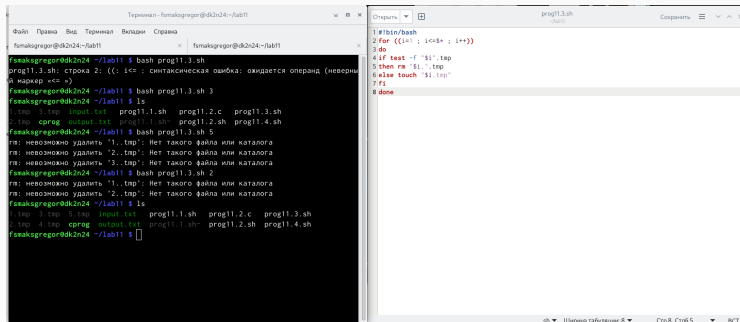
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если они существуют).
(рис. (fig:008?; fig:009?))

A screenshot of a code editor window. The title bar shows 'prog11.3.sh' and the file path '~/.lab11'. The editor contains a shell script with 8 lines of code. The code is as follows:

```
1 #!/bin/bash
2 for ((i=1 ; i<=$* ; i++))
3 do
4 if test -f "$i".tmp
5 then rm "$i".tmp
6 else touch "$i.tmp"
7 fi
8 done
```

Рис. 8: Третья программа

Выполнение лабораторной работы



```
Терминал - fsmaksregor@dk2n24:~/lab11
Файл Правка Вид Терминал Выход Справка
fsmaksregor@dk2n24:~/lab11 fsmaksregor@dk2n24:~/lab11

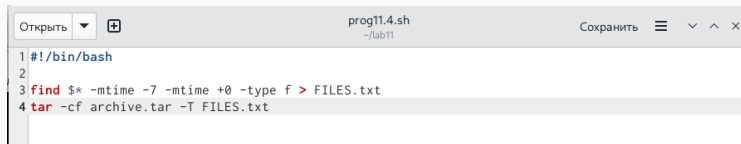
fsmaksregor@dk2n24:~/lab11 $ bash prog11.3.sh
prog11.3.sh: строка 2: ((: i<= : синтаксическая ошибка: ожидается операнд (неверный
маркер «<= »)
fsmaksregor@dk2n24:~/lab11 $ bash prog11.3.sh 3
fsmaksregor@dk2n24:~/lab11 $ ls
1.tmp 3.tmp input.txt prog11.1.sh prog11.2.c prog11.3.sh
2.tmp cprog output.txt prog11.1.sh prog11.2.sh prog11.4.sh
fsmaksregor@dk2n24:~/lab11 $ bash prog11.3.sh 5
гв: невозможно удалить '1..tmp': Нет такого файла или каталога
гв: невозможно удалить '2..tmp': Нет такого файла или каталога
гв: невозможно удалить '3..tmp': Нет такого файла или каталога
fsmaksregor@dk2n24:~/lab11 $ bash prog11.3.sh 2
гв: невозможно удалить '1..tmp': Нет такого файла или каталога
гв: невозможно удалить '2..tmp': Нет такого файла или каталога
fsmaksregor@dk2n24:~/lab11 $ ls
1.tmp 3.tmp 5.tmp input.txt prog11.1.sh prog11.2.c prog11.3.sh
2.tmp 4.tmp cprog output.txt prog11.1.sh prog11.2.sh prog11.4.sh
fsmaksregor@dk2n24:~/lab11 $
```

```
Открыть
prog11.3.sh
~lab11
Сохранить
1 #!/bin/bash
2 for ((i=1; i<=5; i++))
3 do
4 if test -f "$i".tmp
5 then rm "$i".tmp
6 else touch "$i.tmp"
7 fi
8 done
```

sh Ширина табуляции: 8 Стр 8, Стр 5 ВСТ

Рис. 9: Результат

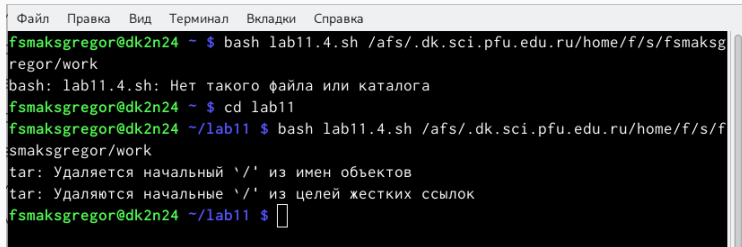
4. Написать командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду `find`). (рис. (fig:010?; fig:011?; fig:012?))



The image shows a code editor window with a light gray header bar. On the left of the header is a button labeled 'Открыть' (Open) with a dropdown arrow and a plus icon. In the center of the header, the file name 'prog11.4.sh' is displayed above the path '~/lab11'. On the right of the header is a button labeled 'Сохранить' (Save) followed by a hamburger menu icon, and then three small icons: a downward arrow, an upward arrow, and a close 'X' icon. The main area of the editor contains four lines of text: line 1 is '#!/bin/bash' in blue; line 2 is empty; line 3 is 'find \$* -mtime -7 -mtime +0 -type f > FILES.txt' with 'find' in red; and line 4 is 'tar -cf archive.tar -T FILES.txt' with 'tar' in red. A vertical line on the left side of the editor indicates the current line being edited.

```
1 #!/bin/bash
2
3 find $* -mtime -7 -mtime +0 -type f > FILES.txt
4 tar -cf archive.tar -T FILES.txt
```

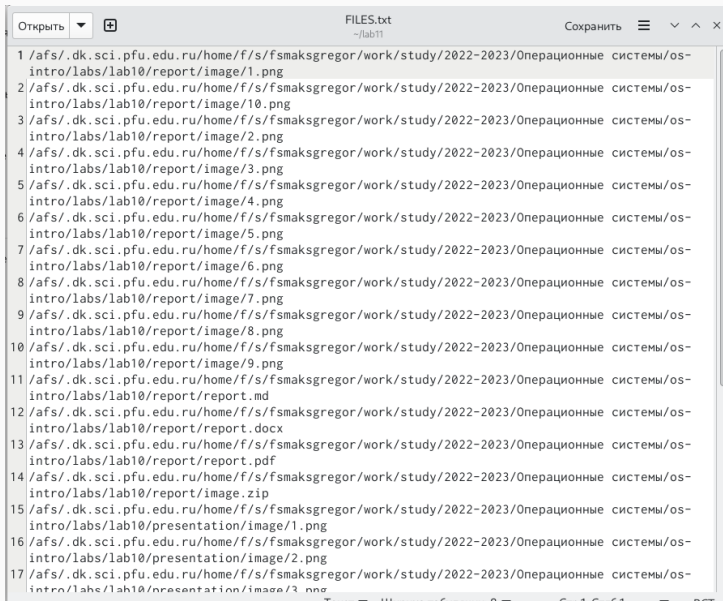
Рис. 10: Четвертая программа



```
Файл  Правка  Вид  Терминал  Вкладки  Справка
fsmaksgregor@dk2n24 ~ $ bash lab11.4.sh /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work
bash: lab11.4.sh: Нет такого файла или каталога
fsmaksgregor@dk2n24 ~ $ cd lab11
fsmaksgregor@dk2n24 ~/lab11 $ bash lab11.4.sh /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work
tar: Удаляется начальный '/' из имен объектов
tar: Удаляются начальные '/' из целей жестких ссылок
fsmaksgregor@dk2n24 ~/lab11 $
```

Рис. 11: Вызов программы в терминале

Выполнение лабораторной работы



The screenshot shows a text editor window titled "FILES.txt" with a path of "~/lab11". The window has a menu bar with "Открыть" (Open), "Сохранить" (Save), and standard window controls. The main area contains a list of 17 files, each on a new line, numbered 1 through 17. The files are located in the directory ~/afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-intro/labs/lab10/report/ and ~/afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-intro/labs/lab10/presentation/. The files include images (1.png to 10.png, 11.png to 13.png, 14.png, 15.png, 16.png, 17.png), a report (report.md), a document (report.docx), a PDF (report.pdf), and a ZIP file (image.zip).

```
1 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/1.png  
2 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/10.png  
3 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/2.png  
4 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/3.png  
5 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/4.png  
6 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/5.png  
7 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/6.png  
8 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/7.png  
9 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/8.png  
10 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image/9.png  
11 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/report.md  
12 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/report.docx  
13 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/report.pdf  
14 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/report/image.zip  
15 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/presentation/image/1.png  
16 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/presentation/image/2.png  
17 /afs/.dk.sci.pfu.edu.ru/home/f/s/fsmaksgregor/work/study/2022-2023/Операционные системы/os-  
intro/labs/lab10/presentation/image/3.png
```

В процессе выполнения данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX. Научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

1. Лабораторная работа № 10. Программирование в командном процессоре ОС UNIX. Командные файлы [Электронный ресурс]. URL: <https://esystem.rudn.ru/>.

спасибо за внимание!