

Лабораторная работа №13

Операционные системы

Павлова Татьяна Юрьевна

Содержание

| | | |
|----------|---------------------------------------|-----------|
| 1 | Цель работы | 5 |
| 2 | Задание | 6 |
| 3 | Теоретическое введение | 7 |
| 4 | Выполнение лабораторной работы | 9 |
| 5 | Выводы | 15 |

Список иллюстраций

| | | |
|------|----------------------------|----|
| 4.1 | lab13-1.sh | 9 |
| 4.2 | Компиляция файла | 9 |
| 4.3 | input.txt | 10 |
| 4.4 | output.txt | 10 |
| 4.5 | 2.c | 11 |
| 4.6 | lab13-2.sh | 11 |
| 4.7 | Компиляция файла | 12 |
| 4.8 | lab13-3.sh | 12 |
| 4.9 | Компиляция файла | 12 |
| 4.10 | lab13-4.sh | 13 |
| 4.11 | Компиляция файла | 13 |
| 4.12 | archive.tar | 14 |

Список таблиц

1 Цель работы

Целью данной лабораторной работы является изучение основ программирования в оболочке ОС UNIX, а также научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

2 Задание

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами: `-i` input file — прочитать данные из указанного файла; `-o` output file — 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 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 `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`).

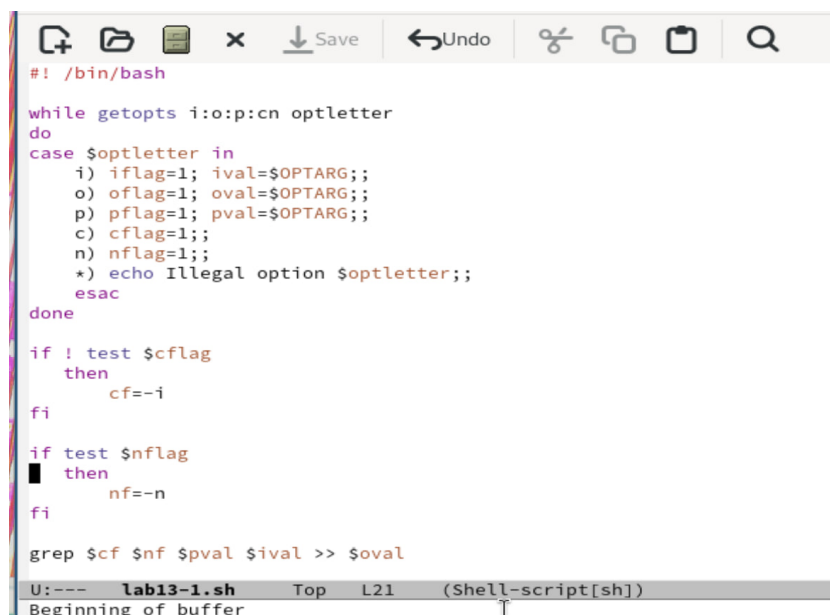
3 Теоретическое введение

Bash (Bourne Again Shell) — это мощная командная оболочка Unix, которая используется для выполнения различных задач в терминале. Bash предоставляет интерактивный интерфейс, в котором пользователи могут вводить команды, а затем получать результаты. Она также поддерживает скрипты оболочки, которые представляют собой текстовые файлы, содержащие последовательность команд Bash для автоматизации задач. Bash широко используется в средах Unix и Linux, а также поддерживается Windows с помощью подсистемы Windows для Linux (WSL). Перечислим основные возможности этой оболочки. Обработка команд. Bash может обрабатывать как простые, так и сложные команды. Простые состоят из одного действия и, возможно, некоторых аргументов. Сложные команды могут содержать несколько простых, объединенных с помощью операторов конвейера (`|`), перенаправления ввода и вывода (`<`, `>`, `>>`), условных операторов (`if/else`, `case/esac`, `while/do`). О них мы расскажем позже. Расширенный ввод, редактирование строк. Оболочка предоставляет функции расширенного ввода, такие как автодополнение, которое предлагает возможные варианты завершения команд и имен файлов по мере их ввода. Она также поддерживает историю, позволяя пользователям просматривать ранее введенные команды, а затем повторно их использовать. Возможность создания и запуска скриптов. Скрипты оболочки — это текстовые файлы, содержащие последовательность команд. Их можно создавать с помощью текстового редактора, а затем запускать в терминале, что дает возможность пользователям автоматизировать задачи и управлять системой. Скрипты могут содержать условные операторы, циклы, функции для обеспече-

ния дополнительной гибкости и контроля.

4 Выполнение лабораторной работы

Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами: `-i` input file — прочитать данные из указанного файла; `-o` output file — 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, определяемые ключом `-p` (рис. 1), (рис. 2), (рис. 3), (рис. 4).



```
#!/bin/bash

while getopts i:o:p:cn optletter
do
case $optletter in
  i) iflag=1; ival=$OPTARG;;
  o) oflag=1; oval=$OPTARG;;
  p) pflag=1; pval=$OPTARG;;
  c) cflag=1;;
  n) nflag=1;;
  *) echo Illegal option $optletter;;
  esac
done

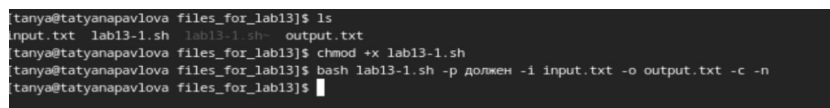
if ! test $cflag
then
  cf=-i
fi

if test $nflag
then
  nf=-n
fi

grep $cf $nf $pval $ival >> $oval
```

U:--- lab13-1.sh Top L21 (Shell-script[sh])
Beginning of buffer

Рис. 4.1: lab13-1.sh



```
tanya@tatyana-pavlova files_for_lab13]$ ls
input.txt lab13-1.sh output.txt
tanya@tatyana-pavlova files_for_lab13$ cat lab13-1.sh
tanya@tatyana-pavlova files_for_lab13$ chmod +x lab13-1.sh
tanya@tatyana-pavlova files_for_lab13$ bash lab13-1.sh -p должен -i input.txt -o output.txt -c -n
tanya@tatyana-pavlova files_for_lab13$
```

Рис. 4.2: Компиляция файла

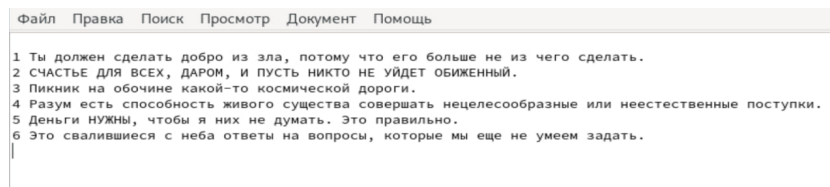


Рис. 4.3: input.txt

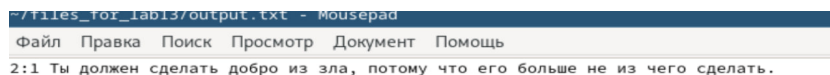


Рис. 4.4: output.txt

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. 5), (рис. 6), (рис. 7).

```

#include <stdlib.h>
#include <stdio.h>

int main () {
    int n;
    printf ("Введите число: ");
    scanf ("%d", &n);
    if(n>0){
        exit(1);
    }
    else if (n==0) {
        exit(0);
    }
    else {
        exit(2);
    }
}

```

Рис. 4.5: 2.c

```

#!/bin/bash

gcc -o cprog 2.c
./cprog
case $? in
0) echo "Число равно нулю";;
1) echo "Число больше нуля";;
2) echo "Число меньше нуля";;
esac

```

Рис. 4.6: lab13-2.sh

```
[tanya@tatyana-pavlova files_for_lab13]$ bash lab13-2.sh
Введите число: 6
Число больше нуля
[tanya@tatyana-pavlova files_for_lab13]$
```

Рис. 4.7: Компиляция файла

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. 8), (рис. 9).

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

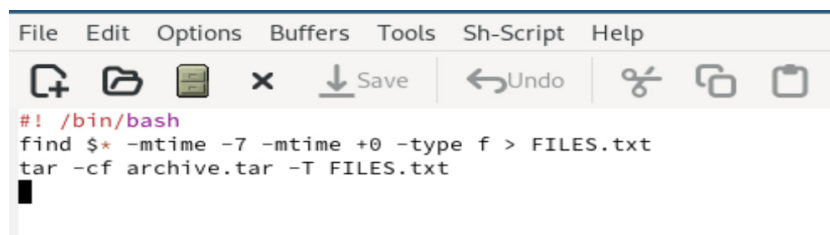
Рис. 4.8: lab13-3.sh

```
[tanya@tatyana-pavlova files_for_lab13]$ bash lab13-3.sh 4
[tanya@tatyana-pavlova files_for_lab13]$ ls
1.tmp  2.c~  3.tmp  cprog  lab13-1.sh  lab13-2.sh  lab13-3.sh  output.txt
2.c    2.tmp  4.tmp  input.txt  lab13-1.sh~  lab13-2.sh~  lab13-3.sh~
```

Рис. 4.9: Компиляция файла

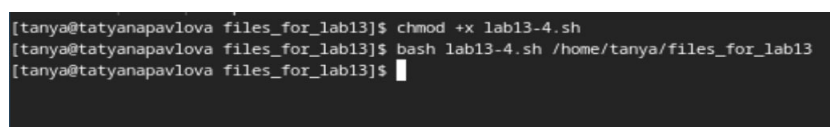
Написать командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировать его так, чтобы

запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find) (рис. 10), (рис. 11), (рис. 12).

A screenshot of a text editor window with a menu bar (File, Edit, Options, Buffers, Tools, Sh-Script, Help) and a toolbar with icons for file operations. The editor contains a shell script with the following content:

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

Рис. 4.10: lab13-4.sh

A screenshot of a terminal window showing the execution of the script. The prompt is [tanya@tatyapavlova files_for_lab13]. The user enters 'chmod +x lab13-4.sh', then 'bash lab13-4.sh /home/tanya/files_for_lab13', and finally 'tar -cf archive.tar -T FILES.txt'. The output shows the files being processed and the archive being created.

```
[tanya@tatyapavlova files_for_lab13]$ chmod +x lab13-4.sh
[tanya@tatyapavlova files_for_lab13]$ bash lab13-4.sh /home/tanya/files_for_lab13
[tanya@tatyapavlova files_for_lab13]$
```

Рис. 4.11: Компиляция файла



archive.tar

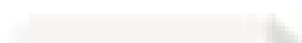


Рис. 4.12: archive.tar

5 Выводы

При выполнении данной лабораторной работы, я изучила основы программирования в оболочке ОС UNIX, а также научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.