

Лабораторная работа 11

Гэинэ Андрей

Содержание

Цель работы	3
Задание	4
Ход работы	5
Выводы	8

Цель работы

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

Задание

1. Используя команды `getopts` `grep`, написать 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`.
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 ∞ (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`).

Ход работы

Код задания 1.

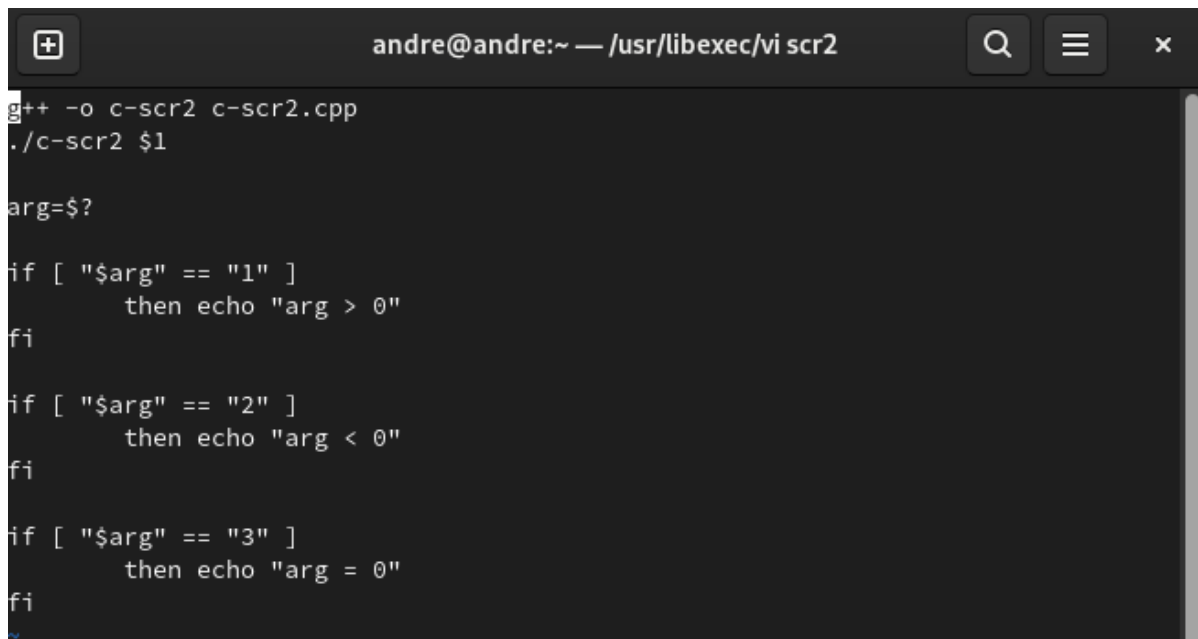
A screenshot of a terminal window with a dark background. The window title bar shows 'andre@andre:~ — /usr/libexec/vi scr1'. The terminal contains a shell script for task 1. The script uses a while loop with getopt to parse command-line options: -i (input file), -o (output file), -p (pattern), -c (case-sensitive), and -n (numeric). It then uses grep to search for the pattern in the input file, either case-insensitively (-i) or case-sensitively (-c), and either matching (-n) or not matching (-N) the pattern. The results are printed to the output file.

```
while getopts i:o:p:cn optletter
do case $optletter in
    i) inp=${OPTARG};;
    o) outp=${OPTARG};;
    p) templ=${OPTARG};;
    c) reg=true;;
    n) num=true;;
esac
done

if $reg
then if $num
    then grep -i -n $templ $inp > $outp
    else grep -i $templ $inp > $outp
    fi
else if $num
    then grep -n $templ $inp > $outp
    else grep $templ $templ $inp > $outp
    fi
fi
```

Рис. 1: Рис.1

Код файла задания 2.

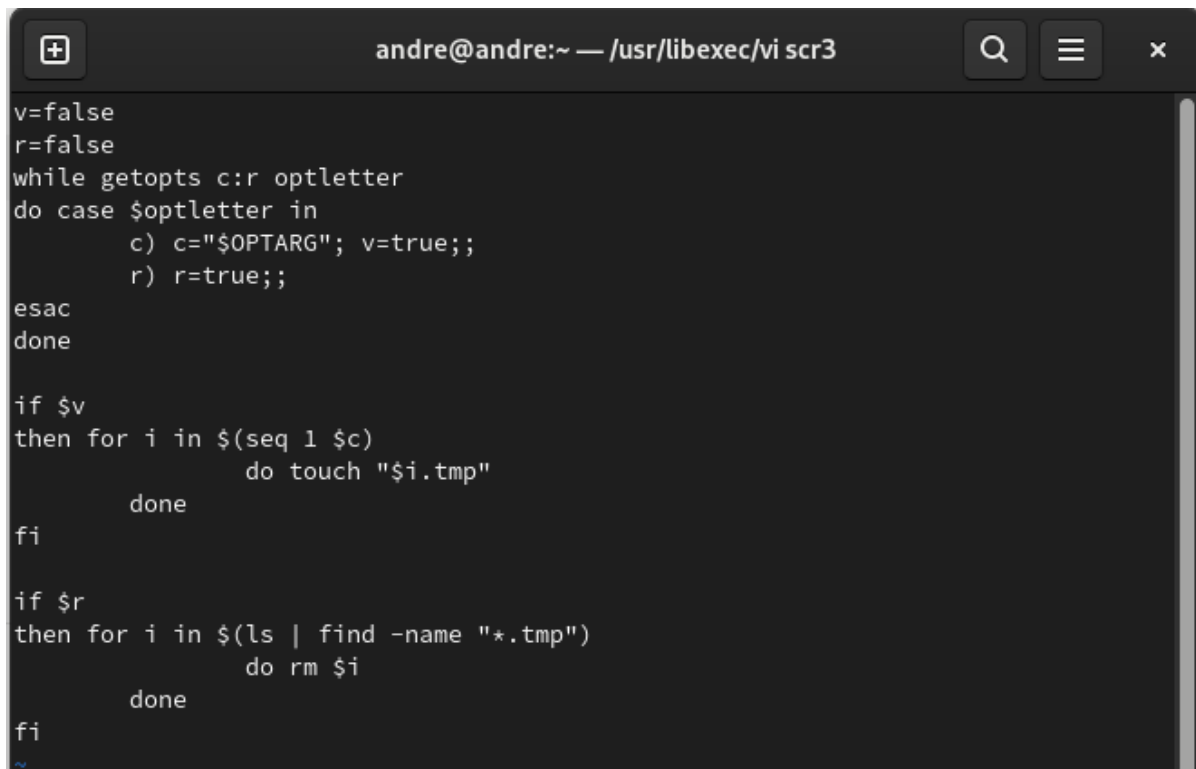


The image shows a terminal window with a dark background. The title bar at the top reads "andre@andre:~ — /usr/libexec/vi scr2". The terminal content shows the following commands and output:

```
g++ -o c-scr2 c-scr2.cpp
./c-scr2 $1
arg=$?
if [ "$arg" == "1" ]
    then echo "arg > 0"
fi
if [ "$arg" == "2" ]
    then echo "arg < 0"
fi
if [ "$arg" == "3" ]
    then echo "arg = 0"
fi
```

Рис. 2: Рис.3

Код задания 3.



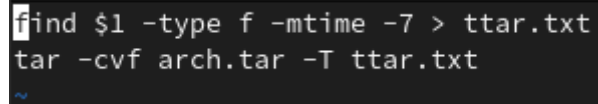
```
andre@andre:~ — /usr/libexec/vi scr3
v=false
r=false
while getopts c:r optletter
do case $optletter in
    c) c="$OPTARG"; v=true;;
    r) r=true;;
esac
done

if $v
then for i in $(seq 1 $c)
    do touch "$i.tmp"
    done
fi

if $r
then for i in $(ls | find -name "*.tmp")
    do rm $i
    done
fi
~
```

Рис. 3: Рис.6

Код задания 4



```
find $1 -type f -mtime -7 > ttar.txt
tar -cvf arch.tar -T ttar.txt
~
```

Рис. 4: Рис.8

Выводы

Благодаря данной работе мы научились работать с vi.