

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 12

дисциплина: Операционные системы

Студент: Хосе Фернадо Леон Атупанья

Группа: НПМбд-02-20

МОСКВА 2021 г.

Цель работы\

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

Задания

11. 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 о коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.

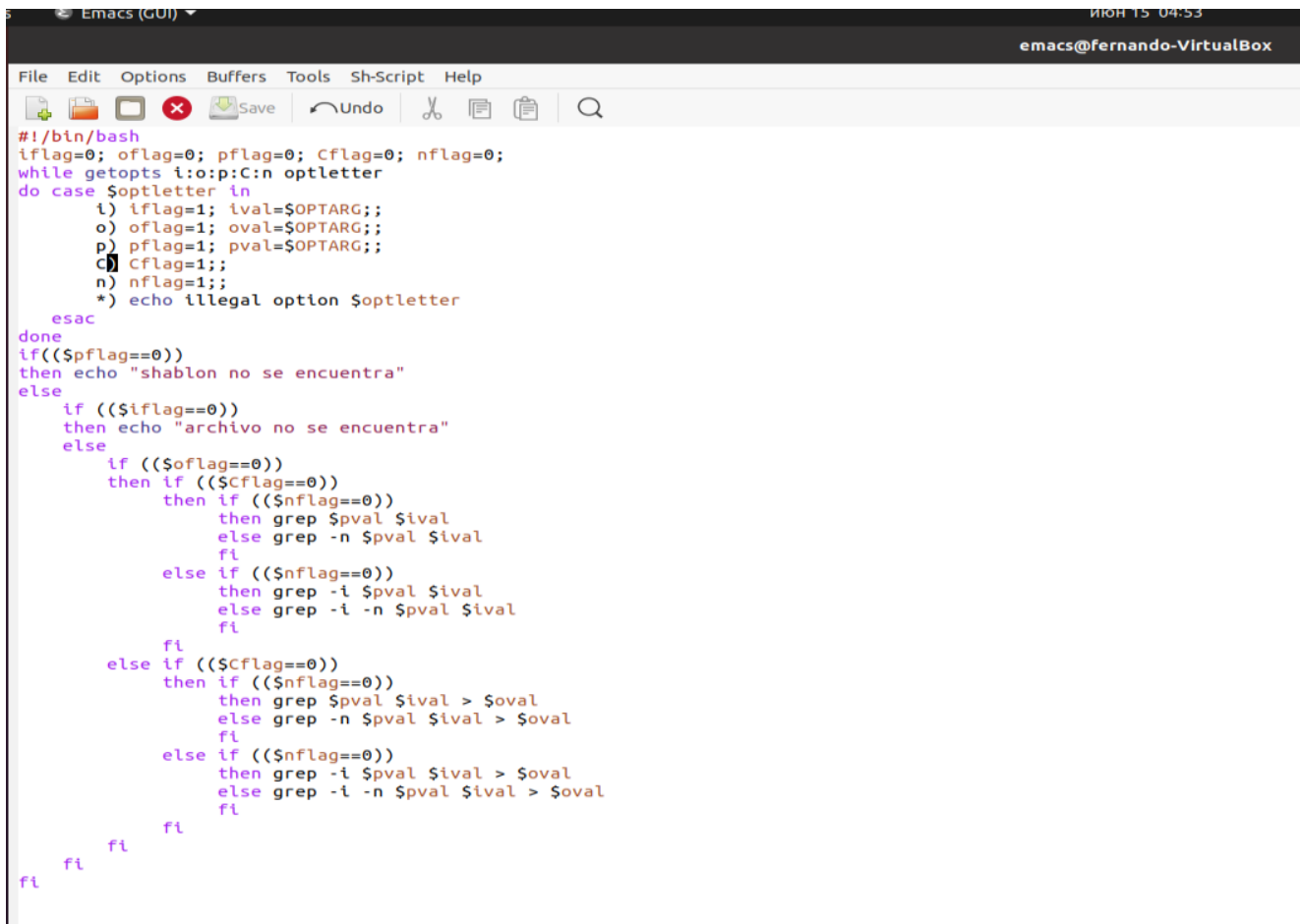
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N (например 1.tmp, 2.tmp, 3.tmp, 4.tmp и т.д.). Число файлов, которые необходимо создать, передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы (если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

Выполнение лабораторной работы

1. Используя команды getoptс grep, написал 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. Для данной задачи я создал файл prog1.sh и написал соответствующие скрипты (рис. -@fig:001).



```
#!/bin/bash
iflag=0; oflag=0; pflag=0; Cflag=0; nflag=0;
while getopts i:o:p:C:n 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 $optletter
    esac
done
if (($pflag==0))
then echo "shablon no se encuentra"
else
if (($iflag==0))
then echo "archivo no se encuentra"
else
if (($oflag==0))
then if (($Cflag==0))
then if (($nflag==0))
then grep $pval $ival
else grep -n $pval $ival
fi
else if (($nflag==0))
then grep -i $pval $ival
else grep -i -n $pval $ival
fi
fi
else if (($Cflag==0))
then if (($nflag==0))
then grep $pval $ival > $oval
else grep -n $pval $ival > $oval
fi
else if (($nflag==0))
then grep -i $pval $ival > $oval
else grep -i -n $pval $ival > $oval
fi
fi
fi
fi
```

{ #fig:001 width=70% }

Далее я проверил работу написанного скрипта, используя различные опции (например, команда «./prog.sh -i ~/a.txt -o ~/b.txt -p song -C -n»), предварительно добавив право на исполнение файла (команда «chmod +x *.sh») и создав 2 файла, которые необходимы для выполнения программы: a.txt и b.txt (рис. -@fig:002). Скрипт работает корректно.

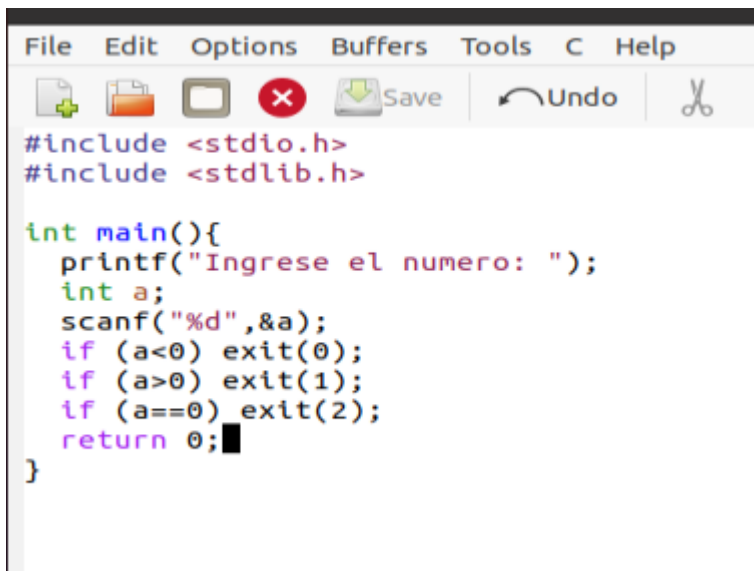
```
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ chmod +x *.sh
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ cat ~/a.txt
Alfa left and never came back. He is from Rusia
If he says he is from Spain, it is a lie
Baby you are my kaif
The ice melts between us, because we are from Colombia
Bokrug sveta is the best flirt to eat, for people coming from Germany
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog1.sh -i ~/a.txt -o ~/b.txt -p from -C -n
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ cat ~/b.txt
Alfa left and never came back. He is from Rusia
If he says he is from Spain, it is a lie
The ice melts between us, because we are from Colombia
Bokrug sveta is the best flirt to eat, for people coming from Germany
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog1.sh -i ~/a.txt -o ~/b.txt -p from -n
bash: ./prog1.sh: No such file or directory
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog1.sh -i ~/a.txt -o ~/b.txt -p from -n
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ cat ~/b.txt
1:Alfa left and never came back. He is from Rusia
2:If he says he is from Spain, it is a lie
4:The ice melts between us, because we are from Colombia
5:Bokrug sveta is the best flirt to eat, for people coming from Germany
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog1.sh -i ~/a.txt -C -n
bash: ./prog1.sh: No such file or directory
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog1.sh -i ~/a.txt -C -n
shablon no se encuentra
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog1.sh -o ~/b.txt -p from -n
archivo no se encuentra
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$
```

{ #fig:002 width=70% }

2. Написал 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. Для данной задачи я создал 2 файла: prog2.c и prog2.sh (рис. -@fig 003) и написал соответствующие скрипты (рис. -@fig:004) (рис. -@fig:005).

```
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ touch prog2.c prog2.sh
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ emacs &
[2] 5892
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ emacs &
[3] 5917
```

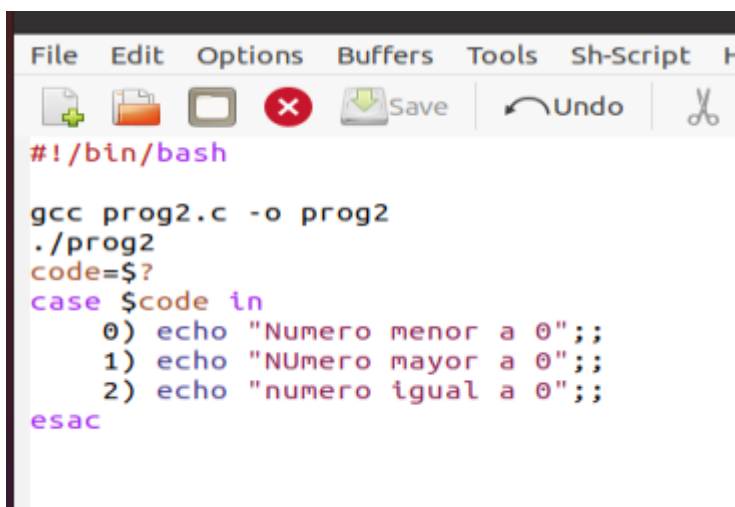
{ #fig:003 width=70% }

A screenshot of a text editor window showing a C program. The menu bar includes File, Edit, Options, Buffers, Tools, C, and Help. The toolbar has icons for file operations and editing. The code is as follows:

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

int main(){
    printf("Ingrese el numero: ");
    int a;
    scanf("%d",&a);
    if (a<0) exit(0);
    if (a>0) exit(1);
    if (a==0) exit(2);
    return 0;
}
```

{ #fig:004 width=70% }

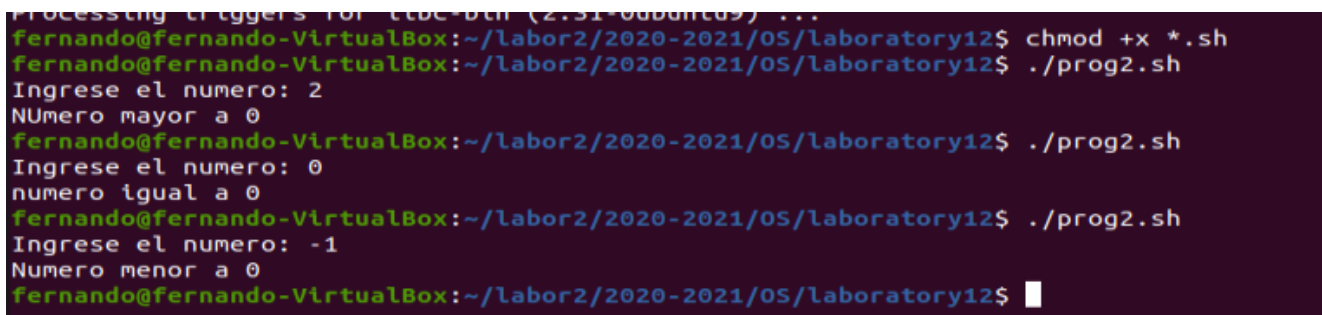
A screenshot of a text editor window showing a shell script. The menu bar includes File, Edit, Options, Buffers, Tools, Sh-Script, and H. The toolbar has icons for file operations and editing. The script is as follows:

```
#!/bin/bash

gcc prog2.c -o prog2
./prog2
code=$?
case $code in
    0) echo "Numero menor a 0";;
    1) echo "Numero mayor a 0";;
    2) echo "numero igual a 0";;
esac
```

{ #fig:005 width=70% }

Затем я проверил работу написанных скриптов (команда ". /prog2.sh") предварительно добавив право выполнения файла (команда "chmod + x *. sh") (рис. - @рис.006). Скрипты работают правильно.

A screenshot of a terminal window showing the execution of the shell script. The prompt is fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12\$. The commands and outputs are as follows:

```
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ chmod +x *.sh
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog2.sh
Ingrese el numero: 2
Numero mayor a 0
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog2.sh
Ingrese el numero: 0
numero igual a 0
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$ ./prog2.sh
Ingrese el numero: -1
Numero menor a 0
fernando@fernando-VirtualBox:~/labor2/2020-2021/05/laboratory12$
```

{ #fig:006 width=70% }

3. Написал 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). Для данной задачи я создал файл: prog3.sh и написал соответствующий скрипт (рис. -@fig:007).

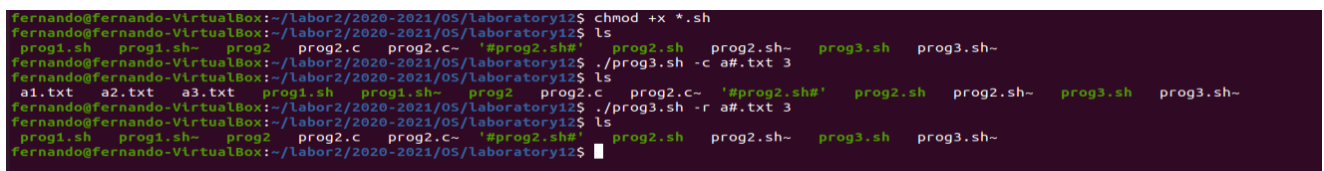


```
File Edit Options Buffers Tools Sh-Script Help
+ Save Undo
#!/bin/bash

opt=$1;
form=$2;
num=$3;
function Files(){
    for ((i=1; i<=$num; i++)) do
        file=$(echo $form | tr '#' "$i")
        if [ $opt == "-r" ]
        then
            rm -f $file
        elif [ $opt == "-c" ]
        then
            touch $file
        fi
    done
}
Files
```

{ #fig:007 width=70% }

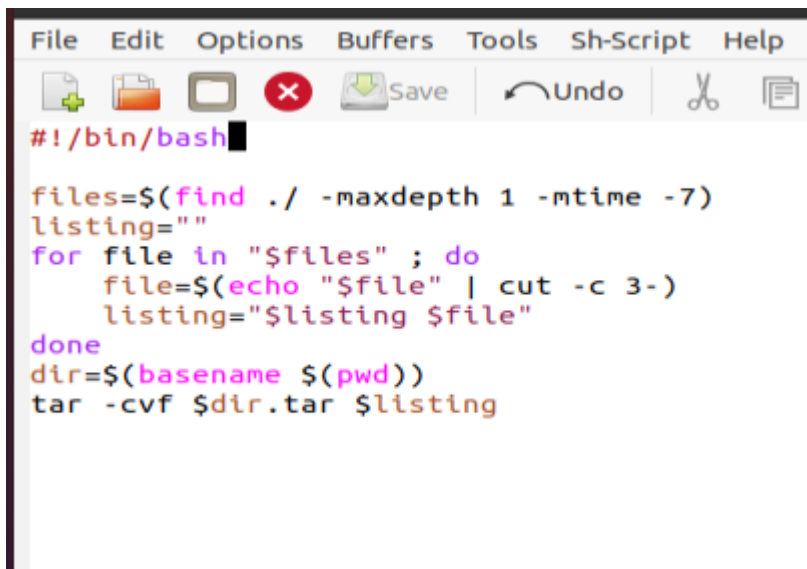
Далее я проверил работу написанного скрипта (команда «./prog3.sh»), предварительно добавив право на исполнение файла (команда «chmod +x *.sh»). Сначала я создал три файла (команда «./prog3.sh -c a#.txt 3»), удовлетворяющие условию задачи, а потом удалил их (команда «./prog3.sh -r a#.txt 3») (рис. -@fig:008). Скрипт работает корректно.



```
fernando@fernando-VirtualBox:~/Labor2/2020-2021/05/Laboratory12$ chmod +x *.sh
fernando@fernando-VirtualBox:~/Labor2/2020-2021/05/Laboratory12$ ls
prog1.sh prog1.sh~ prog2 prog2.c prog2.c~ '#prog2.sh#' prog2.sh prog2.sh~ prog3.sh prog3.sh~
fernando@fernando-VirtualBox:~/Labor2/2020-2021/05/Laboratory12$ ./prog3.sh -c a#.txt 3
fernando@fernando-VirtualBox:~/Labor2/2020-2021/05/Laboratory12$ ls
a1.txt a2.txt a3.txt prog1.sh prog1.sh~ prog2 prog2.c prog2.c~ '#prog2.sh#' prog2.sh prog2.sh~ prog3.sh prog3.sh~
fernando@fernando-VirtualBox:~/Labor2/2020-2021/05/Laboratory12$ ./prog3.sh -r a#.txt 3
fernando@fernando-VirtualBox:~/Labor2/2020-2021/05/Laboratory12$ ls
prog1.sh prog1.sh~ prog2 prog2.c prog2.c~ '#prog2.sh#' prog2.sh prog2.sh~ prog3.sh prog3.sh~
fernando@fernando-VirtualBox:~/Labor2/2020-2021/05/Laboratory12$
```

{ #fig:008 width=70% }

4. Написал командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории. Модифицировала его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду find). Для данной задачи я создал файл: prog4.sh и написал соответствующий скрипт (рис. -@fig:009).

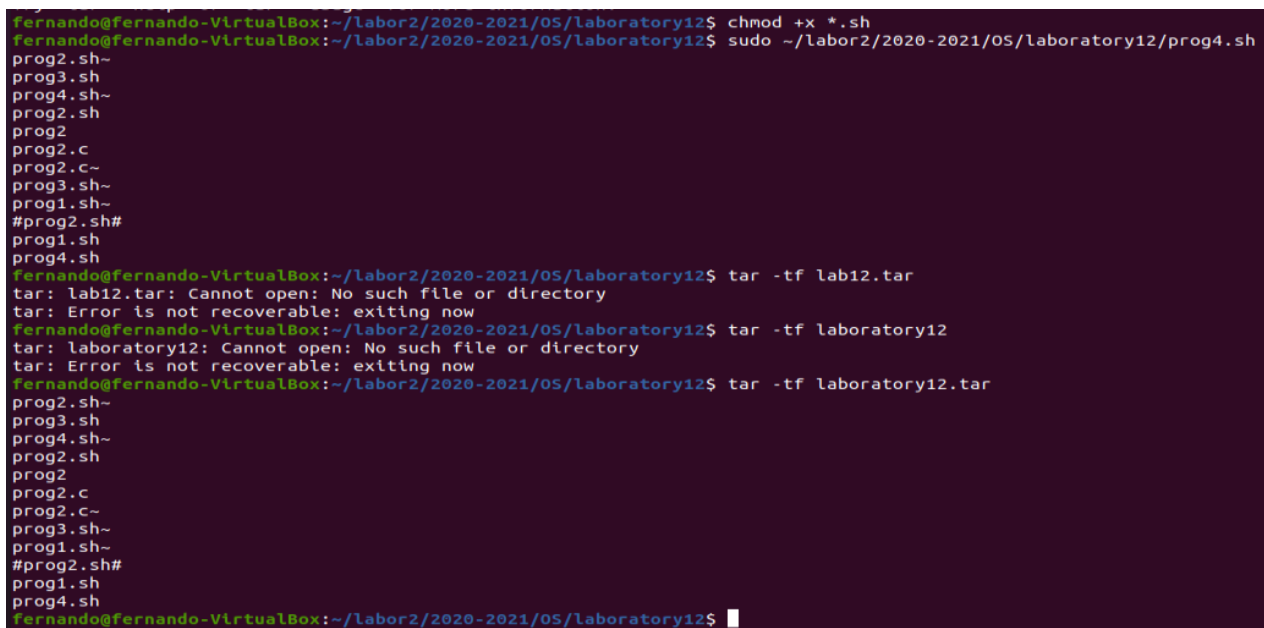


```
File Edit Options Buffers Tools Sh-Script Help
+ Save Undo
#!/bin/bash

files=$(find ./ -maxdepth 1 -mtime -7)
listing=""
for file in "$files" ; do
    file=$(echo "$file" | cut -c 3-)
    listing="$listing $file"
done
dir=$(basename $(pwd))
tar -cvf $dir.tar $listing
```

{ #fig:009 width=70% }

Далее я проверил работу написанного скрипта (команды «`sudo ~/labor2/2020-2021/OS/laboratory12/prog4.sh`» и «`tar -tf lab12.tar`»), предварительно добавив право на исполнение файла (команда «`chmod +x *.sh`») (рис. -@fig:010) (рис. -@fig:011). Скрипт работает корректно.



```
fernando@fernando-VirtualBox:~/labor2/2020-2021/OS/laboratory12$ chmod +x *.sh
fernando@fernando-VirtualBox:~/labor2/2020-2021/OS/laboratory12$ sudo ~/labor2/2020-2021/OS/laboratory12/prog4.sh
prog2.sh~
prog3.sh~
prog4.sh~
prog2.sh~
prog2
prog2.c~
prog2.c~
prog3.sh~
prog1.sh~
#prog2.sh#
prog1.sh~
prog4.sh~
fernando@fernando-VirtualBox:~/labor2/2020-2021/OS/laboratory12$ tar -tf lab12.tar
tar: lab12.tar: Cannot open: No such file or directory
tar: Error is not recoverable: exiting now
fernando@fernando-VirtualBox:~/labor2/2020-2021/OS/laboratory12$ tar -tf laboratory12
tar: laboratory12: Cannot open: No such file or directory
tar: Error is not recoverable: exiting now
fernando@fernando-VirtualBox:~/labor2/2020-2021/OS/laboratory12$ tar -tf laboratory12.tar
prog2.sh~
prog3.sh~
prog4.sh~
prog2.sh~
prog2
prog2.c~
prog2.c~
prog3.sh~
prog1.sh~
#prog2.sh#
prog1.sh~
prog4.sh~
fernando@fernando-VirtualBox:~/labor2/2020-2021/OS/laboratory12$
```

{ #fig:010 width=70% }

Контрольные вопросы

1. Команда `getopts` осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных. Синтаксис команды следующий:
`getopts option-string variable [arg ...]`
Флаги – это опции командной строки, обычно помеченные знаком минус;
Например, для команды `ls` флагом может являться `-F`.
Строка опций `option-string` – это список возможных букв и чисел

соответствующего флага. Если ожидается, что некоторый флаг будет сопровождаться некоторым аргументом, то за символом, обозначающим этот флаг, должно следовать двоеточие. Соответствующей переменной присваивается буква данной опции. Если команда `getopts` может распознать аргумент, то она возвращает истину. Принято включать `getopts` в цикл `while` и анализировать введенные данные с помощью оператора `case`.

Функция `getopts` включает две специальные переменные среды – `OPTARG` и `OPTIND`. Если ожидается дополнительное значение, то `OPTARG` устанавливается в значение этого аргумента.

Функция `getopts` также понимает переменные типа массив, следовательно, можно использовать её в функции не только для синтаксического анализа аргументов функций, но и для анализа введенных пользователем данных.

2. При перечислении имён файлов текущего каталога можно использовать следующие символы:

- `*` – соответствует произвольной, в том числе и пустой строке;
- `?` – соответствует любому одинарному символу;
- `[c1-c2]` – соответствует любому символу, лексикографически находящемуся между символами `c1` и `c2`. Например,
- `echo *` – выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды `ls`;
- `ls *.c` – выведет все файлы с последними двумя символами, совпадающими с `.c`.
- `echo prog.?` – выведет все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются `prog..`
- `[a-z]*` – соответствует произвольному имени файла в текущем каталоге, начинающемуся с любой строчной буквы латинского алфавита.

3. Часто бывает необходимо обеспечить проведение каких-либо действий циклически и управление дальнейшими действиями в зависимости отрезультатов проверки некоторого условия. Для решения подобных задач язык программирования `bash` предоставляет возможность использовать такие управляющие конструкции, как `for`, `case`, `if` и `while`. С точки зрения командного процессора эти управляющие конструкции являются обычными командами и могут использоваться как при создании командных файлов, так и при работе в интерактивном режиме. Команды, реализующие подобные конструкции, по сути, являются операторами языка программирования `bash`. Поэтому при описании языка программирования `bash` термин оператор будет использоваться наравне с термином команда.

Команды ОС UNIX возвращают код завершения, значение которого может быть использовано для принятия решения о дальнейших действиях.

Команда `test`, например, создана специально для использования в командных файлах. Единственная функция этой команды заключается в выработке кода завершения.

4. Два несложных способа позволяют вам прерывать циклы в оболочке `bash`. Команда `break` завершает выполнение цикла, а команда `continue` завершает данную итерацию блока операторов. Команда `break` полезна для завершения цикла `while` в ситуациях, когда условие перестаёт быть правильным. Команда `continue` используется в ситуациях, когда больше нет необходимости выполнять блок операторов, но вы можете захотеть продолжить проверять данный блок на других условных выражениях.
5. Следующие две команды ОС UNIX используются только совместно с управляющими конструкциями языка программирования `bash`: это команда `true`, которая всегда возвращает код завершения, равный нулю (т.е. истина), и команда `false`, которая всегда возвращает код завершения, не равный нулю (т.е. ложь).
Примеры бесконечных циклов:
- ```
while true
do echo hello andy
done
until false
do echo hello mike
done
```
6. Строка `if test -f mans/i.s` проверяет, существует ли файл *mans/i.s* и является ли этот файл обычным файлом. Если данный файл является каталогом, то команда вернет нулевое значение (ложь).
7. Выполнение оператора цикла `while` сводится к тому, что сначала выполняется последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `while`, а затем, если последняя выполненная команда из этой последовательности команд возвращает нулевой код завершения (истина), выполняется последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `do`, после чего осуществляется безусловный переход на начало оператора цикла `while`. Выход из цикла будет осуществлён тогда, когда последняя выполненная команда из последовательности команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `while`, возвратит ненулевой код завершения (ложь).  
При замене в операторе цикла `while` служебного слова `while` на `until` условие, при выполнении которого осуществляется выход из цикла, меняется на противоположное. В остальном оператор цикла `while` и оператор цикла `until` идентичны.

## Выводы



Выполняя эту лабораторную работу, я изучил основы программирования в оболочке операционной системы UNIX и научился писать более сложные пакетные файлы с использованием логических конструкций управления и циклов.