

отчёта по лабораторной работе 12

Содержание

0.1 Цель работы	4
0.2 Выполнение работы.	4
0.3 Вывод:	10
0.4 Ответы на контрольные вопросы:	10

List of Tables

List of Figures

0.1 Цель работы

изучить основы программирования в оболочке ОС UNIX, научиться писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

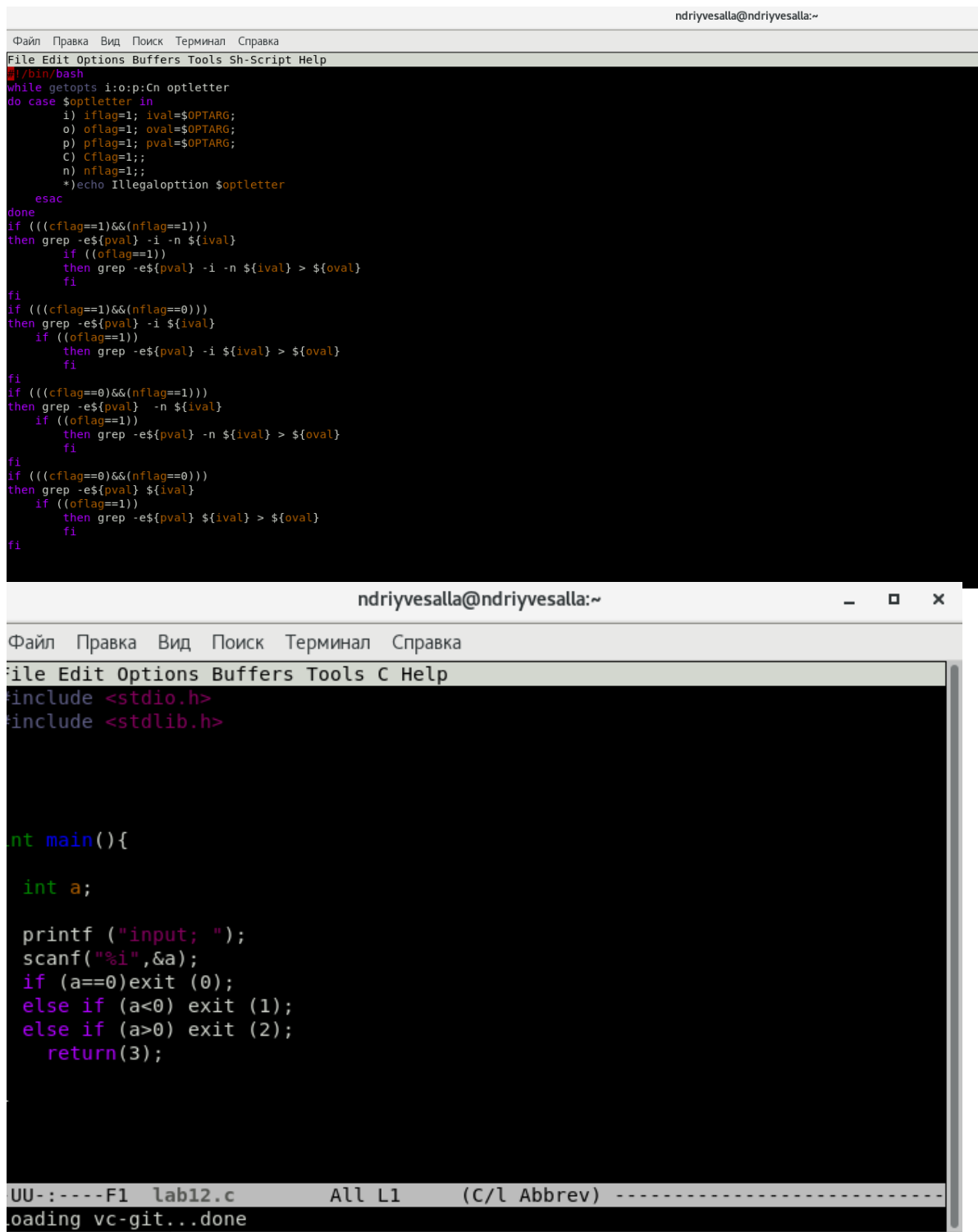
0.2 Выполнение работы.

Ход работы:

1.Используя команды `grep`, написала 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`.

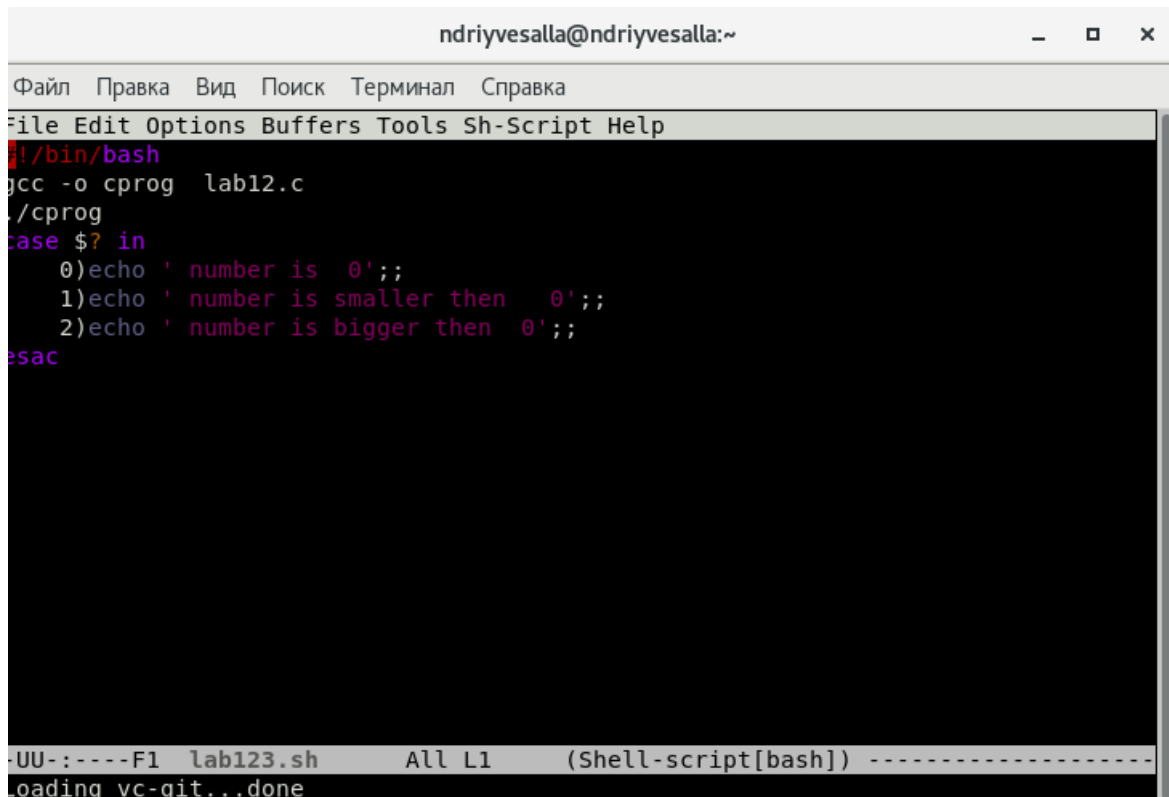


The image consists of two screenshots of a terminal window. The top screenshot shows a shell script for validating command-line options. The script uses a while loop to process options until no more are left. It checks for flags like 'i', 'o', 'p', 'C', and 'n', each with an optional argument. It uses grep to check if the provided values are integers and if they are greater than the optional values. The bottom screenshot shows a C program named lab12.c. It includes stdio.h and stdlib.h. The main function prompts the user for input, reads an integer, and then uses exit(n) to terminate the program based on the value: 0 for non-negative, 1 for negative, and 2 for positive. It returns 3 if it reaches the end of the function.

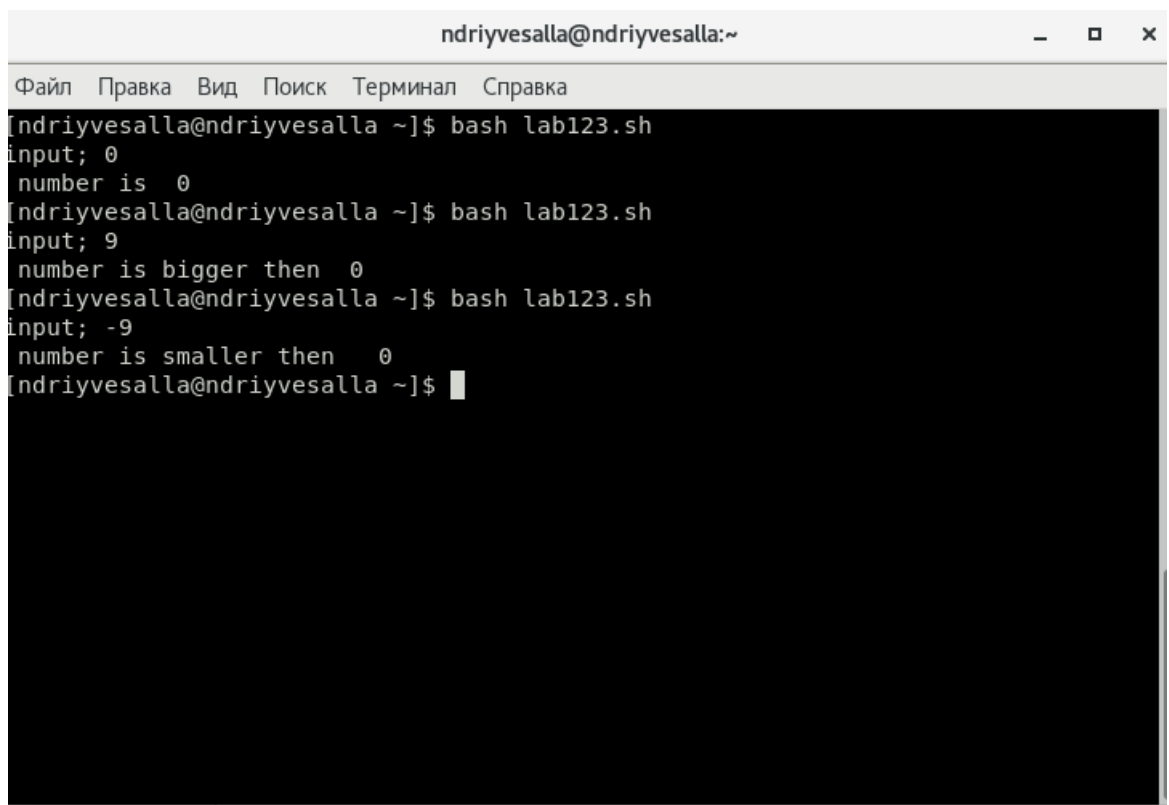
```
ndriyvesalla@ndriyvesalla:~  
Файл Правка Вид Поиск Терминал Справка  
File Edit Options Buffers Tools Sh-Script Help  
#!/bin/bash  
while getopts i:op:Cn optletter  
do case $optletter in  
    i) iflag=1; ival=$OPTARG;  
    o) oflag=1; oval=$OPTARG;  
    p) pflag=1; pval=$OPTARG;  
    C) cflag=1;;  
    n) nflag=1;;  
    *)echo Illegaloption $optletter  
    esac  
done  
if (((cflag==1)&&(nflag==1)))  
then grep -e${pval} -i -n ${ival}  
    if ((oflag==1))  
    then grep -e${pval} -i -n ${ival} > ${oval}  
    fi  
fi  
if (((cflag==1)&&(nflag==0)))  
then grep -e${pval} -i ${ival}  
    if ((oflag==1))  
    then grep -e${pval} -i ${ival} > ${oval}  
    fi  
fi  
if (((cflag==0)&&(nflag==1)))  
then grep -e${pval} -n ${ival}  
    if ((oflag==1))  
    then grep -e${pval} -n ${ival} > ${oval}  
    fi  
fi  
if (((cflag==0)&&(nflag==0)))  
then grep -e${pval} ${ival}  
    if ((oflag==1))  
    then grep -e${pval} ${ival} > ${oval}  
    fi  
fi  
fi  
ndriyvesalla@ndriyvesalla:~  
Файл Правка Вид Поиск Терминал Справка  
File Edit Options Buffers Tools C Help  
#include <stdio.h>  
#include <stdlib.h>  
  
int main(){  
  
    int a;  
  
    printf ("input; ");  
    scanf ("%i",&a);  
    if (a==0)exit (0);  
    else if (a<0) exit (1);  
    else if (a>0) exit (2);  
    return(3);  
  
UU-:----F1 lab12.c All L1 (C/l Abbrev) -----  
loading vc-git...done
```

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.



```
ndriyvesalla@ndriyvesalla:~  
Файл Правка Вид Поиск Терминал Справка  
File Edit Options Buffers Tools Sh-Script Help  
#!/bin/bash  
gcc -o cprog lab12.c  
./cprog  
case $? in  
  0)echo ' number is 0';;  
  1)echo ' number is smaller then 0';;  
  2)echo ' number is bigger then 0';;  
esac  
UU-:----F1 lab123.sh All L1 (Shell-script[bash]) -----  
Loading vc-git...done
```

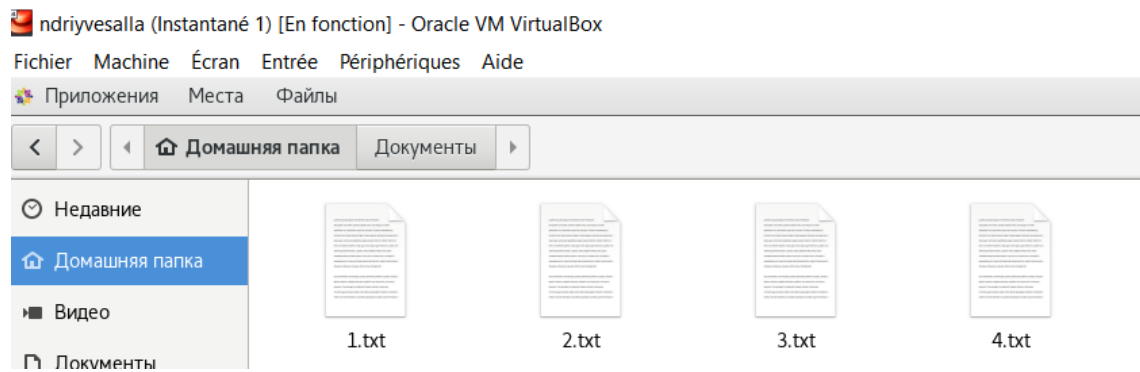


```
ndriyvesalla@ndriyvesalla:~  
Файл Правка Вид Поиск Терминал Справка  
[ndriyvesalla@ndriyvesalla ~]$ bash lab123.sh  
input; 0  
number is 0  
[ndriyvesalla@ndriyvesalla ~]$ bash lab123.sh  
input; 9  
number is bigger then 0  
[ndriyvesalla@ndriyvesalla ~]$ bash lab123.sh  
input; -9  
number is smaller then 0  
[ndriyvesalla@ndriyvesalla ~]$
```

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).

```
ndriyvesalla@ndriyvesalla:~
Файл Правка Вид Поиск Терминал Справка
Изображения 0 Шаблоны
[ndriyvesalla@ndriyvesalla ~]$ ls -l
итого 27900
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla      0 май 28 22:23 1.txt
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla      0 май 28 22:23 2.txt
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla      0 май 28 22:23 3.txt
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla      0 май 28 22:23 4.txt
drwxrwxr-x. 2 ndriyvesalla ndriyvesalla    26 май 25 22:04 backup
-rwxrwxr-x. 1 ndriyvesalla ndriyvesalla  8464 май 28 22:03 cprog
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla      0 май 28 20:22 lab121
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    220 май 28 20:54 lab121.cpp
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla      0 май 28 20:25 lab121.txt
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    173 май 28 21:29 lab123.sh
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    175 май 28 21:21 lab123.sh~
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    341 май 28 22:23 lab124.sh
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    341 май 28 22:19 lab124.sh~
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    207 май 28 22:04 #lab12.c#
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    197 май 28 21:27 lab12.c
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    196 май 28 20:37 lab12.c~
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    238 май 28 21:04 lab12.cpp~
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    795 май 28 20:22 lab12.sh
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla    792 май 28 20:20 lab12.sh~
-rw-rw-r--. 1 ndriyvesalla ndriyvesalla      0 май 28 20:23 lab12.txt
drwxr-xr-x. 2 ndriyvesalla ndriyvesalla     77 май 28 12:07 nos
```

```
Файл Правка Вид Поиск Терминал Справка
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
let dflag=0;
while getopts a:d optletter
do case $optletter in
    a) aflag=1; aval=$OPTARG;;
    b) dflag=1;;
    *)echo illegaloption $optletter
    esac
done
#echo ${aval}
if ((dflag==0))
then for ((i=1; i<=aval;i++))
do touch ${i}.txt
done
fi
if ((dflag==1))
then for ((i=1;i<=aval;i++))
do rm ${i}.txt
done
fi
```



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 (использовала команду `find`).


```
ndriyvesalla@ndriyvesalla:~
Файл Правка Вид Поиск Терминал Справка
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash

tar -cf 8.tar $@
tar -cf 8l.tar
find $@ -mtime -7 -exec tar -rf 8l.tar '{}' ';'

UU-:----F1 lab125.sh All L1 (Shell-script[bash]) -----
loading vc-git...done
```

```
ndriyvesalla@ndriyvesalla:~
Файл Правка Вид Поиск Терминал Справка
tar: Удаляется начальный '/' из имен объектов
tar: /lab12.txt: Функция stat завершилась с ошибкой: Нет такого файла или каталога
tar: Завершение работы с состоянием неисправности с из-за возникших ошибок
tar: Робкий отказ от создания пустого архива
Попробуйте 'tar --help' или 'tar --usage' для
получения дополнительных сведений.
find: '/lab12.txt': Нет такого файла или каталога
[ndriyvesalla@ndriyvesalla ~]$ ls
1.txt 4.txt backup lab121.cpp lab123.sh~ lab125.sh lab12.c~ lab12.sh~
pandoc-2.5 script1.sh~ script3.sh script.sh~ Документы Музыка Шаблоны
2.txt 8.tar cprog lab121.txt lab124.sh lab125.sh~ lab12.cpp~ lab12.txt
pandoc-2.5-linux.tar.gz script2.sh script4.sh work Загрузки 06
щедоступные
3.txt 9l.tar lab121 lab123.sh lab124.sh~ lab12.c lab12.sh nos
script1.sh script2.sh~ script4.sh~ Видео Изображения Рабочий стол
[ndriyvesalla@ndriyvesalla ~]$
```

Вывод: изучил основы программирования в оболочке ОС UNIX, научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

Ответы на контрольные вопросы:

1. Команда `getopts` является встроенной командой командной оболочки `bash`, предназначенной для разбора параметров сценариев. Она обрабатывает исключительно однобуквенные параметры как с аргументами, так и без них и этого вполне достаточно для передачи сценариям любых входных данных.

2. При генерации имен используют метасимволы:

* произвольная (возможно пустая) последовательность символов;

? один произвольный символ;

[...] любой из символов, указанных в скобках перечислением и/или с указанием диапазона;

`cat f*` выдаст все файлы каталога, начинающиеся с "f";

`cat *f*` выдаст все файлы, содержащие "f";

`cat program.?` выдаст файлы данного каталога с однобуквенными расширениями, скажем "program.c" и "program.o", но не выдаст "program.com";

`cat [a-d]*` выдаст файлы, которые начинаются с "a", "b", "c", "d". Аналогичный эффект дадут и команды "`cat [abcd]*`" и "`cat [bdac]*`".

3. Операторы `&&` и `||` являются управляющими операторами. Если в командной строке стоит `command1 && command2`, то `command2` выполняется в том, и только в том случае, если статус выхода из команды `command1` равен нулю, что говорит об успешном ее завершении. Аналогично, если командная строка имеет вид `command1 || command2`, то команда `command2` выполняется тогда, и только тогда, когда статус выхода из команды `command1` отличен от нуля.

4. Оператор break завершает выполнение ближайшего включающего цикла или условного оператора, в котором он отображается.

5. Команда true всегда возвращает ноль в качестве выходного статуса для индикации успеха. Команда false всегда возвращает не-ноль в качестве выходного статуса для индикации неудачи. Во всех управляющих конструкциях в качестве логического значения используется код возврата из программы, указанной в качестве условия. Код возврата 0 – истина, любое другое значение – ложь. Программа true – всегда завершается с кодом 0, false – всегда завершается с кодом 1.

6. Введенная строка означает условие существования файла man\$s/\$i.\$s

7. Цикл While выполняется до тех пор, пока указанное в нем условие истинно. Когда указанное условие становится ложным - цикл завершается. Цикл Until выполняется до тех пор, пока указанное в нем условие ложно.