

Лабораторная работа №12

Дисциплина: Операционные системы

Колчева Юлия Вячеславовна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выводы	14
4	Контрольные вопросы	15

List of Tables

List of Figures

2.1	Первая программа	7
2.2	Проверка работы1	8
2.3	Программа на си	9
2.4	Программа sh	10
2.5	Проверка работы	10
2.6	Программа 3	11
2.7	Проверка работы	12
2.8	Программа4	13
2.9	Последняя проверка	13

1 Цель работы

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

2 Выполнение лабораторной работы

Используя команды `getopts` `grep`, написала командный файл, который анализирует командную строку с ключами: `-i inputfile`—прочитать данные из указанного файла; `-o outputfile`—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`. Для данной задачи я создала файл `prog1.sh` и написала соответствующие скрипты.(рис. 2.1)

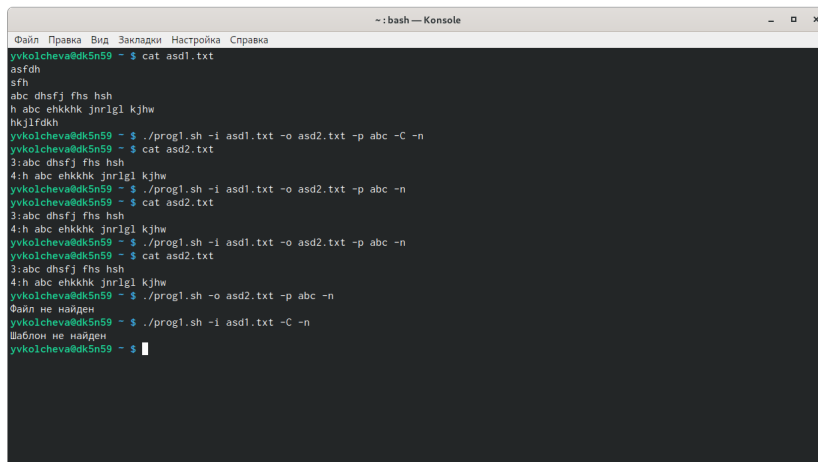
```

emacs@dk5n59
File Edit Options Buffers Tools Sh-Script Help

#!/bin/bash
iflag=0; oflag=0; pflag=0; Cflag=0; nflag=0;
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 $optletter
esac
done
if (($flag==0))
then echo "Шаблон не найден"
else
    if (($iflag==0))
    then echo "Файл не найден"
    else
        if (($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
fi

U:--- prog1.sh Top L40 (Shell-script[sh]) Br мая 25 11:56 2.54
cause of the error in your initialization file. Start Emacs with
the '--debug-init' option to view a complete error backtrace.
U:%%*- *Warnings* Bot L8 (Special) Br мая 25 11:56 2.54

```



```
~: bash — Konsole
Файл Правка Вид Закладки Настройка Справка
yukoicheva@dk5n59 ~$ cat asd1.txt
asfdh
sfh
abc dhsfj fhs hsh
h abc ehkkhk jnr1gl kjhw
hkjlfdkh
yukoicheva@dk5n59 ~$ ./prog1.sh -i asd1.txt -o asd2.txt -p abc -C -n
yukoicheva@dk5n59 ~$ cat asd2.txt
3:abc dhsfj fhs hsh
4:h abc ehkkhk jnr1gl kjhw
yukoicheva@dk5n59 ~$ ./prog1.sh -i asd1.txt -o asd2.txt -p abc -n
yukoicheva@dk5n59 ~$ cat asd2.txt
3:abc dhsfj fhs hsh
4:h abc ehkkhk jnr1gl kjhw
yukoicheva@dk5n59 ~$ ./prog1.sh -i asd1.txt -o asd2.txt -p abc -n
yukoicheva@dk5n59 ~$ cat asd2.txt
3:abc dhsfj fhs hsh
4:h abc ehkkhk jnr1gl kjhw
yukoicheva@dk5n59 ~$ ./prog1.sh -o asd2.txt -p abc -n
Файл не найден
yukoicheva@dk5n59 ~$ ./prog1.sh -i asd1.txt -C -n
Шаблон не найден
yukoicheva@dk5n59 ~$
```

Figure 2.2: Проверка работы1

Написала 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 создала 2 файла: `chislo.c` и `chislo.sh` и написала соответствующие скрипты. (рис. 2.3) (рис. 2.4)

The image shows a screenshot of an Emacs editor window. The title bar at the top reads "emacs@dk5n59". The menu bar includes "File", "Edit", "Options", "Buffers", "Tools", "C", and "Help". The main editing area contains the following C code:

```
#include <stdio.h>
#include <stdlib.h>
int main () {
    printf ("введите число\n");
    int a;
    scanf ("%d", &a);
    if (a<0) exit (0);
    if (a>0) exit (1);
    if (a==0) exit (2);
    return 0;
}
```

Below the code, the status bar shows "U:--- chislo.c All L11 (C/*l Abbrev) Вт мая 25 12:05 1.68". The output buffer at the bottom displays the following messages:

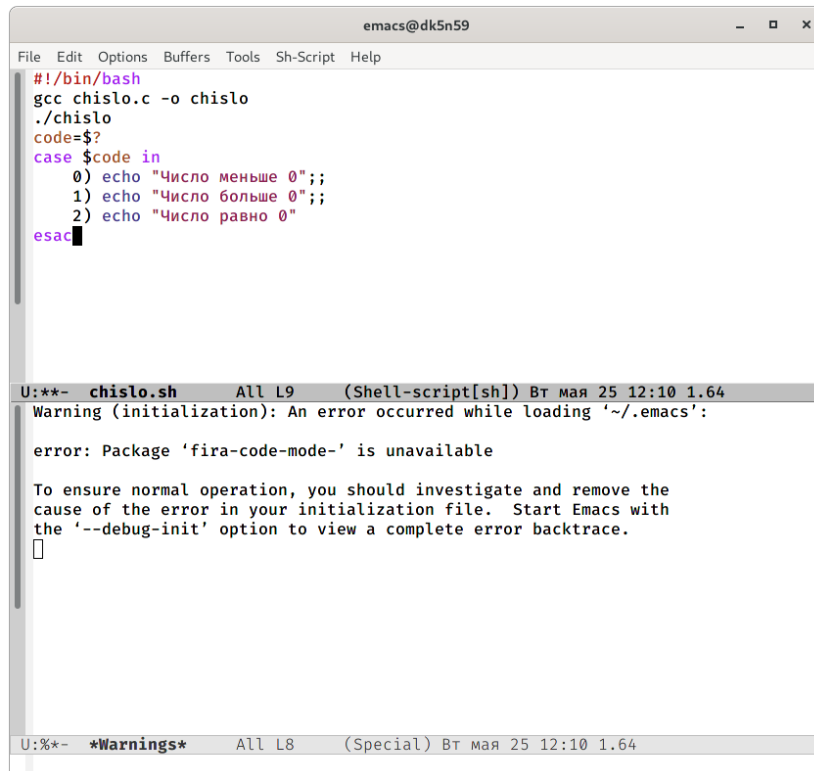
```
Warning (initialization): An error occurred while loading '~/.emacs':

error: Package 'fira-code-mode-' is unavailable

To ensure normal operation, you should investigate and remove the
cause of the error in your initialization file. Start Emacs with
the '--debug-init' option to view a complete error backtrace.
□
```

The status bar at the very bottom shows "U:%*- *Warnings* All L8 (Special) Вт мая 25 12:05 1.68" and "Wrote /afs/.dk.sci.pfu.edu.ru/home/y/v/yvkolcheva/chislo.c".

Figure 2.3: Программа на си



```
emacs@dk5n59
File Edit Options Buffers Tools Sh-Script Help
#!/bin/bash
gcc chislo.c -o chislo
./chislo
code=$?
case $code in
  0) echo "Число меньше 0";;
  1) echo "Число больше 0";;
  2) echo "Число равно 0";;
esac

U:***- chislo.sh All L9 (Shell-script[sh]) Вт мая 25 12:10 1.64
Warning (initialization): An error occurred while loading '~/.emacs':

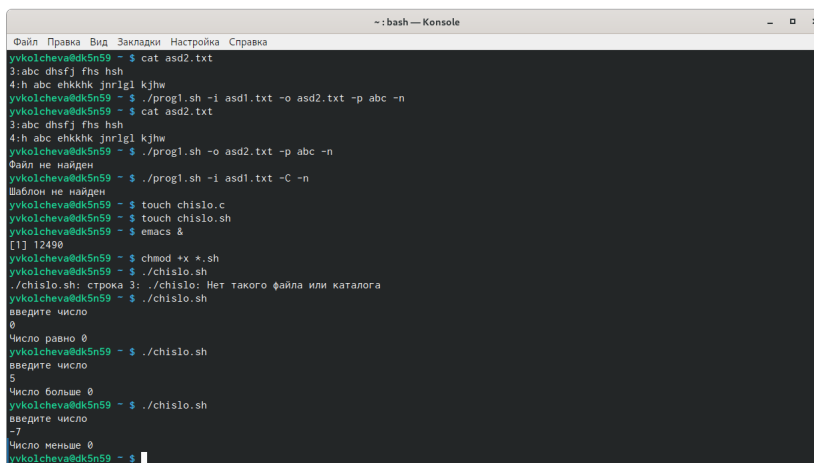
error: Package 'fira-code-mode-' is unavailable

To ensure normal operation, you should investigate and remove the
cause of the error in your initialization file. Start Emacs with
the '--debug-init' option to view a complete error backtrace.

U:%*- *Warnings* All L8 (Special) Вт мая 25 12:10 1.64
```

Figure 2.4: Программа sh

Далее я проверила работу написанных скриптов (./chislo.sh) (рис. 2.5)



```
-: bash — Konsole
Файл Правка Вид Закладки Настройка Справка
yukoicheva@dk5n59 ~$ cat asd2.txt
3:abc dhsfj fhs hsh
4:h abc ehkkhk jnrigl kjhw
yukoicheva@dk5n59 ~$ ./prog1.sh -i asd1.txt -o asd2.txt -p abc -n
yukoicheva@dk5n59 ~$ cat asd2.txt
3:abc dhsfj fhs hsh
4:h abc ehkkhk jnrigl kjhw
yukoicheva@dk5n59 ~$ ./prog1.sh -o asd2.txt -p abc -n
Файл не найден
yukoicheva@dk5n59 ~$ ./prog1.sh -i asd1.txt -C -n
Шаблон не найден
yukoicheva@dk5n59 ~$ touch chislo.c
yukoicheva@dk5n59 ~$ touch chislo.sh
yukoicheva@dk5n59 ~$ emacs &
[1] 12490
yukoicheva@dk5n59 ~$ chmod +x *.sh
yukoicheva@dk5n59 ~$ ./chislo.sh
./chislo.sh: строка 3: ./chislo: Нет такого файла или каталога
yukoicheva@dk5n59 ~$ ./chislo.sh
введите число
0
Число равно 0
yukoicheva@dk5n59 ~$ ./chislo.sh
введите число
5
Число больше 0
yukoicheva@dk5n59 ~$ ./chislo.sh
введите число
-7
Число меньше 0
yukoicheva@dk5n59 ~$
```

Figure 2.5: Проверка работы

Написала командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N. Число файлов, которые необходимо создать,

передаётся в аргументы командной строки. Этот же командный файл должен уметь удалять все созданные им файлы. Для данной задачи я создала файл: file.sh и написала соответствующий скрипт. (рис. 2.6)

```

emacs@dk5n59
File Edit Options Buffers Tools Sh-Script Help

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

U:--- file.sh All L10 (Shell-script[sh]) Вт мая 25 12:32 2.99
Warning (initialization): An error occurred while loading '~/.emacs':
error: Package 'fira-code-mode-' is unavailable
To ensure normal operation, you should investigate and remove the
cause of the error in your initialization file. Start Emacs with
the '--debug-init' option to view a complete error backtrace.
U:%*- *Warnings* All L8 (Special) Вт мая 25 12:32 2.99
Wrote /afs/.dk.sci.pfu.edu.ru/home/y/v/yvkolcheva/file.sh

```

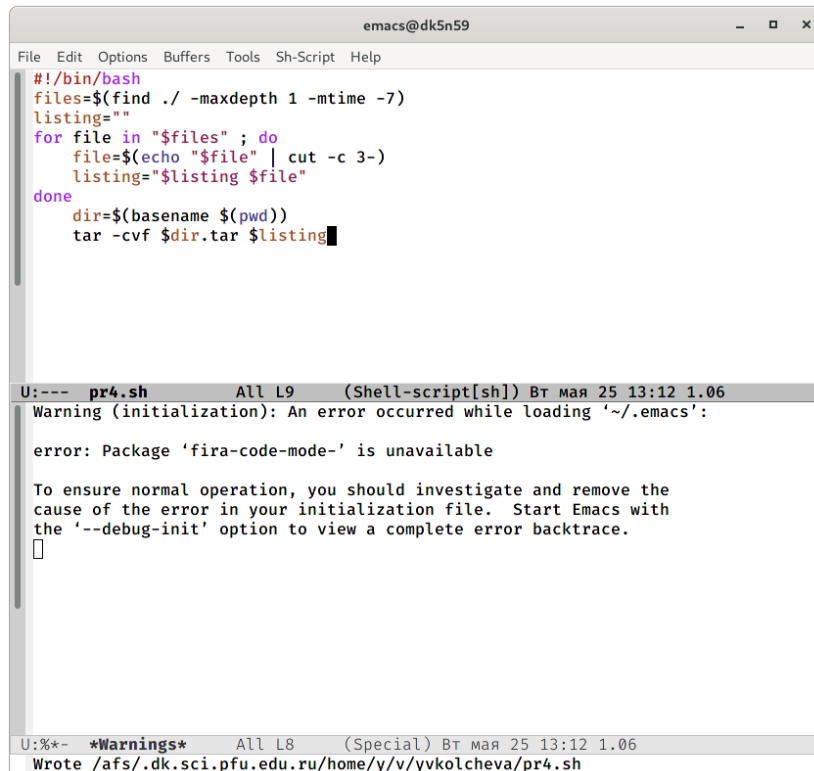
Figure 2.6: Программа 3

Далее я проверила работу написанного скрипта (./files.sh), предварительно добавив право на исполнение файла (команда «chmod+x files.sh»). Сначала я создала три файла (команда «./files.sh -c abc#.txt 3»), удовлетворяющие условию задачи, а потом удалила их (команда «./files.sh -r abc#.txt 3» (рис. 2.7)

```
~: bash — Konsole
Файл Правка Вид Закладки Настройка Справка
введите число
-?
Число меньше 0
yvkolcheva@dk5n59 ~$ touch.sh
bash: touch.sh: команда не найдена
yvkolcheva@dk5n59 ~$ touch file.sh
yvkolcheva@dk5n59 ~$ emacs &
[2] 13342
yvkolcheva@dk5n59 ~$ chmod +x *.sh
[1]- Завершён emacs
yvkolcheva@dk5n59 ~$ ls
asd1.txt  chislo.c  file.sh~  kl.o      prog2.sh~  var      Документы  'Презентация лаб2.pdf'
asd2.txt  chislo.c  format.sh~ labor2    prog1s.sh~ work     Загрузки   'Рабочий стол'
backup.sh~ chislo.sh GNUstep  my_os    public     yvkolch2.cpp  Изображения  Шаблоны
chic1o    chislo.sh~ kl      prog1.sh  public_html yvkolch2.o  Музыка
chislo    file.sh~  kl.cpp   prog1.sh~ tmp        Видео      Общедоступные
yvkolcheva@dk5n59 ~$ ./file.sh -c abc&.txt 3
yvkolcheva@dk5n59 ~$ ls
abc1.txt  backup.sh~  chislo.sh  GNUstep  my_os    public     yvkolch2.cpp  Изображения  Шаблоны
abc2.txt  chic1o     chislo.sh~ kl      prog1.sh  public_html yvkolch2.o  Музыка
abc3.txt  chislo    file.sh~  kl.cpp   prog1.sh~ tmp        Видео      Общедоступные
asd1.txt  chislo.c  file.sh~  kl.o      prog2.sh~  var      Документы  'Презентация лаб2.pdf'
asd2.txt  chislo.c~ format.sh~ labor2    prog1s.sh~ work     Загрузки   'Рабочий стол'
yvkolcheva@dk5n59 ~$ ./file.sh -r abc&.txt 3
yvkolcheva@dk5n59 ~$ ls
asd1.txt  chislo.c  file.sh~  kl.o      prog2.sh~  var      Документы  'Презентация лаб2.pdf'
asd2.txt  chislo.c~ format.sh~ labor2    prog1s.sh~ work     Загрузки   'Рабочий стол'
backup.sh~ chislo.sh GNUstep  my_os    public     yvkolch2.cpp  Изображения  Шаблоны
chic1o    chislo.sh~ kl      prog1.sh  public_html yvkolch2.o  Музыка
chislo    file.sh~  kl.cpp   prog1.sh~ tmp        Видео      Общедоступные
yvkolcheva@dk5n59 ~$
```

Figure 2.7: Проверка работы

Написала командный файл, который с помощью команды tarзапаковывает в архив все файлы в указанной директории. Модифицировала его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад. Для данной задачи я создала файл: pr4.sh и написала соответствующий скрипт.(рис. 2.8)



```
emacs@dk5n59
File Edit Options Buffers Tools Sh-Script Help
#!/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

U:--- pr4.sh All L9 (Shell-script[sh]) Вт мая 25 13:12 1.06
Warning (initialization): An error occurred while loading '~/.emacs':

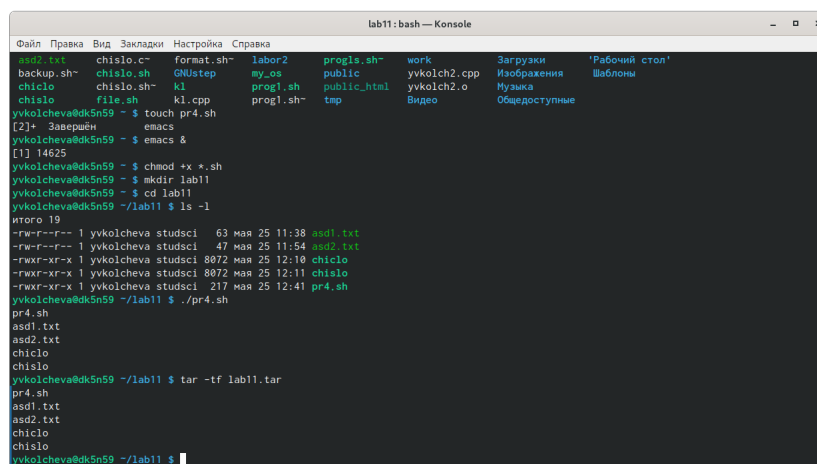
error: Package 'fira-code-mode-' is unavailable

To ensure normal operation, you should investigate and remove the
cause of the error in your initialization file. Start Emacs with
the '--debug-init' option to view a complete error backtrace.

U:%*- *Warnings* All L8 (Special) Вт мая 25 13:12 1.06
Wrote /afs/.dk.sci.pfu.edu.ru/home/y/v/yvkolcheva/pr4.sh
```

Figure 2.8: Программа4

Далее я проверила работу написанного скрипта, предварительно добавив право на исполнение файла (команда «`chmod +x pr4.sh`»)и создав отдельный lab11 с несколькими файлами. (рис. 2.9)



```
lab11: bash — Konsole
Файл Правка Вид Закладки Настройка Справка
backup.sh~ chislo.c~ format.sh~ labor2~ prog1e.sh~ work~ Загрузки~ 'Рабочий стол'
chic1o~ chislo.sh~ GNUstep~ my_os~ public~ yvkolch2.cpp~ Изображения~ Шаблоны
chislo~ chislo.sh~ kl~ prog1.sh~ public_html~ yvkolch2.o~ Музыка
chislo~ file.sh~ kl.cpp~ prog1.sh~ tmp~ Видео~ Общедоступные
yvkolcheva@dk5n59 ~ $ touch pr4.sh
[2]+  Завершён emacs
yvkolcheva@dk5n59 ~ $ emacs &
[1] 14625
yvkolcheva@dk5n59 ~ $ chmod +x *.sh
yvkolcheva@dk5n59 ~ $ mkdir lab11
yvkolcheva@dk5n59 ~ $ cd lab11
yvkolcheva@dk5n59 ~/lab11 $ ls -l
total 10
-rw-r--r-- 1 yvkolcheva studsci 63 мая 25 11:38 asd1.txt
-rw-r--r-- 1 yvkolcheva studsci 47 мая 25 11:54 asd2.txt
-rwxr-xr-x 1 yvkolcheva studsci 8072 мая 25 12:10 chic1o
-rwxr-xr-x 1 yvkolcheva studsci 8072 мая 25 12:11 chislo
-rwxr-xr-x 1 yvkolcheva studsci 217 мая 25 12:41 pr4.sh
yvkolcheva@dk5n59 ~/lab11 $ ./pr4.sh
pr4.sh
asd1.txt
asd2.txt
chic1o
chislo
yvkolcheva@dk5n59 ~/lab11 $ tar -tf lab11.tar
pr4.sh
asd1.txt
asd2.txt
chic1o
chislo
yvkolcheva@dk5n59 ~/lab11 $
```

Figure 2.9: Последняя проверка

3 Выводы

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

4 Контрольные вопросы

1). Команда `getopts` осуществляет синтаксический анализ командной строки, выделяя флаги, 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 введённые 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 введённых пользователем данных.

2). При перечислении имён файлов текущего каталога можно использовать следующие символы: 1 – соответствует произвольной, в том числе и пустой строке; 2 ? – соответствует любому одинарному символу; 3 `[c1-c2]` – соответствует любому символу, лексикографически находящемуся между символами `c1` и `c2`. Например, 1.1 `echo` – выведет имена всех файлов текущего каталога, что представляет собой простейший аналог команды `ls`; 1.2. `ls.c` – выведет все файлы с последними двумя символами, совпадающими

с.с. 1.3. *echoprogram*.?–выведет все файлы, состоящие из пяти или шести символов, первыми пятью символами которых являются *prog*.. 1.4.[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 от результатов 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, неравный 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*

`fmans/i.s`, `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.