

Лабораторная работа №12

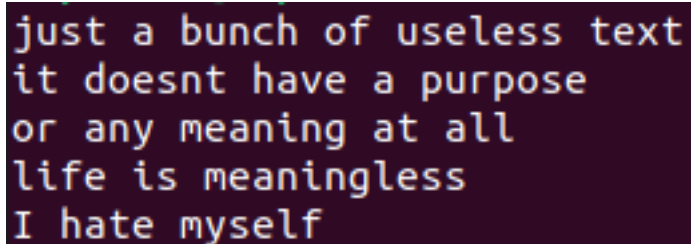
Покрас Илья Михайлович

Цель работы

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

Выполнение лабораторной работы

1. Используя команды `getopts` `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;
 - `-п` — выдавать номера строк. а затем ищет в указанном файле нужные строки, определяемые ключом `-р`. (рис. -@fig:001)



```
just a bunch of useless text
it doesnt have a purpose
or any meaning at all
life is meaningless
I hate myself
```

text

```

#!/bin/bash
while getops 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 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} > ${oval}
    fi
fi
if (((cflag==1)&&(nflag==1)))
then grep -e${pval} -i -n ${ival}
    if ((oflag==1))
    then grep -e${pval}-i -n ${ival} > ${oval}
    fi
fi
if (((cflag==1)&&(nflag==1)))
then grep -e${pval} -i -n ${ival}
    if ((oflag==1))
    then grep -e${pval}-i -n ${ival} > ${oval}
    fi
fi

```

script 1

```
impokras@impokras-VirtualBox:~/lab12$ bash lab12.sh -ilab12.txt -oresult.txt -plife -c
lab12.sh: недопустимый параметр - c
Illegal option ?
life is meanengless
impokras@impokras-VirtualBox:~/lab12$ bash lab12.sh -ilab12.txt -oresult.txt -plife -C
life is meanengless
impokras@impokras-VirtualBox:~/lab12$ bash lab12.sh -ilab12.txt -oresult.txt -plifE -C
life is meanengless
impokras@impokras-VirtualBox:~/lab12$
```

script 1 result

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.)

```
#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);
}
```

script 2 c

```
#!/bin/bash
gcc -o cprog lab12.c
./cprog
case $? in
    0)echo 'input number equal to 0';;
    1)echo 'input number smaller than 0';;
    2)echo 'input number bigger than 0';;
esac
```

script 2 .sh

```
impokras@impokras-VirtualBox:~/lab12$ bash lab12.3.sh
Input: 1
input number bigger than 0
impokras@impokras-VirtualBox:~/lab12$ emacs lab12.3.sh
impokras@impokras-VirtualBox:~/lab12$ bash lab12.3.sh
Input: 3
input number is bigger than 0
impokras@impokras-VirtualBox:~/lab12$ emacs
```

script 2 result

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.).

```
#!/bin/bash
let dflag=0;
while getopts a:d optletter
do case $optletter in
    a) aflag=1; aval=$OPTARG;;
    d) 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
```

script 3


```
-rw-rw-r-- 1 impokras impokras 359 окт 25 23:27 12.4.sh
-rw-rw-r-- 1 impokras impokras 347 окт 25 23:25 12.4.sh~
-rwxrwxr-x 1 impokras impokras 16240 окт 25 23:18 cprog
-rw-rw-r-- 1 impokras impokras 190 окт 25 23:17 lab12.3.sh
-rw-rw-r-- 1 impokras impokras 181 окт 25 23:14 lab12.3.sh~
-rw-rw-r-- 1 impokras impokras 192 окт 25 23:02 lab12.c
-rw-rw-r-- 1 impokras impokras 823 окт 25 22:51 lab12.sh
-rw-rw-r-- 1 impokras impokras 823 окт 25 22:49 lab12.sh~
-rw-rw-r-- 1 impokras impokras 100 окт 25 22:57 lab12.txt
-rw-rw-r-- 1 impokras impokras 0 окт 25 22:58 result.txt
```

script 3 result

Написал командный файл, который с помощью команды tar запаковывает в архив все файлы в указанной директории.

```
#!/bin/bash
tar -cf 12.tar $@
tar -cf 12l.tar
find $@ -mtime -7 -exec tar -rf 9l.tar '{}' ';' █
```

script 4

```
impokras@impokras-VirtualBox:~/lab12$ bash lab12.5.sh /lab12
tar: Удаляется начальный '/' из имен объектов
tar: /lab12: функция stat завершилась с ошибкой: Нет такого файла или каталога
tar: Завершение работы с состоянием неисправности из-за возникших ошибок
tar: Робкий отказ от создания пустого архива
tar: Попробуйте «tar --help» или «tar --usage» для
получения более подробного описания.
find: '/lab12': Нет такого файла или каталога
impokras@impokras-VirtualBox:~/lab12$ ls
12.4.sh  12.tar  lab12.3.sh  lab12.5.sh  lab12.sh  lab12.txt
12.4.sh~ cprog  lab12.3.sh~ lab12.c     lab12.sh~ result.txt
impokras@impokras-VirtualBox:~/lab12$ █
```

script 4 result

Заключение и выводы

В ходе работы я изучил основы программирования в оболочке ОС UNIX и научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

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 mans/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.