# Лабораторная работа №12

Выполнил Бабков Дмитрий Николаевич, НПМбд-01-20

# Цель работы

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

# Задание

Написать командные файлы, выполняющие указанные действия

# Ход работы

Командный файл ищет во входном файле заданный текст, а потом выводит результат в выходной файл (Рис.1)

```
task1.sh           [-M--] 115 L:[  1+25  26/ 26] *(1455/1455b) <EOF>
nflag=""<------>#Присвоение переменной nflag пустой строки
Cflag="-i"<---->#Присвоение переменной Cflag оператора -i команды grep
while getopts i:o:p:Cn optletter; do<-->#Чтение всех операторов
    case $optletter in
<------>i) ival=$OPTARG;;<----->#Присвоение переменной имени входного файла
<------>o) oval=$OPTARG;;<----->#Присвоение переменной имени выходного файла
<------>p) pval=$OPTARG;;<----->#Присвоение переменной шаблона поиска
<------>C) Cflag="";;<-><------>#Удаление из переменной оператора -i
<------>n) nflag="-n";;><------>#Присвоение оператора -n команды grep для вывода номера строки
<------>*) echo "Illegal option $optletter"<--->#Вывод ошибки при попытке ввести непредусмотренные операторы
    esac
done

if [ -z $ival ]; then
    echo "No input file"; exit<>#Следующие три if - сообщения об ошибках, связанных с недостаточностью введенных данных
fi

if [ -z $oval ]; then
    echo "No output file"; exit
fi

if [ -z $pval ]; then
    echo "No pattern"; exit
fi

grep $nflag $Cflag $pval $ival > $oval<>#Вывод в файл output результата работы команды grep с заданными параметрами
```

# Ход работы

Командный файл анализирует код выхода программы и выводит сообщение о введенном в программу числе на экран (Рис.2, 3)

prog.c [-M--] 1

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

int main(){

    int n;

    scanf("%d", &n);

    if (n < 0) {
<------>exit(1);
    }

    if (n > 0) {
<------>exit(2);
    }

    exit(0);
}
```

task2.sh [----] 70 L:[ 1+ 7 8/ 8] *(388 / 388b) <EOF>

```
./prog<><------>#Запуск программы на языке C
case $? in<---->#Анализ кода завершения программы
0)<---->s="равно нулю.";;<----->#Соответствующие выводы
1)<---->s="меньше нуля.";;
2)<---->s="больше нуля.";;
esac

echo "Ведённое число $s"<------>#Вывод с соответствующим числу выводом
```

# Ход работы

Командный файл создает указанное число файлов или удаляет то, что создал. (Рис.4)

```
task3.sh           [----] 80 L:[  1+17  18/ 20] *(619 / 636b) 0010 0x00A
num=0
while getopts c:d ol<-->#Поиск операторов
do
case $ol in
    c)num=$OPTARG;;<--->#Кол-во файлов, которое нужно создать
    d)del=true;;<------->#Файлы должны быть удалены
    *)echo "Wait, that's illegal!"<---->#Анализ некорректных операторов
esac
done

if [ $num -gt 0 ]<------>#Создание файлов, если кол-во не равно нулю
then for (( i=1; i<=$num; i++ ))
    do
<------>touch ${i}.tmp
    done
fi

if [ "$del" = true ]<-->#Удаление всех файлов с расширением .tmp, если это нужно
then rm *.tmp
fi
```

# Ход работы

Командный файл с помощью команды **tar** запаковывает файлы указанной директории (Рис.5)

# Ход работы

Модифицированный командный файл, который запаковывает файлы, которые были изменены менее недели назад (Рис.6)

```
task4.sh           [----] 166 L:[  1+ 9  10/ 12] *(545 / 585b) 0010 0x00A
path=$1><------><------>#Имя директории, файлы которой нужно запаковать
arcName="dirArch.tar"

if [ -z $path ]; then<->#Проверка на наличие пути
    echo "No path"
    exit
fi

tar -cvf $arcName -T /dev/null<>#Создание пустого архива
tar -rvf $arcName `find $path -type f -mtime -7 -not -name "$arcName*"`>#Архивирование файлов директории, которые были изменены менее недели назад, исключая сам архив

gzip $arcName<->#Сжатие ахива
```

# Вывод

В ходе выполнения лабораторной работы я научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

# Спасибо за внимание