

Лабораторная работа №11

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Абдулгалимов Мурад

Содержание

1 Цель работы	5
2 Задание	6
3 Выполнение лабораторной работы	7
3.0.1 Используя команды getopt, grep, написал командный файл, который анализирует командную строку с ключами:	7
3.0.2 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. (рис. 3.2 3.3)	8
3.0.3 Написал командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до \square (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. (рис. 3.4)	9
3.0.4 Написал 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). (рис. 3.5)	10
4 Выводы	11
5 Контрольные вопросы	12
Список литературы	14

Список иллюстраций

3.1 Программа 1	7
3.2 Программа 2	8
3.3 Программа 2	8
3.4 Программа 3	9
3.5 Программа 4	10

Список таблиц

1 Цель работы

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

2 Задание

1. Используя команды getopt grep, написать командный файл, который анализирует командную строку с ключами: – -iinputfile — прочитать данные из указанного файла; – -ooutputfile — вывести данные в указанный файл; – -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 -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 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.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до \square (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).

3 Выполнение лабораторной работы

3.0.1 Используя команды getopt 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;
- -С – различать большие и малые буквы;
- -n – выдавать номера строк. (рис. 3.1)

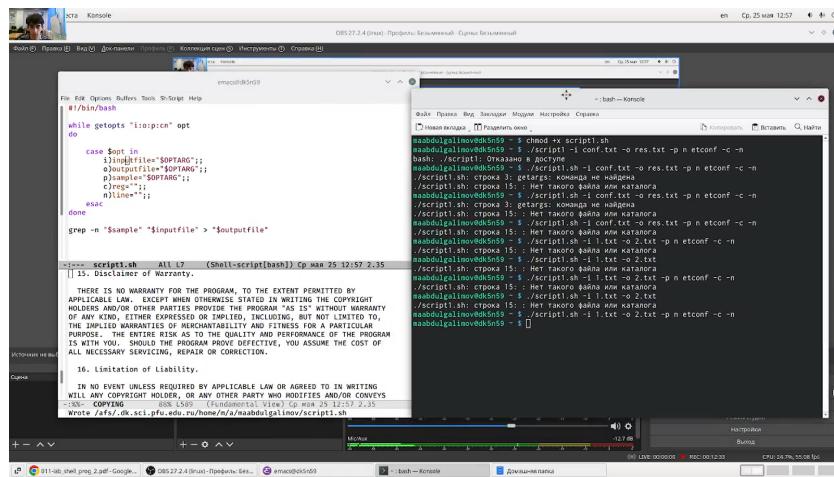


Рис. 3.1: Программа 1

3.0.2 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. (рис. 3.2 3.3)

```

int main(int argc, char *argv[])
{
    if (atoi(argv[1]) > 0) {
        exit(1);
    } else if (atoi(argv[1]) == 0) {
        exit(2);
    } else {
        exit(3);
    }
    return 0;
}

```

Рис. 3.2: Программа 2

```

#!/bin/bash
RS=$result
SRC=compare.cpp
if [ "$SRC" = "SRES" ]
then
    echo "creating SRES..."
    rm -r $SRC
fi
if [ "$RS" != "" ]
then
    if [ "$rs" == "1" ]
    then
        echo "Input > 0"
    fi
    if [ "$rs" == "2" ]
    then
        echo "Input == 0"
    fi
    if [ "$rs" == "3" ]
    then
        echo "Input < 0"
    fi
fi
exit 0

```

Рис. 3.3: Программа 2

3.0.3 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. (рис. 3.4)

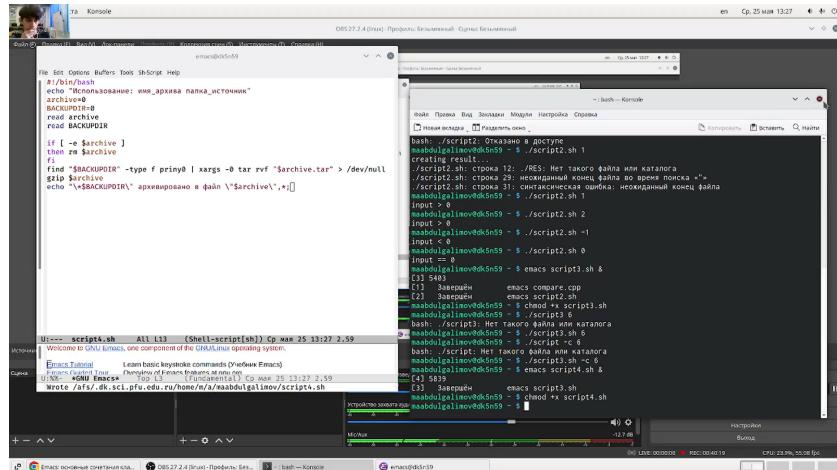
```

#!/bin/bash
while getopts "gi" opt
do
    case $opt in
        g)*"$OPTARG"; for i in $(seq 1 $n); do touch "$i.txt"; done;;
        i)*for i in $(find -name "*.tmp"); do rm $i; done;;
        esac
    done

```

Рис. 3.4: Программа 3

3.0.4 Написал 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). (рис. 3.5)



```

#!/bin/bash
#Использование: имя_архива папка_источник
tar -cvf $1 $2
BACKUPDIR=$1
read archive
read BACKUPDIR
if [ -e $archive ]
then rm $archive
fi
find "$BACKUPDIR" -type f priny0 | xargs -o tar rvf "$archive.tar" > /dev/null
gzip $archive
echo "У$BACKUPDIR" архивирован в файл \'$archive\'."

```

Рис. 3.5: Программа 4

4 Выводы

Научился писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.

5 Контрольные вопросы

1. Она осуществляет синтаксический анализ командной строки, выделяя флаги, и используется для объявления переменных.².
2. При генерации имен файлов 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 - выдаст файлы данного каталога с однобуквенными расширениями, program.? - 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. for, case, if, while.
4. Break, continue.
5. 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.

6. Означает условие существования файла `mans/i.$s`.
7. Если речь идет о 2-х параллельных действиях, то это `while`. когда мы показываем, что сначала делается 1-е действие. потом оно заканчивается при наступлении 2-го действия, применяем `until`.

Список литературы