

Лабораторная работа №11

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Аскеров А.Э.

22 апреля 2023

Российский университет дружбы народов, Москва, Россия

Вступление

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

Выполнение лабораторной работы

Задание 1

1. Используя команды `getopts` `grep`, напомним 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`.

Задание 1

Откроем командный файл, напишем программу, сохраним её, дадим файлу право на выполнение, запустим его и проверим наличие файла, в который выводятся данные.

```
[aeaskerov@fedora lab11]$ vi lab111
[aeaskerov@fedora lab11]$ chmod +x lab111
[aeaskerov@fedora lab11]$ ./lab111 -i example.txt -o output.txt -p h -c -n
[aeaskerov@fedora lab11]$ ls
compare.cpp  example.txt  lab111  output.txt  presentation  report
[aeaskerov@fedora lab11]$
```

Рис. 1: Открытие командного файла, разрешение на исполнение, запуск, проверка (сформировался ли файл output.txt, куда выводятся данные)

```
while getopts "i:o:p:c:n" opt
do
case $opt in
i)inputfile="$OPTARG";;
o)outputfile="$OPTARG";;
p)shablon="$OPTARG";;
c)registr="";;
n)number="";;
esac
done

grep -n "$shablon" "$inputfile" > "$outputfile"
```

Рис. 2: Программа для задания один

Задание 2

2. Напишем 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 коде завершения в оболочку. Командный файл должен вызывать эту программу и, проанализировав с помощью команды `$?`, выдать сообщение о том, какое число было введено.

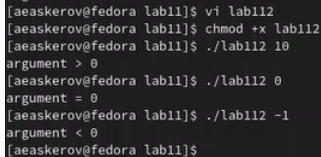
Напишем программу на языке C++.

```
#include <iostream>
using namespace std;

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

Рис. 3: Программа на языке C++

Откроем командный файл, напомним программу, сохраним её, дадим файлу право на выполнение и запустим его.



```
[aeaskerov@fedora lab11]$ vi lab112
[aeaskerov@fedora lab11]$ chmod +x lab112
[aeaskerov@fedora lab11]$ ./lab112 10
argument > 0
[aeaskerov@fedora lab11]$ ./lab112 0
argument = 0
[aeaskerov@fedora lab11]$ ./lab112 -1
argument < 0
[aeaskerov@fedora lab11]$
```

Рис. 4: Открытие командного файла, разрешение на исполнение, запуск


```
#!/bin/bash

CC=g++
EXEC=compare
SRC=compare.cpp

if [ "$SRC" -nt "$EXEC" ]
then
    echo "Rebuilding $EXEC ....."
    $CC -o $EXEC $SRC
fi

./$EXEC $1
ec=$?
if [ "$ec" == "1" ]
then
    echo "argument > 0"
fi
if [ "$ec" == "2" ]
then
    echo "argument = 0"
fi
if [ "$ec" == "3" ]
then
    echo "argument < 0"
fi
```

Рис. 5: Программа для задания два

Задание 3

3. Напишем 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).

Откроем командный файл, напишем программу, сохраним её, дадим файлу право на выполнение, запустим его и проверим, были ли созданы и затем удалены пять файлов формата tmp.

```
[aeaskerov@fedora lab11]$ vi lab113
[aeaskerov@fedora lab11]$ chmod +x lab113
[aeaskerov@fedora lab11]$ ./lab113 -c 5
[aeaskerov@fedora lab11]$ ls
1.tmp 3.tmp 5.tmp  compare.cpp  example.txt  lab112  output.txt  report
2.tmp 4.tmp  compare  compare.cpp~  lab111      lab113  presentation
[aeaskerov@fedora lab11]$ ./lab113 -r
[aeaskerov@fedora lab11]$ ls
compare  compare.cpp~  lab111  lab113  presentation
compare.cpp  example.txt  lab112  output.txt  report
[aeaskerov@fedora lab11]$
```

Рис. 6: Открытие командного файла, разрешение на исполнение, запуск, проверка

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

Рис. 7: Программа для задания три

Задание 4

4. Напишем командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории. Модифицируем его так, чтобы запаковывались только те файлы, которые были изменены менее недели тому назад (использовать команду `find`).

Откроем командный файл, напишем программу, сохраним её, дадим файлу право на выполнение, создадим каталог `catalog2`, перейдём в него, создадим в нём несколько файлов, запустим командный файл.

```
[aeaskerov@fedora ~]$ touch lab114.sh
[aeaskerov@fedora ~]$ vi lab114.sh
[aeaskerov@fedora ~]$ chmod +x lab114.sh
[aeaskerov@fedora ~]$ mkdir catalog2
[aeaskerov@fedora ~]$ cd catalog2
[aeaskerov@fedora catalog2]$ touch file1.txt file2.txt file3.txt
[aeaskerov@fedora catalog2]$ ls
file1.txt file2.txt file3.txt
[aeaskerov@fedora catalog2]$ sudo ~/lab114.sh
file1.txt
file2.txt
file3.txt
[aeaskerov@fedora catalog2]$
```

Рис. 8: Открытие командного файла, разрешение на исполнение, создание каталога и файлов в нём, запуск командного файла

Задание 4

Проверим, были ли запакованы файлы в каталоге.



Рис. 9: Проверка наличия архива

```
#!/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
```

Рис. 10: Программа для задания четыре

Заключение

Изучены основы программирования в оболочке ОС UNIX. Приобретён навык написания более сложных командных файлов с использованием логических управляющих конструкций и циклов.