

Презентация по лабораторной работе №13

Программирование в командном процессоре ОС UNIX. Ветвления и циклы

Вакутайпа М.

30 апреля 2024

Российский университет дружбы народов, Москва, Россия

Информация

- Вакутайпа Милдред
- НКА 02-23
- факультет физико математических и естественных наук
- Российский университет дружбы народов
- 1032239009@rudn.ru
- <https://wakutaipa.github.io/ru/>

.....
.....

Цель работы

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

Задание

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку
2. Написать на языке Си программу, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю.
3. Написать командный файл, создающий указанное число файлов, пронумерованных последовательно от 1 до N
4. Написать командный файл, который с помощью команды `tar` запаковывает в архив все файлы в указанной директории.

Выполнение лабораторной работы

Создаю файл `file1` и в нем написала код, который анализирует командную строку с ключами `-i` (прочитать данные из указанного файла), `-o` (вывести данные в указанный файл), `-p` (указать шаблон для поиска), `-C` (различать большие и малые буквы), `-n` (выдавать номера строк) используя команды `getopts` `grep`:

командный файл, который анализирует командную строку

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

```
grep -n "$template" "$inputfile.txt" > "$outputfile.txt"
```

командный файл, который анализирует командную строку

```
mwakutaipa@mwakutaipa:~$ ./file1.txt -i file1 -o output -p n etconf -C -n
mwakutaipa@mwakutaipa:~$ cat output.txt
1:while getopts "i:o:p:C:n" opt
3:case $opt in
4:i)inputfile="$OPTARG";;
8:n)number="";;
10:done
12:grep -n "$template" "$inputfile.txt" > "$outputfile.txt"
```

Рис. 1: Запуск file1

программа, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю.

Программа завершается с помощью функции `exit(n)`, передавая информацию о коде завершения в оболочку:

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

int main()
{
    int n;
    printf("Enter a number: ");
    scanf("%d", &n);
    if(n>0)
```

программа, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю.

```
{  
    exit(1);  
}  
  
else if (n==0) {  
    exit(0);}   
else  
{  
    exit(2);  
}  
}
```

программа, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю.

Далее создала командный файл который вызывает эту программу и, проанализировав с помощью команды \$?, выдает сообщение о том, какое число было введено:

```
gcc -o cprog file2.c  
./cprog
```

```
case $? in  
0) echo "равно нулю";;  
1) echo "больше нуля";;  
2) echo "меньше нуля";;
```

```
esac
```

программа, которая вводит число и определяет, является ли оно больше нуля, меньше нуля или равно нулю.

Создала исполняемый файл и запустила:

A terminal window with a black background and green text. The prompt is 'mwakutaipa@mwakutaipa:~\$'. The user enters 'gedit file2.c', then 'gedit command_file.sh', then 'chmod +x command_file.sh', and finally './command_file.sh'. The program prompts 'Enter a number: 7' and outputs 'больше нуля'.

```
mwakutaipa@mwakutaipa:~$ gedit file2.c
mwakutaipa@mwakutaipa:~$ gedit command_file.sh
mwakutaipa@mwakutaipa:~$ chmod +x command_file.sh
mwakutaipa@mwakutaipa:~$ ./command_file.sh
Enter a number: 7
больше нуля
```

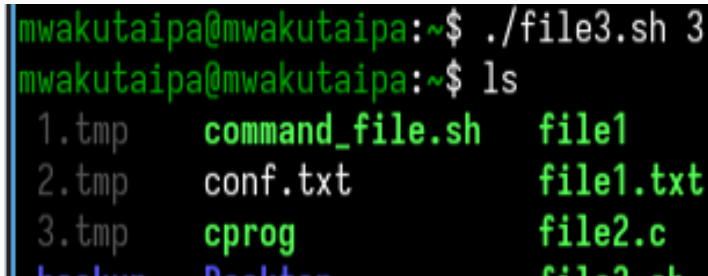
Рис. 2: Результаты программы

командный файл, создающий указанное число файлов

Я 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 (если они существуют):

```
for((i=1; i<=$*; i++))  
do  
if test -f "$i".tmp  
then rm "$i".tmp  
else touch "$i.tmp"  
fi  
done
```


Создала исполняемый файл и запустила:

A terminal window with a black background and green text. The prompt is 'mwakutaipa@mwakutaipa:~\$'. The first command is './file3.sh 3'. The second command is 'ls'. The output of 'ls' shows a directory listing with three columns of files: '1.tmp', '2.tmp', and '3.tmp' in the first column; 'command_file.sh', 'conf.txt', and 'cprog' in the second column; and 'file1', 'file1.txt', 'file2.c', and 'file2.sh' in the third column.

```
mwakutaipa@mwakutaipa:~$ ./file3.sh 3
mwakutaipa@mwakutaipa:~$ ls
1.tmp      command_file.sh  file1
2.tmp      conf.txt         file1.txt
3.tmp      cprog           file2.c
hookup     Backup          file2.sh
```

Рис. 3: Создание файлов с помощью командного файла

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

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

```
find $* -mtime -7 -mtime +0 -type f > FILES.txt  
tar -cf archive.tar -T FILES.txt
```

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

```
mwakutaipa@mwakutaipa:~$ gedit file4.sh
mwakutaipa@mwakutaipa:~$ chmod +x file4.sh
mwakutaipa@mwakutaipa:~$ ./file4.sh /home/mwakutaipa/work
tar: Удаляется начальный '/' из имен объектов
tar: Удаляются начальные '/' из целей жестких ссылок
mwakutaipa@mwakutaipa:~$ ls ~/work
blog os study wakutaipa.github.io
mwakutaipa@mwakutaipa:~$ ls
1.tmp          command_file.sh  file1.txt       git-extended
2.tmp          conf.txt         file2.c         helloworld.cpr
3.tmp          cprog           file3.sh        '#lab7.sh#'
archive.tar    Desktop         file4.sh        lab7.sh
```

Рис. 4: Результаты кода

Выводы

При выполнении проделанной работы я научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.