

Презентация по лабораторной работе №12

Операционные системы

Скрипникова София Дмитриевна

29 апреля 2023

Российский университет дружбы народов, Москва, Россия

Цель работы

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

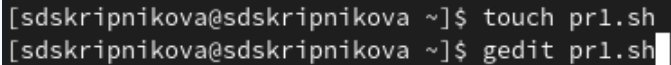
Задание

1. Используя команды `getopts` `grep`, написать командный файл, который анализирует командную строку с ключами:
 - `-i`inputfile — прочитать данные из указанного файла;
 - `-o`outputfile — вывести данные в указанный файл;
 - `-р`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 `o` 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 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).
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).

Выполнение лабораторной работы

1. Создала файл для программы 1

A terminal window with a dark background and light gray text. It shows two lines of commands entered at the prompt. The first line is '[sdskripnikova@sdskripnikova ~]\$ touch pr1.sh' and the second line is '[sdskripnikova@sdskripnikova ~]\$ gedit pr1.sh'. A white cursor is visible at the end of the second line.

```
[sdskripnikova@sdskripnikova ~]$ touch pr1.sh  
[sdskripnikova@sdskripnikova ~]$ gedit pr1.sh
```

Рис. 1: Создание файла

2. Написала текст программы 1

```
1 #!/bin/bash
2 t1=$1
3 t2=$2
4 s1=$(date +%s)
5 s2=$(date +%s)
6 ((t=$s2 - $s1))
7 while ((t<t1))
8 do
9     echo "Ожидание"
10    sleep 1
11    s2=$(date +%s)
12    ((t=$s2 - $s1))
13 done
14 s1=$(date +%s)
15 s2=$(date +%s)
16 ((t=$s2 - $s1))
17 while ((t<t2))
18 do
19     echo "Выполнение"
20    sleep 1
21    s2=$(date +%s)
22    ((t=$s2 - $s1))
23 done
```

3. Проверила работу написанной программы

```
[sdskripnikova@sdskripnikova ~]$ chmod +x pr1.sh
[sdskripnikova@sdskripnikova ~]$ ./pr1.sh 3 5
Ожидание
Ожидание
Ожидание
Выполнение
Выполнение
Выполнение
Выполнение
Выполнение
[sdskripnikova@sdskripnikova ~]$
```

Рис. 3: Результат работы программы

4. Отредактировала текст

```
1 #!/bin/bash
2
3 function ozhidanie
4
5 {
6
7     s1=$(date +%s)
8
9     s2=$(date +%s)
10
11     ((t=s2 - s1))
12
13     while ((t<t1))
14     do
15
16
17     echo "Ожидание"
18
19     sleep 1
20
21     s2=$(date +%s)
22
23     ((t=s2 -s1))
24
25     done
26
27 }
28
29 function vipolnenie
30
31 {
32
33     s1=$(date +%s)
```

```
54
55 t1=$1
56
57 t2=$2
58
59 command=$3
60
61 while true
62 do
63
64
65     if [ "$command" == "Выход" ]
66
67     then
68
69     echo "Выход"
70
71     exit 0
72
73     fi
74
75     if [ "$command" == "Ожидание" ]
76
77     then ozhidanie
78
79     fi
80
81     if [ "$command" == "Выполнение" ]
82
83     then vipolnenie
84
85     fi
86
87     echo "Следующее действие: "
88
89     read command
90
91 done
```

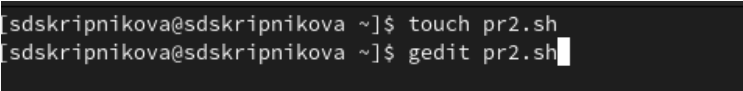
Рис. 5: Текст

6. Проверила работу написанной программы

```
[sdscripnikova@sdscripnikova ~]$ gedit pr1.sh  
[sdscripnikova@sdscripnikova ~]$ ./pr1.sh 3 5  
Следующее действие:  
ожидание  
Следующее действие:  
выполнение  
Следующее действие:  
█
```

Рис. 6: Результат работы программы

7. Создала файл для программы 2

A terminal window with a dark background and light gray text. It shows two lines of commands being executed. The first line is '[sdscripnikova@sdscripnikova ~]\$ touch pr2.sh' and the second line is '[sdscripnikova@sdscripnikova ~]\$ gedit pr2.sh' followed by a white cursor block.

```
[sdscripnikova@sdscripnikova ~]$ touch pr2.sh  
[sdscripnikova@sdscripnikova ~]$ gedit pr2.sh
```

Рис. 7: Создание файла

8. Написала текст программы 2

```
1 #!/bin/bash
2
3 a=$1
4
5 if [ -f /usr/share/man/man1/$a.1.gz ]
6
7 then
8
9     gunzip -c /usr/share/man/man1/$1.1.gz | less
10
11 else
12
13     echo "Справки по данной команде нет"
14
15 fi
```

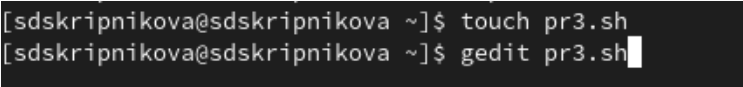
Рис. 8: Создание файла

9. Проверила работу написанной программы

```
.\" DO NOT MODIFY THIS FILE!  It was generated by help2man 1.48.5.
.TH MKDIR "1" "January 2023" "GNU coreutils 9.0" "User Commands"
.SH NAME
mkdir \- make directories
.SH SYNOPSIS
.B mkdir
[\fI\,OPTION\[/\fR]... \fI\,DIRECTORY\[/\fR...
.SH DESCRIPTION
.\" Add any additional description here
.PP
Create the DIRECTORY(ies), if they do not already exist.
.PP
Mandatory arguments to long options are mandatory for short options too.
.TP
\fB\m\[/fR, \fB\-\-mode\[/fR=\fI\,MODE\[/\fR
set file mode (as in chmod), not a=rwx \- umask
.TP
\fB\p\[/fR, \fB\-\-parents\[/fR
no error if existing, make parent directories as needed,
with their file modes unaffected by any \fB\m\[/fR option.
.TP
\fB\v\[/fR, \fB\-\-verbose\[/fR
```

Рис. 9: Результат работы программы

10. Создала файл для программы 3

A terminal window with a black background and white text. It shows two commands being executed in a shell. The first command is 'touch pr3.sh' and the second is 'gedit pr3.sh'. Both commands are preceded by the prompt '[sdscripnikova@sdscripnikova ~]\$'.

```
[sdscripnikova@sdscripnikova ~]$ touch pr3.sh  
[sdscripnikova@sdscripnikova ~]$ gedit pr3.sh
```

Рис. 10: Создание файла

11. Написала текст программы 3

```
1 #!/bin/bash
2
3 a=$1
4
5 for ((i=0; i<$a; i++))
6
7 do
8
9     ((char=$RANDOM%26+1))
10
11     case $char in
12
13 1) echo -n a;; 2) echo -n b;; 3) echo -n c;; 4) echo -n d;; 5) echo -n e;; 6) echo -n f;; 7) echo -n g;; 8)
    echo -n h;; 9) echo -n i;;
14
15 10) echo -n j;; 11) echo -n k;; 12) echo -n l;; 13) echo -n m;; 14) echo -n n;; 15) echo -n o;; 16) echo -n
    p;; 17) echo -n q;;
16
17 18) echo -n r;; 19) echo -n s;; 20) echo -n t;; 21) echo -n u;; 22) echo -n v;; 23) echo -n w;; 24) echo -n
    x;; 25) echo -n y;;
18
19 26) echo -n z;;
20
21     esac
22
23 done
24
25 echo
```

Рис. 11: Создание файла

12. Проверила работу написанной программы

```
[sdskripnikova@sdskripnikova ~]$ chmod +x pr3.sh  
[sdskripnikova@sdskripnikova ~]$ ./pr3.sh 15  
nfflvroypmvgsrv  
[sdskripnikova@sdskripnikova ~]$
```

Рис. 12: Результат работы программы

Выводы

В ходе выполнения данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.