

Лабораторная работа № 12

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Талебу тенке франк устон

01 января 1970

Российский университет дружбы народов, Москва, Россия

Объединённый институт ядерных исследований, Дубна, Россия

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

. Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом). Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой (`> /dev/tty#`, где `#` — номер терминала куда перенаправляется вывод), в котором также запущен этот файл, но не фоновом, а в привилегированном режиме. Доработать программу так, чтобы имелась возможность взаимодействия трёх и более процессов.

2. Реализовать команду `man` с помощью командного файла. Изучите содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд. Каждый архив можно открыть командой `less` сразу же просмотрев содержимое справки. Командный файл должен получать в виде аргумента командной строки название команды и в виде результата выдавать справку об этой команде

(рис. (fig:001?)).

```
2
3 Loc
4 exe
5 thouston@username:~$ mkdir lab12
6 thouston@username:~$ cd lab 12
7 bash: cd: слишком много аргументов
8 thouston@username:~$ cd lab12
9 thouston@username:~/lab12$ touch prog1.sh
10 thouston@username:~/lab12$ ls
11 prog1.sh
12 thouston@username:~/lab12$ touch prog2.sh
13 thouston@username:~/lab12$ touch prog3.sh
14 thouston@username:~/lab12$ ls
15 prog1.sh prog2.sh prog3.sh
16 thouston@username:~/lab12$ gedit prog1.sh
17
18 don
```

Рис. 1: Название рисунка

(рис. (fig:002?)).



```
1 #!/bin/bash
2
3 Lockfile="./lock.file"
4 exec{fn}>$lockfile
5
6 while test -f "$lockfile"
7 do
8   if flock -n ${fn}
9   then
10     echo "File is blocked"
11     sleep 5
12     echo "File is unblocked"
13     flock -u ${fn}
14   else
15     echo "File is blocked"
16     sleep 5
17   fi
18 done
```

Рис. 2: Название рисунка

(рис. (fig:003?)).

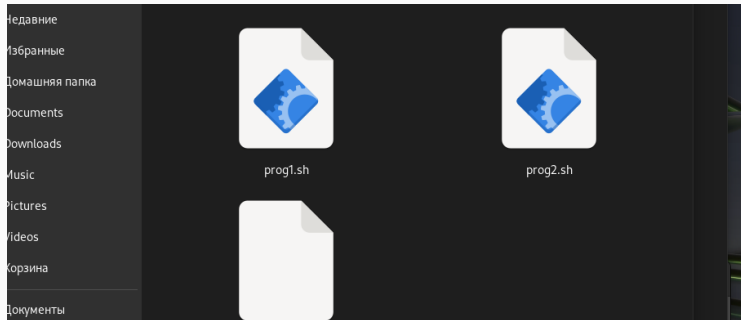


Рис. 3: Название рисунка

(рис. (fig:004?)).



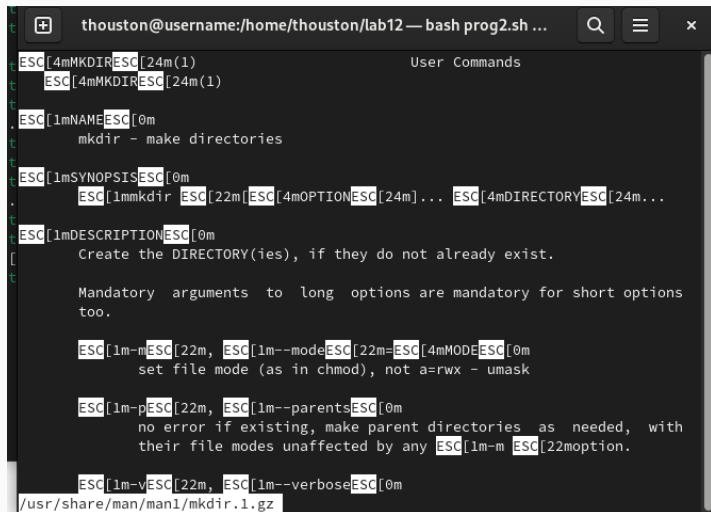
The image shows a terminal window with a dark title bar. The title bar contains the text "C6, 11 мая 13:56" on the left, "en" on the right, and a window control icon. Below the title bar is a light gray toolbar with buttons labeled "Открыть", a dropdown arrow, a "+" icon, the filename "prog2.sh", the path "~/lab12", a "Сохранить" button, a menu icon, and a close "x" button. The main area of the terminal is white and contains a shell script with the following lines:

```
1 #!/bin/bash
2
3 a=$1
4 if
5 test -f "/usr/share/man/man1/$a.1.gz"
6 then less /usr/share/man/man1/$a.1.gz
7 else
8 echo "There is no such command"
9 fi
```

Рис. 4: Название рисунка

Выполнение лабораторной работы

(рис. (fig:005?)).



```
thouston@username:/home/thouston/lab12 — bash prog2.sh ...
User Commands
mkdir - make directories
SYNOPSIS
  mkdir [-m, --mode=MODE] DIRECTORY...
DESCRIPTION
  Create the DIRECTORY(ies), if they do not already exist.

  Mandatory arguments to long options are mandatory for short options
  too.

  -m, --mode=MODE
    set file mode (as in chmod), not a=rwx - umask

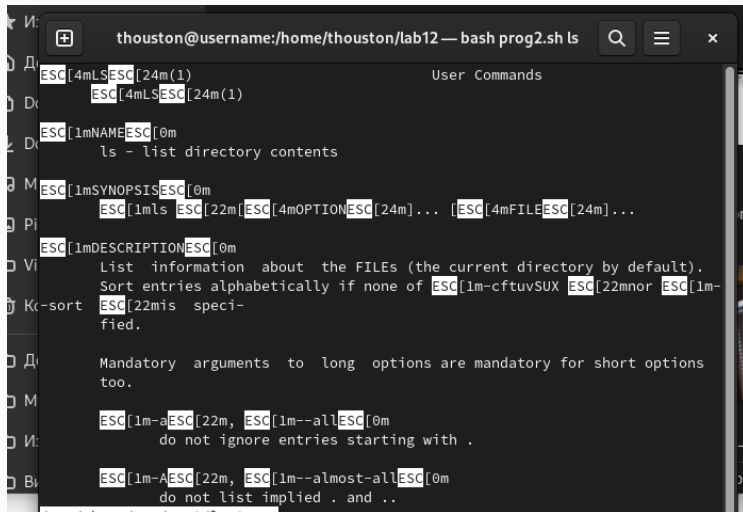
  -p, --parents
    no error if existing, make parent directories as needed, with
    their file modes unaffected by any -m option.

  -v, --verbose
    print a message for each directory created.

/usr/share/man/man1/mkdir.1.gz
```


Выполнение лабораторной работы

(рис. (fig:006?)).



```
thouston@username:/home/thouston/lab12 — bash prog2.sh ls
ESC[4mLSESC[24m(1)                                User Commands
ESC[4mLSESC[24m(1)

ESC[1mNAMEESC[0m
ls - list directory contents

ESC[1mSYNOPSISESC[0m
ESC[1mls ESC[22mESC[4mOPTIONESC[24m... [ESC[4mFILEESC[24m]...

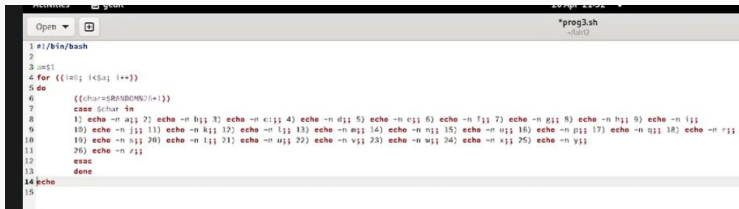
ESC[1mDESCRIPTIONESC[0m
List information about the FILES (the current directory by default).
Sort entries alphabetically if none of ESC[1m-cftuvSUX ESC[22mnor ESC[1m-
ESC[22mis speci-
fied.

Mandatory arguments to long options are mandatory for short options
too.

ESC[1m-aESC[22m, ESC[1m--allESC[0m
do not ignore entries starting with .

ESC[1m-AESC[22m, ESC[1m--almost-allESC[0m
do not list implied . and ..
```

(рис. (fig:007?)).



```
1 #!/bin/bash
2
3 am$1
4 for ((i=0; i<$a; i++))
5 do
6     {{char=${RANDOM%256}}}
7     case $char in
8         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;;
9         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;; 18) echo -n r;;
10        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;;
11        26) echo -n z;;
12    esac
13 done
14 echo
15
```

Рис. 7: Название рисунка

Спасибо за внимание!

