

Лабораторная работа 12 Программирование в командном процессоре ОС UNIX. Расширенное программирование

Хусаинова Динара Айратовна

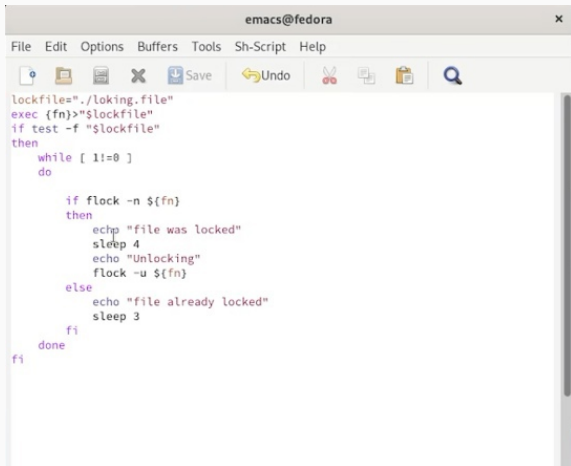
25.05.2022

RUDN

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

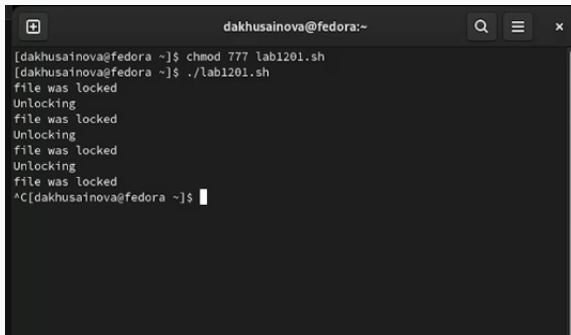
Упрощённый механизм семафоров

Напишем командный файл, реализующий упрощённый механизм семафоров(рис. 1,2).

The image shows a screenshot of the Emacs text editor window titled 'emacs@fedora'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The toolbar contains icons for opening files, saving, undo, redo, and search. The main text area displays a shell script for a simplified semaphore mechanism. The script defines a lockfile, uses 'exec' to acquire it, and then enters a loop where it checks if the file is locked. If locked, it prints a message and sleeps for 4 seconds before attempting to unlock. If not locked, it prints a message and sleeps for 3 seconds. The script uses 'flock' for file locking and 'echo' for output.

```
lockfile="./locking.file"
exec {fn}>"$lockfile"
if test -f "$lockfile"
then
  while [ 1!=0 ]
  do
    if flock -n ${fn}
    then
      echo "file was locked"
      sleep 4
      echo "Unlocking"
      flock -u ${fn}
    else
      echo "file already locked"
      sleep 3
    fi
  done
fi
```

Figure 1: Открываем редактор

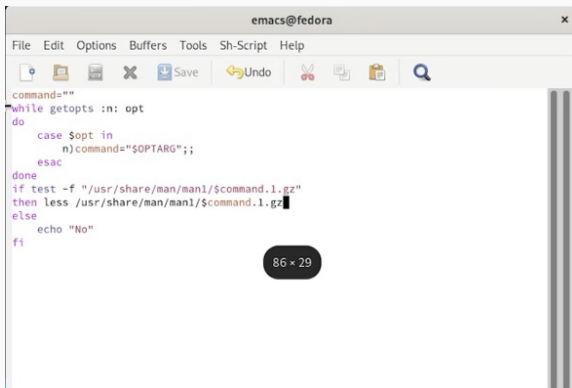
A terminal window titled 'dakhusainova@fedora:~' with search, menu, and close icons in the title bar. The terminal shows the execution of 'chmod 777 lab1201.sh' followed by './lab1201.sh'. The script outputs a series of 'file was locked' and 'Unlocking' messages, and the session ends with a Ctrl-C signal.

```
[dakhusainova@fedora ~]$ chmod 777 lab1201.sh
[dakhusainova@fedora ~]$ ./lab1201.sh
file was locked
Unlocking
file was locked
Unlocking
file was locked
Unlocking
file was locked
^C[dakhusainova@fedora ~]$
```

Figure 2: Результат

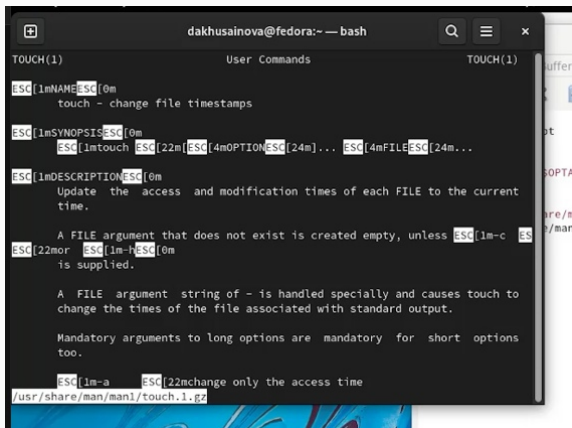
Реализация команды man с помощью командного файла

Реализуем команду man с помощью командного файла(рис. 3,4).



```
command=""
while getopts :n: opt
do
    case $opt in
        n) command="$OPTARG";;
    esac
done
if test -f "/usr/share/man/man1/$command.1.gz"
then less /usr/share/man/man1/$command.1.gz
else
    echo "No"
fi
```

Figure 3: Открываем редактор



```
dakhusainova@fedora:~ — bash
TOUCH(1)                                User Commands                                TOUCH(1)

ESC[1mNAMEESC[0m
    touch - change file timestamps

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

ESC[1mDESCRIPTIONESC[0m
    Update the access and modification times of each FILE to the current
    time.

    A FILE argument that does not exist is created empty, unless ESC[1m-c ESC[22mor
    ESC[1m-hESC[0m is supplied.

    A FILE argument string of - is handled specially and causes touch to
    change the times of the file associated with standard output.

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

    ESC[1m-a ESC[22mchange only the access time
/usr/share/man/man1/touch.1.gz
```

Figure 4: Результат

Генерация случайной последовательности букв

Используя встроенную переменную `$RANDOM`, напишем командный файл, генерирующий случайную последовательность букв латинского алфавита(рис. 5,6).

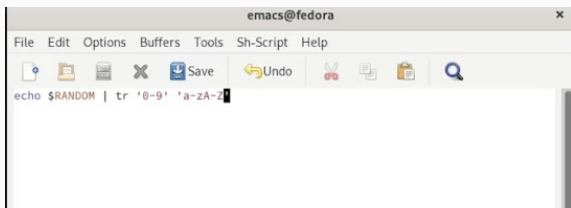


Figure 5: Генерация букв

```
[dakhusainova@fedora ~]$ chmod +x lab1203.sh
[dakhusainova@fedora ~]$ ./lab1203.sh
dcdff
[dakhusainova@fedora ~]$ ./lab1203.sh
ba tee
[dakhusainova@fedora ~]$ ./lab1203.sh
cg hhg
[dakhusainova@fedora ~]$ ./lab1203.sh
cfbaa
[dakhusainova@fedora ~]$ ./lab1203.sh
ifjf
[dakhusainova@fedora ~]$ ./lab1203.sh
dcgf
[dakhusainova@fedora ~]$ ./lab1203.sh
beedc
[dakhusainova@fedora ~]$ ./lab1203.sh
cgjjc
[dakhusainova@fedora ~]$ █
```

Figure 6: Результат

Мы изучили основы программирования в оболочке ОС UNIX и научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.