

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

Chvanova A.D.

NEC-2022, 24 May, Moscow

RUDN University, Moscow, Russian Federation

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

Написать 3 программных файла

Выполнение условного оператора `if` сводится к тому, что сначала выполняется последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `if`. Затем, если последняя выполненная команда из этой последовательности команд возвращает нулевой код завершения (истина), то будет выполнена последовательность команд (операторов), которую задаёт список-команд в строке, содержащей служебное слово `then`. Фраза `elif` проверяется в том случае, когда предыдущая проверка была ложной. Строка, содержащая служебное слово `else`, является необязательной.

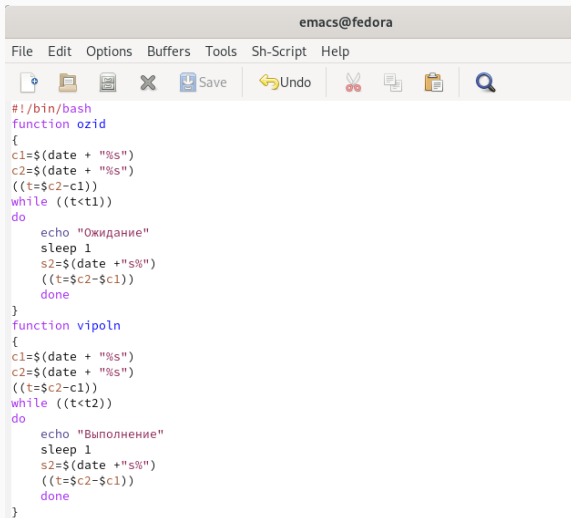
1. Написать командный файл, реализующий упрощённый механизм семафоров. Командный файл должен в течение некоторого времени t_1 дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени $t_2 < t_1$, также выдавая информацию 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 имела возможность взаимодействия трёх и более процессов.

Создаем файл , а также открываем emacs в фоновом режиме. Делаем файл исполняемым (рис. 1)

```
[adchvanova@fedora ~]$ touch pr1.sh
\[adchvanova@fedora ~]$ emacs &
[1] 62554
[adchvanova@fedora ~]$ chm
chmem chmod
[adchvanova@fedora ~]$ chmod +x pr1.sh
```

Figure 1: Создание файла, открытие emacs в фоновом режиме. Команда chmod +x, которая дает право на исполнение.

Пишем программу.(рис. 2 - 3)

The image shows a screenshot of the Emacs text editor running on a Fedora system. The title bar at the top reads 'emacs@fedora'. Below the title bar is a menu bar with 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. Underneath the menu bar is a toolbar with icons for opening files, saving, undo, redo, and search. The main editing area contains a shell script written in Bash. The script defines two functions: 'ozid' and 'vipoln'. Both functions calculate the time difference between two dates and enter a loop. The 'ozid' function prints 'Ожидание' (Waiting) and the 'vipoln' function prints 'Выполнение' (Execution). Both functions use 'sleep 1' and update the time difference variable in each iteration of the loop.

```
#!/bin/bash
function ozid
{
c1=$(date + "%s")
c2=$(date + "%s")
((t=$c2-c1))
while ((t<t1))
do
    echo "Ожидание"
    sleep 1
    s2=$(date + "%s")
    ((t=$c2-$s1))
done
}
function vipoln
{
c1=$(date + "%s")
c2=$(date + "%s")
((t=$c2-c1))
while ((t<t2))
do
    echo "Выполнение"
    sleep 1
    s2=$(date + "%s")
    ((t=$c2-$s1))
done
}
```

Figure 2: Скрипт программы

Выполнение лабораторной работы

```
t1=$1
t2=$2
com=$3
while true
do
    if [ "$com" == "Выход" ]
    then
        echo "Выход"
        exit 0
    fi

    if [ "$com" == "Ожидание" ]
    then
        echo ozid
    fi
    if [ "$com" == "Выполнение" ]
    then
        echo vipoln
    fi
    echo "Введите следующее действие"
    read com
done
```

Figure 3: Скрипт программы

Проверяем его работу. (рис. 4)

```
[adchvanova@fedora ~]$ ./pr1.sh 2 3 Ожидание /home/adchvanova  
ozid  
Введите следующее действие  
Выполнение  
v!poln  
Введите следующее действие  
Выход  
Выход  
[adchvanova@fedora ~]$
```

Figure 4: работа программы

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 или сообщение об отсутствии справки, если соответствующего файла нет в каталоге `man1`.

Создаем файл , а также открываем emacs в фоновом режиме. Делаем файл исполняемым (рис. 5)

```
[adchvanova@fedora ~]$ ./pr1.sh 2 3 Ожидание /home/adchvanova  
ozid  
Введите следующее действие  
Выполнение  
vipo1n  
Введите следующее действие  
Выход  
Выход  
[adchvanova@fedora ~]$
```

Figure 5: Создание файла, открытие emacs в фоновом режиме. Команда `chmod +x`, которая дает право на исполнение.

Пишем программу.(рис. 6)

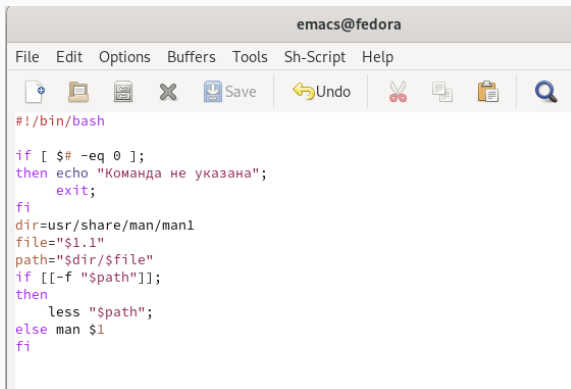
The image shows a screenshot of the Emacs text editor running on a Fedora system. The window title is 'emacs@fedora'. The menu bar includes 'File', 'Edit', 'Options', 'Buffers', 'Tools', 'Sh-Script', and 'Help'. The toolbar contains icons for creating a new file, opening a file, saving, undo, redo, and search. The main text area contains a shell script written in Bash. The script starts with a shebang line, followed by an if-then-else-fi block that checks if a command is provided. If not, it prints an error message and exits. If provided, it constructs a path to a man page and either displays it with 'less' or runs 'man'.

Figure 6: Скрипт программы

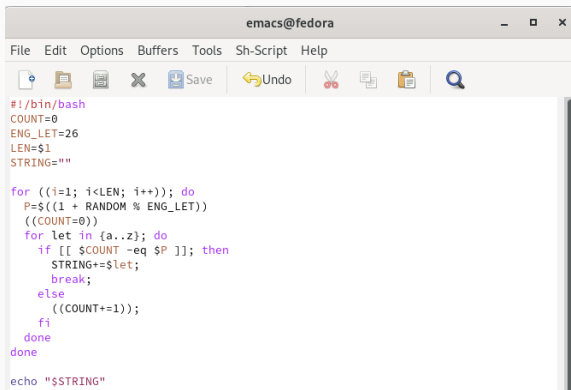
- Используя встроенную переменную `$RANDOM`, напишите командный файл, генерирующий случайную последовательность букв латинского алфавита. Учтите, что `$RANDOM` выдаёт псевдослучайные числа в диапазоне от 0 до 32767.

Создаем файл , а также открываем emacs в фоновом режиме. Делаем файл исполняемым (рис. 7)

```
[adchvanova@fedora ~]$ touch pr3.sh  
[adchvanova@fedora ~]$ emacs &  
[1] 64379  
[adchvanova@fedora ~]$ chmod +x pr3.sh
```

Figure 7: Создание файла, открытие emacs в фоновом режиме. Команда `chmod +x`, которая дает право на исполнение.

Пишем программу.(рис. 8)

The image shows a screenshot of the Emacs text editor window. The title bar at the top reads "emacs@fedora". Below the title bar is a menu bar with the following items: "File", "Edit", "Options", "Buffers", "Tools", "Sh-Script", and "Help". Underneath the menu bar is a toolbar containing icons for opening a file, saving a file, undo, redo, and search. The main editing area contains a shell script written in a syntax-highlighted format. The script starts with a shebang line, followed by variable declarations for COUNT, ENG_LET, LEN, and STRING. It then enters a loop that iterates from 1 to LEN. Inside the loop, it generates a random character from the set of lowercase letters (a-z) and appends it to the STRING variable. The loop ends with a 'done' statement, and the script concludes with an 'echo' command to display the final STRING.

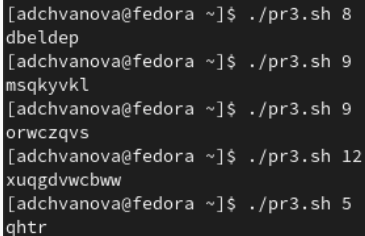
```
#!/bin/bash
COUNT=0
ENG_LET=26
LEN=$1
STRING=""

for ((i=1; i<LEN; i++)); do
    P=$((1 + RANDOM % ENG_LET))
    ((COUNT=0))
    for let in {a..z}; do
        if [[ $COUNT -eq $P ]]; then
            STRING+=$let;
            break;
        else
            ((COUNT+=1));
        fi
    done
done

echo "$STRING"
```

Figure 8: Скрипт программы

Проверяем его работу. (рис. 9)



```
[adchvanova@fedora ~]$ ./pr3.sh 8
dbeldep
[adchvanova@fedora ~]$ ./pr3.sh 9
msqkyvkl
[adchvanova@fedora ~]$ ./pr3.sh 9
orwczqvs
[adchvanova@fedora ~]$ ./pr3.sh 12
xuqgdvwcbww
[adchvanova@fedora ~]$ ./pr3.sh 5
qhtr
```

Figure 9: работа программы

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

Спасибо за внимание!
