

Программирование в командном процессоре ОС UNIX. Расширенное программирование

Лабораторная работа №12

Паулу А. Ж.

Российский университет дружбы народов, Москва, Россия

Информация

- Паулу Антонью Жоау
- студент 1 курса, группа НММбд-02-22
- Российский университет дружбы народов



Вводная часть

- Командный процессор ОС UNIX
- Командные файлы

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

- Ознакомиться с теоретическим материалом.
- Выполнить упражнения.
- Ответить на контрольные вопросы.

Выполнение лабораторной работы №12

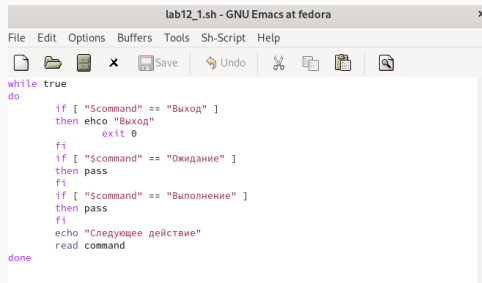
Первая программа

```
[azpaulu@fedora ~]$ touch lab12_1.sh
```

```
Открыть ▾  lab12_1.sh
#!/bin/bash
t1=$1
t2=$2
s1=$(date +%s)
s2=$(date +%s)
((t-$s2-$s1))
while ((t < t1)) do
    echo "Ожидаете"
    sleep 1
    s2=$(date +%s)
    ((t-$s2-$s1))
done
s1=$(date +%s)
s2=$(date +%s)
((t-$s2-$s1))
while ((t < t2)) do
    echo "Выполнение"
    sleep 1
    s2=$(date +%s)
    ((t-$s2-$s1))
done
t1=$1
t2=$2
command=$3
while true
do
    if [ "$command" == "Выход" ]
    then echo "Выход"
        exit 0
    fi
```

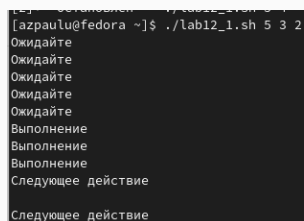
```
[azpaulu@fedora ~]$ ./lab12_1.sh 3 4
Ожидаете
Ожидаете
Ожидаете
Выполнение
Выполнение
Выполнение
Выполнение
```

Первая программа доработка



```
lab12_1.sh - GNU Emacs at fedora
File Edit Options Buffers Tools Sh-Script Help
[Icons: New, Open, Save, Undo, Cut, Copy, Paste, Find]

while true
do
    if [ "$command" == "Выход" ]
    then echo "Выход"
        exit 0
    fi
    if [ "$command" == "Ожидание" ]
    then pass
    fi
    if [ "$command" == "Выполнение" ]
    then pass
    fi
    echo "Следующее действие"
    read command
done
```



```
[azpaulu@fedora ~]$ ./lab12_1.sh 5 3 2
Ожидайте
Ожидайте
Ожидайте
Ожидайте
Выполнение
Выполнение
Выполнение
Следующее действие
Следующее действие
```

Вторая программа

```
[azpaulu@fedora ~]$ touch lab12_2.sh
```

```
Открыть ▾ ⓘ lab12_2.sh  
~  
  
#!/bin/bash  
c=$1  
if [ -f /usr/share/man/man1/${c}.1.gz ]  
then  
    gunzip -c /usr/share/man/man1/${c}.1.gz | less  
else  
    echo "Справка по данной команде нет"  
fi
```

Вторая программа


```
azpaulu@fedora ~]$ ./lab12_2.sh man
[4]+  Остановлен ./lab12_2.sh man
azpaulu@fedora ~]$ ./lab12_2.sh cd
[5]+  Остановлен ./lab12_2.sh cd
```

```
azpaulu@fedora:~$ ./bin/bash ./lab12_2.sh man
.\"
.\" * The above line should force tbl to be a preprocessor *
.\" Man page for man
.\"
.\" Copyright (C) 1994, 1995, Graeme W. Wilford. (Wilf.)
.\" Copyright (C) 2001-2019 Colin Watson.
.\"
.\" You may distribute under the terms of the GNU General Public
.\" License as specified in the file COPYING that comes with the
.\" man-db distribution.
.\"
.\" Sat Oct 29 13:09:31 GMT 1994 Wilf. (G.Wilford@surrey.ac.uk)
.\"
.DG
.TH MAN 1 "2022-02-04" "2.10.0" "Manual pager utils"
.SH NAME
man \- an interface to the system reference manuals
.SH SYNOPSIS
.\" The general command line
.B man
.RI [\i] "man options" [\i]
.RI [\i][\i] section [\i]
.RI page \ \i[\i]\i[\i]\i[\i] \ \i[\i]\i[\i]
```

```
azpaulu@fedora:~$ ./bin/bash ./lab12_2.sh cd
so man/builtins.1
```

Третья программа

```
[azpaulu@fedora ~]$ touch lab12_3.sh
```

```
Открыть ▾  lab12_3.sh  
~/  
#!/bin/bash  
k=$1  
for (( i=0; i<$k; i++ )) do  
    (( char=${RANDOM%26+1} ))  
    case $char in  
        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;;  
        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;;  
        19) echo -n s;;  
        20) echo -n t;;  
        21) echo -n u;;  
        22) echo -n v;;  
        23) echo -n w;;  
    esac  
done
```

```
[azpaulu@fedora ~]$ ./lab12_3.sh 7  
ku8nzfg  
[azpaulu@fedora ~]$ ./lab12_3.sh 9  
wedsgwjpp  
[azpaulu@fedora ~]$
```

3. Команда `seq` в Linux используется для генерации чисел от ПЕРВОГО до ПОСЛЕДНЕГО шага INCREMENT. Параметры: `seq LAST`: если задан только один аргумент, он создает числа от 1 до LAST с шагом шага, равным 1. Если LAST меньше 1, значение не выдает. `seq FIRST LAST`: когда заданы два аргумента, он генерирует числа от FIRST до LAST с шагом 1, равным 1. Если LAST меньше FIRST, он не выдает никаких выходных данных. `seq FIRST INCREMENT LAST`: когда заданы три аргумента, он генерирует числа от FIRST до LAST на шаге INCREMENT. Если LAST меньше, чем FIRST, он не производит вывод. `seq -f «FORMAT» FIRST INCREMENT LAST`: эта команда используется для генерации последовательности в форматированном виде. FIRST и INCREMENT являются необязательными. `seq -s «STRING» ПЕРВЫЙ`

5. Отличия командной оболочки `zsh` от `bash`:

В `zsh` более быстрое автодополнение для `cd` с помощью `Tab`. В `zsh` существует калькулятор `zcalc`, способный выполнять вычисления внутри терминала. В `zsh` поддерживаются числа с плавающей запятой. В `zsh` поддерживаются структуры данных «хэш». В `zsh` поддерживается раскрытие полного пути на основе неполных данных. В `zsh` поддерживается замена части пути. В `zsh` есть возможность отображать разделенный экран, такой же как разделенный экран `vim`.

6. `for((a=1; a<= LIMIT; a++))` синтаксис данной конструкции верен, потому что, используя двойные круглые скобки, можно не писать `$` перед переменными.

Результаты

В ходе выполнения лабораторной работы были изучены основы программирования в оболочке ОС UNIX. Научились писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.