Отчёт по лабораторной работе №12

Дисциплина “Операционные системы”

Батова Ирина Сергеевна, НММбд-01-22

26 апреля 2023

Российский университет дружбы народов, Москва, Россия

Вводная часть

Цель работы

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

Основная часть

Программа 1

- В данном скрипте мы вводим как переменные время ожидания и время выполнения (вводятся пользователем при запуске командного файла), а также два счетчика времени и еще изменяемые счетчик (разница двух предыдущих счетчиков). Далее мы пишем два цикла while - для ожидания и для выполнения. Внутри каждого из циклов мы выводим соответствующее сообщение и делаем паузу в 1 секунду для занесения изменений в счетчик. Между циклами мы обновляем все три счетчика для корректной работы второго цикла.

Программа 1

```
#!/bin/bash

ti=$1
me=$2
s1=$(date +"%s")
s2=$(date +"%s")
((t=$s2-$s1))

while ((t<ti))
do
    echo "Ожидание"
    sleep 1
    s2=$(date +"%s")
    ((t=$s2-$s1))
done

s1=$(date +"%s")
s2=$(date +"%s")
((t=$s2-$s1))

while ((t<me))
do
    echo "Выполнение"
    sleep 1
    s2=$(date +"%s")
    ((t=$s2-$s1))
done
```

Проверка программы 1

- Далее добавляем право на выполнение файла командой ’chmod +x *.sh’ и выполняем скрипт командой ‘./file31.sh (аргументы)’.

```
[isbatova@fedora ~]$ chmod +x *.sh
[isbatova@fedora ~]$ ./file31.sh 7 2
Ожидание
Ожидание
Ожидание
Ожидание
Ожидание
Ожидание
Ожидание
Выполнение
```

Измененная программа 1

- В измененном скрипте мы заносим обновление счетчиков и циклы while под функции, и вводим три переменных - время ожидания, время выполнения и переменную-указание к действию. Далее под циклом while true рассматриваем три варианта значения переменной-указания к действию с помощью if и обращаемся к соответствующей функции (или осуществляем выход). В конце выводим предложение ввести следующей действие и осуществляем аналогичные действия.

Измененная программа 1

```
#!/bin/bash

function waiting
{
s1=$(date +"%s")
s2=$(date +"%s")
((t=$s2-$s1))

while ((t<ti))
do
    echo "Ожидание"
    sleep 1
    s2=$(date +"%s")
    ((t=$s2-$s1))
done
}

function todo
{
s1=$(date +"%s")
s2=$(date +"%s")
((t=$s2-$s1))

while ((t<me))
do
    echo "Выполнение"
    sleep 1
    s2=$(date +"%s")
    ((t=$s2-$s1))
done
}
```

Измененная программа 1

```
ti=$1
me=$2
com=$3
while true
do
    if [ $com == Выход ]
    then
        echo "Выход"
        exit 0
    fi
    if [ $com == Ожидание ]
    then waiting
    fi

    if [ $com == Выполнение ]
    then todo
    fi

    echo "Введите следующее действие:"
    read command

done
```

Проверка измененной программы 1

- Далее выполняем скрипт командой ‘./file31.sh (аргументы)’. Программа работает корректно

```
[isbatova@fedora ~]$ ./file31.sh 1 2 Ожидание > /dev/tty
Ожидание
Введите следующее действие:
```

Программа 2

- В данном скрипте мы вводим переменную, которая принимает значение, введенное пользователем при запуске командной файла (название команды). Далее мы проверяем, есть ли информация по данной команде и с помощью if выводим информацию по введенной пользователем команде или сообщение, что информация по данной команде отсутствует.

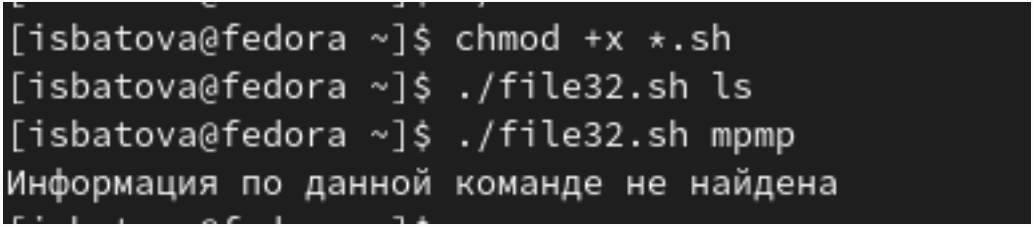
Программа 2

```
#!/bin/bash

a=$1
if [ -f /usr/share/man/man1/$a.1.gz ]
then
    gunzip -c /usr/share/man/man1/$a.1.gz | less
else
    echo "Информация по данной команде не найдена"
fi
```

Проверка программы 2

- Далее добавляем право на выполнение файла командой ’chmod +x *.sh’ и выполняем скрипт командой ‘./file32.sh (аргумент)’. Программа работает корректно как при введении названия существующей команды, так и не существующей команды



Программа 3

- В данном скрипте мы вводим переменную, которая принимает значение, введенное пользователем при запуске командной файла (количество символов). Далее с помощью цикла for мы выводим нужное количество символов. Внутри цикла используется встроенная переменная $RANDOM для определения случайного номера и команда case для непосредственного вывода символа (каждая команда вывода символа обозначается под своим порядковым номером, который и выбирает встроенная переменная $RANDOM).

Программа 3

```
#!/bin/bash
a=$1
for (( i=0; i<$a; i++))
do
    ((b=$RANDOM%26+1))
    case $b 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;; 24) echo -n x;;
        25) echo -n y; 26) echo -n z;;
    esac
done
echo
```

- Далее добавляем право на выполнение файла командой ’chmod +x *.sh’ и выполняем скрипт командой ‘./file33.sh (аргументы)’. Для проверки корректности выполнения вводим несколько чисел

```
[isbatova@fedora ~]$ chmod +x *.sh
[isbatova@fedora ~]$ ./file33.sh 6
ulphej
[isbatova@fedora ~]$ ./file33.sh 23
lumazndwutloqaahoxrrgqq
```

Вывод

Вывод

В ходе данной лабораторной работы мной были изучены основы программирования в оболочке ОС UNIX. Помимо этого, я научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов.