

# Российский университет дружбы народов

## Факультет физико-математических и естественных наук

---

### Отчёт по лабораторной работе №13

#### Программирование в командном процессоре ОС UNIX. Расширенное программирование

**Дисциплина:** Операционные системы

**Студент:** Оразгелдиева Огулнур

**Группа:** НПИбд-02-20

**Студ. номер:** 1032205431

2021, Москва

---

#### Лабораторная работа №13

##### Программирование в командном процессоре ОС UNIX. Расширенное программирование

###### Цель:

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

###### Задачи:

1. Ознакомиться с теоретическим материалом
2. Написать командный файл, реализующий упрощённый механизм семафоров. Запустить командный файл в одном виртуальном терминале в фоновом режиме, перенаправив его вывод в другой, в котором также запущен этот файл, но не фоновом, а в привилегированном режиме.
3. Реализовать команду man с помощью командного файла. Изучить содержимое каталога /usr/share/man/man1. 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.

4. Используя встроенную переменную *RANDOM*, написать командный файл, генерирующий случайную последовательность буквлат. *RANDOM* выдаёт псевдослучайные числа в диапазоне от 0 до 32767)
- 

### Теоретические сведения [1]

Командный процессор (командная оболочка, интерпретатор команд shell) — это программа, позволяющая пользователю взаимодействовать с операционной системой компьютера.

Bash (Bourne again shell или «возрождённый» shell) – это модифицированная версия программной оболочки Bourne-shell (sh или «Оболочка Борна»). Она является командным процессором, работающим интерактивно в текстовом окне. Bash нужен для приема команд пользователя и их отправки операционной системе для последующей обработки.

Взаимодействие оболочки и операционной системы обеспечивается с помощью специальной программы – терминала.

Командный процессор bash обеспечивает возможность использования переменных типа строка символов. Имена переменных могут быть выбраны пользователем. Пользователь имеет возможность присвоить переменной значение некоторой строки символов.

Использование значения, присвоенного некоторой переменной, называется подстановкой. Для того чтобы имя переменной не сливалось с символами, которые могут следовать за ним в командной строке, при подстановке в общем случае используется следующая форма записи: \${имя переменной}

Оболочка bash позволяет работать с массивами. Для создания массива используется команда set с флагом -A. [1]

test — UNIX-утилита для проверки типа файла и сравнения значений.

test -f file — истина, если file существует и является обычным файлом. [2]

Особенность less заключается в том, что команда не считывает текст полностью, а загружает его небольшими фрагментами.

Перечень всех опций и внутренних команд можно просмотреть в терминале, выполнив команду *man less*.

Использование опций не является обязательным. Открыть файл можно, выполнив следующую команду: *less filename.txt*

Командная строка исчезнет, а в окне терминала откроется указанный вами документ. [3]

---

**Ход работы:**

1. Сначала создаём командный файл *lab13-1.sh* для задания 1 с помощью текстового редактора vi.

Напишем в нём программу (см. рис. 1), по которой в течение некоторого времени  $t_1$  (в нашем случае 3 секунды) дожидаться освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использовать его в течение некоторого времени  $t_2 < t_1$  (5 секунд), также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом).

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
#!/bin/bash  
exec {fn}>./lockfile  
while test -f ./lockfile  
do  
if flock -n ${fn}  
then  
echo "Locked"  
sleep 3  
flock -u ${fn}  
echo "Unlocked"  
sleep 5  
else  
echo "Error"  
sleep 3  
fi  
done
```

Рисунок 1. Код программы для задания 1

**Пояснения:** сначала файлу нужно задать номер (flock работает с дескрипторами), далее идет цикл while: пока файл существует (test -f ./lockfile), проверка заблокирован/разблокирован ли файл в течение некоторого времени. Если условие не выполняется, то выдаётся ошибка, так как файл может выполняться другим терминалом.

Даём права на выполнение командного файла. (см. рис. 2)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab13-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab13-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ █
```

Рисунок 2. Права на выполнение

Теперь запускаем командный файл.

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab13-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab13-1.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-1.sh  
Locked  
Unlocked  
Locked  
Unlocked  
Locked
```

Рисунок 3. Выполнение командного файла

Когда процесс заблокирован выводится сообщение "Locked", когда разблокирован: "Unlocked" (см. рис. 3), а если он используется другим терминалом, то выдается "Error". (см. рис. 4) Когда запущено несколько терминалов для выполнения командного файла, когда в одном процесс разблокирован, в других выводится сообщение об ошибке.

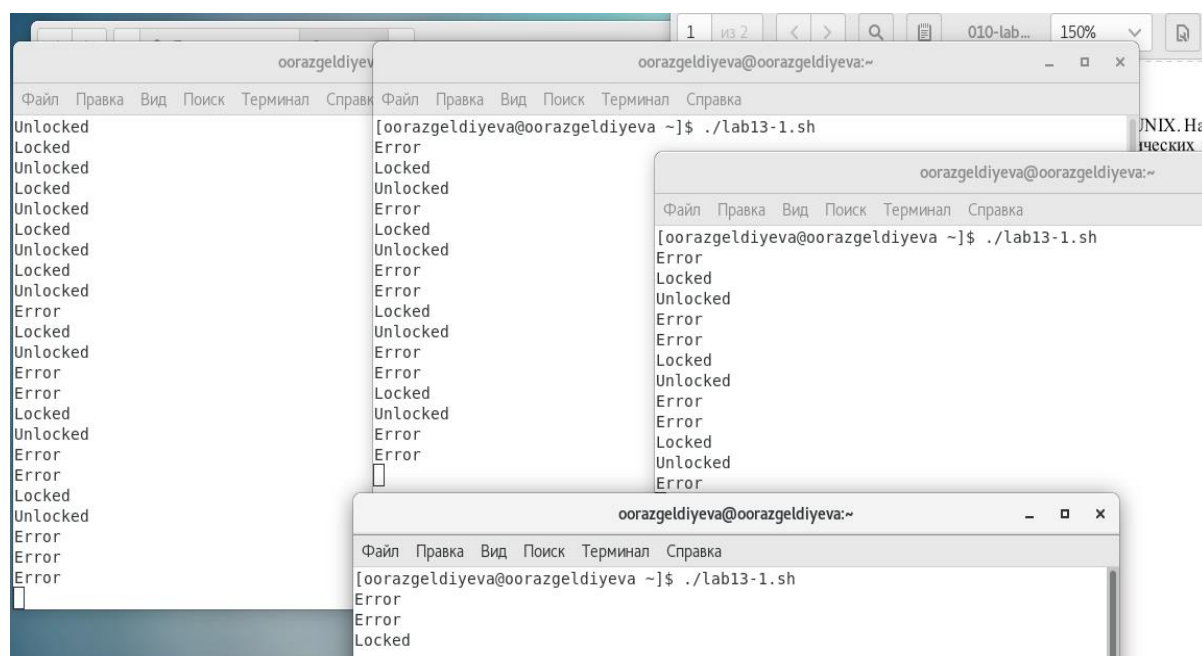


Рисунок 4. Выполнение командного файла несколькими терминалами

Когда запущено несколько терминалов для выполнения командного файла, когда в одном процесс разблокирован, в других выводится сообщение об ошибке.

2. Создаём и открываем командный файл *lab13-2.sh* для выполнения задания 2 с помощью редактора *vi*.

Нажав "i", переходим в режим вставки и пишем код программы, который реализует команду man. (с. рис. 5)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
#!/bin/bash  
cd /usr/share/man/man1  
name=""  
echo "Enter name of the command "  
read name  
less $name*  
~  
~  
~
```

Рисунок 5. Код программы для задания 2

**Пояснения:** так как в каталоге `/usr/share/man/man1` находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд, переходим в этот каталог, используя команду `cd`. После ввода названия команды, открываем архив со справкой об этой команде с помощью `less` в текущем каталоге (`/usr/share/man/man1`).

Даём права на выполнение командного файла и запускаем его. Введём, например, команду `mkdir` для справки. (см. рис. 6-7)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл  Правка  Вид  Поиск  Терминал  Справка  
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab13-2.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-2.sh  
bash: ./lab13-2.sh: Отказано в доступе  
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab13-2.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-2.sh  
Enter name of the command  
mkdir
```

Рисунок 6. Выполнение командного файла

```
Приложения  Места  Терминал
oorazgeldiyeva@oorazgeldiyeva:~
Файл  Правка  Вид  Поиск  Терминал  Справка
MKDIR(1)                                     User Commands                                     MKI

ESC[1mNAMEESC[0m
    mkdir - make directories

ESC[1mSYNOPSISESC[0m
    ESC[1mmkdir ESC[22m[ESC[4mOPTIONESC[24m]... ESC[4mDIRECTORYESC[24m]

ESC[1mDESCRIPTIONESC[0m
    Create the DIRECTORY(ies), if they do not already exist.

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

    ESC[1m-mESC[22m, ESC[1m--modeESC[22m=ESC[4mMODEESC[0m
        set file mode (as in chmod), not a=rwx - umask

    ESC[1m-pESC[22m, ESC[1m--parentsESC[0m
        no error if existing, make parent directories as needed

    ESC[1m-vESC[22m, ESC[1m--verboseESC[0m
mkdir.1.gz
```

Рисунок 7. Справка о команде *mkdir*

3. Создаём и открываем командный файл *lab13-3.sh* для выполнения задания 3 с помощью редактора *vi*. (см. рис. 8)

```
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-2.sh
Enter name of the command
mkdir
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab13-3.sh
```

Рисунок 8. Создание командного файла

Напишем в нём программу, генерирующую случайную последовательность букв латинского алфавита. (см. рис. 9)

```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
#!/bin/bash  
a=$((RANDOM % 10))  
set -a alphabet  
alphabet=( [0]="a" [1]="b" [2]="c" [3]="d" [4]="e" [5]="f"  
           [6]="g" [7]="h" [8]="i" [9]="j" [10]="k" [11]="l"  
           [12]="m" [13]="n" [14]="o" [15]="p" [16]="q"  
           [17]="r" [18]="s" [19]="t" [20]="u" [21]="v"  
           [22]="w" [23]="x" [24]="y" [25]="z")  
for ((i=0; i<a; i++))  
do  
b=$((RANDOM % 26))  
echo ${alphabet[$b]}  
done  
  
~  
~
```

Рисунок 9. Код программы для задания 3

**Пояснения:** сначала задаем переменную, которая будет содержать случайную длину (количество символов) от 1 до 10 символов (не включая 10, т.е. наибольшее 9); далее задаем массив *alphabet* с помощью *set -a*. Вносим в массив элементы - буквы латинского алфавита. Затем идёт создаём цикл *for*, который записывает случайные буквы (*RANDOMRANDOM % 10* - это значит случайные числа от 1 до 10). Даём права на выполнение и запускаем командный файл. (см рис. 10-11)

```
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab13-3.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ chmod +x lab13-3.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$
```

Рисунок 10. Права на выполнение

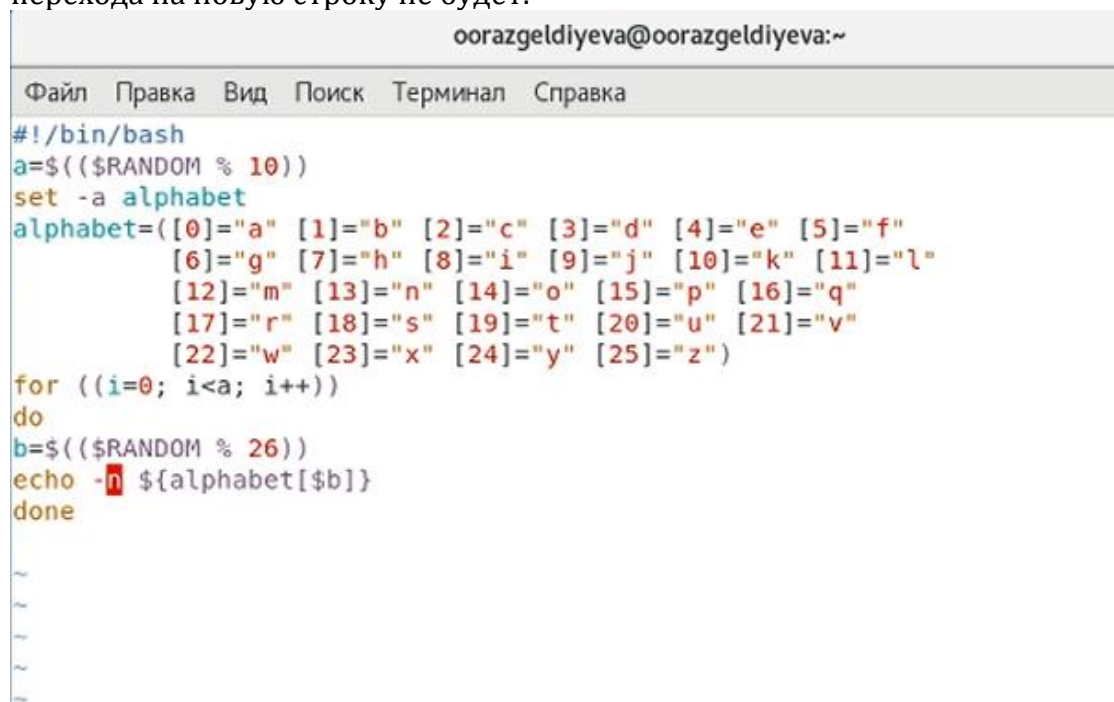
```
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab13-3.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
z  
z  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
k  
h  
e  
y  
c  
t  
b  
q  
x
```

Рисунок 11. Выполнение командного файла

Как видно из рис. 11, в результате выполнения программы выводятся от 1 до 10 случайных букв латинского алфавита, но каждая буква в отдельной строке.



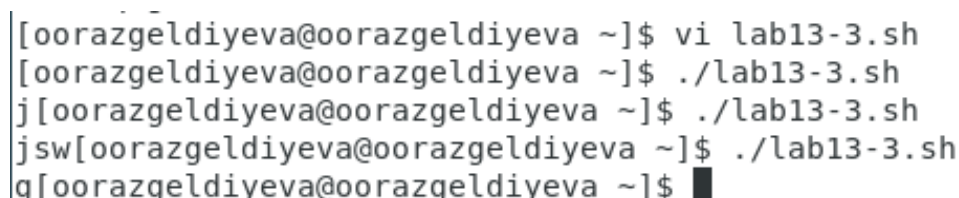
Чтобы исправить, добавим в код программы опцию `-n` к команде `echo`. (см. рис. 12). С помощью этой опции можно вывести буквы в одной строке, т.е после вывода буквы перехода на новую строку не будет.



```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
#!/bin/bash  
a=$((RANDOM % 10))  
set -a alphabet  
alphabet=([0]="a" [1]="b" [2]="c" [3]="d" [4]="e" [5]="f"  
          [6]="g" [7]="h" [8]="i" [9]="j" [10]="k" [11]="l"  
          [12]="m" [13]="n" [14]="o" [15]="p" [16]="q"  
          [17]="r" [18]="s" [19]="t" [20]="u" [21]="v"  
          [22]="w" [23]="x" [24]="y" [25]="z")  
for ((i=0; i<a; i++))  
do  
b=$((RANDOM % 26))  
echo -n ${alphabet[$b]}  
done  
  
~  
~  
~  
~  
~
```

Рисунок 12. Код программы для задания 3

Выполним командный файл. (см. рис. 13)



```
[oorazgeldiyeva@oorazgeldiyeva ~]$ vi lab13-3.sh  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
j[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
jsw[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
q[oorazgeldiyeva@oorazgeldiyeva ~]$ █
```

Рисунок 13. Выполнение командного файла

Как видно из рис. 13, случайные буквы в одной строке. После использования опции не было переходов в новую строку, поэтому, вывод программы "слился" с командной строкой.

Чтобы исправить это, в код программы сделаем следующие изменения: уменьшим число шагов (итераций на 1). Теперь у нас при выполнении цикла выводится на одну букву меньше, поэтому после выполнения цикла `for` еще раз выводим случайную букву с помощью `echo`, но на это раз без использования `-n`. (см. рис. 14)



```
oorazgeldiyeva@oorazgeldiyeva:~  
Файл Правка Вид Поиск Терминал Справка  
#!/bin/bash  
a=$(( $RANDOM % 10 ))  
set -a alphabet  
alphabet=( [0]="a" [1]="b" [2]="c" [3]="d" [4]="e" [5]="f"  
           [6]="g" [7]="h" [8]="i" [9]="j" [10]="k" [11]="l"  
           [12]="m" [13]="n" [14]="o" [15]="p" [16]="q"  
           [17]="r" [18]="s" [19]="t" [20]="u" [21]="v"  
           [22]="w" [23]="x" [24]="y" [25]="z" )  
for ((i=1; i<a; i++))  
do  
b=$(( $RANDOM % 26 ))  
echo -n ${alphabet[$b]}  
done  
b=$(( $RANDOM % 26 ))  
echo ${alphabet[$b]}  
~  
~  
~
```

Рисунок 14. Код программы для задания 3

Выполняем программу. (см. рис. 15)

```
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
gbhdsbw  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
b  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
bfyjwdks  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
k  
[oorazgeldiyeva@oorazgeldiyeva ~]$  
[oorazgeldiyeva@oorazgeldiyeva ~]$ ./lab13-3.sh  
xruznqig  
[oorazgeldiyeva@oorazgeldiyeva ~]$ █
```

Рисунок 15. Выполнение командного файла

Теперь буквы выводятся в одной строке, не "сливаясь" при этом с командной строкой. Строка случайной длины (до 10 букв) состоит из случайных букв латинского алфавита.

---

**Вывод:** на лабораторной работе изучила основы программирования в оболочке ОС UNIX и научилась писать более сложные командные файлы с использованием логических управляющих конструкций и циклов

---

## **Библиография**

[1] - РУДН, Операционные системы, "Программирование в командном процессоре ОС UNIX. Командные файлы"

[2] - Утилита test

[3] - Команда less в Linux

---