

# **РУДН. Операционные системы**

**Отчёт по лабораторной работе №11**

Косинов Никита Андреевич, НПИМбв-02-20

# Содержание

1	Цель работы	5
2	Ход работы	6
3	Написание файла-“семафор”	7
4	Реализация команды <i>tap</i>	8
5	Написание командного файла, создающего случайную строку	10
6	Выводы	12

## Список иллюстраций

4.1	Код командного файла . . . . .	8
4.2	Результат выполнения командного файла . . . . .	8
4.3	Результат выполнения командного файла . . . . .	9
4.4	Результат выполнения командного файла . . . . .	9
5.1	Код командного файла . . . . .	10
5.2	Результат выполнения командного файла . . . . .	11

## **Список таблиц**

# 1 Цель работы

Основой пользования ЭВМ и его работы являются программы - блоки последовательно выполняемых простейших команд. Мы сталкивались ранее с написанием простейших команд и конвейеров в терминале. Но что делать, если нам нужно выполнить множество одинаковых, или зависящих от условия команд, или чтобы они выполнялись автоматически?

Оболочка ОС **Linux** позволяет базово программировать прямиком в терминале и даже сохранять блоки команд в текстовых, но исполняемых файлах. Цель данной работы - познакомиться с основами, предлагаемыми терминалом **Linux** для программирования.

## 2 Ход работы

Лабораторная работа выполнена в терминале **ОС Linux**, командная оболочка **bash** и хостинге хранения проектов **Github**. Действия по лабораторной работе представлены в следующем порядке:

1. написание файла-“семафор”;
2. реализация команды *map*;
3. написание командного файла, создающего случайную строку;

По завершении отчёта, вся рабочая папка отправляется на репозиторий на *github*.

## 3 Написание файла-“семафор”

Перед началом работы создадим новый рабочий каталог **lab09** и перейдём внутрь. Также не забываем синхронизироваться с нашим **Git**.

## 4 Реализация команды *man*

1. Создаём файл *MyMan*.
2. Пишем туда следующий код:

```
1 x=/usr/share/man/man1/$1.1.bz2
2 if [ ! -f "$x" ]
3 then
4     echo Not command $1
5 else
6     less $x
7 fi
```

Рис. 4.1: Код командного файла

3. Проверяем результат с существующей командой.

```
nakosinov@dk8n59 ~/work/study/2022-2023/Операционные системы/os-intro/labs/lab11 $ bash
MyMan 1rx
```

Рис. 4.2: Результат выполнения командного файла



```
RZ(1)                                General Commands Manual                                RZ(1)

NAME
    rx, rb, rz - XMODEM, YMODEM, ZMODEM (Batch) file receive

SYNOPSIS
    rz [- +8abeOpqRtTuUvy]
    rb [- +abqRtuUvy]
    rx [- abceqRtuUv] file
    [-][v]rzCOMMAND

DESCRIPTION
    This program uses error correcting protocols to receive files over a dial-in
    serial port from a variety of programs running under PC-DOS, CP/M, Unix, and
    other operating systems. It is invoked from a shell prompt manually, or au-
    tomatically as a result of an "sz file ..." command given to the calling pro-
    gram.

    While rz is smart enough to be called from cu(1), very few versions of cu(1)
    are smart enough to allow rz to work properly. Unix flavors of Professional-
    YAM are available for such dial-out application.

    Rz (Receive ZMODEM) receives files with the ZMODEM batch protocol. Pathnames
    /usr/share/man/man1/lrx.1.bz2 lines 1-23
```

Рис. 4.3: Результат выполнения командного файла

4. Проверяем результат с несуществующей командой.

```
nakosinov@dk8n59 ~/work/study/2022-2023/Операционные системы/os-intro/labs/lab11 $ bash
MyMan qwerty
Not command qwerty
nakosinov@dk8n59 ~/work/study/2022-2023/Операционные системы/os-intro/labs/lab11 $
```

Рис. 4.4: Результат выполнения командного файла

## 5 Написание командного файла, создающего случайную строку

1. Создаём файл *random*.
2. Пишем туда следующий код. Обратим внимание, что мы рассматриваем случайные числа от 0 до 32759, т.к. это делится на 26, и, следовательно, даёт равновероятное выпадение буквы.

```
1 x=$1
2 A=(a b c d e f g h i j k l m n o p q r s t u v w x y z)
3 B=''
4 ((x+=1))
5 while
6 ((x-=1))
7 do
8     y=$RANDOM
9     while
10    ((y>32759))
11    do
12        y=$RANDOM
13    done
14    B+=${A[$y%26]}
15 done
16 echo $B
```

Рис. 5.1: Код командного файла

3. Проверяем результат с несколькими вариантами введённого числа.

```
nakosinov@dk8n59 ~/work/study/2022-2023/Операционные системы/os-intro/labs/lab11 $ bash
random 10
mqzbxuibwr
nakosinov@dk8n59 ~/work/study/2022-2023/Операционные системы/os-intro/labs/lab11 $ bash
random 5
wiopc
nakosinov@dk8n59 ~/work/study/2022-2023/Операционные системы/os-intro/labs/lab11 $ bash
random 50
hnbzgaskslzoquxgwmfkrpjmlvvaovpkcvfbcxklkxfrgycug
nakosinov@dk8n59 ~/work/study/2022-2023/Операционные системы/os-intro/labs/lab11 $
```

Рис. 5.2: Результат выполнения командного файла

## 6 Выводы

Для работы с операционной и файловой системой очень помогает оболочка командной строки *bash*. При этом, оболочка позволяет сохранять блоки команд в единый программный файл, что сильно упрощает работу с ним.