

Программирование в командном процессоре ОС UNIX.

Плето Плето Мбамби¹

25 апреля, 2023, Москва, Россия

¹Российский Университет Дружбы Народов

Цели и задачи работы

Цель лабораторной работы

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

Задачи лабораторной работы

1 Выполнить 3 задания

Процесс выполнения лабораторной работы

1. Написали командный файл, реализующий упрощённый механизм семафоров. Командный файл в течение некоторого времени t_1 дожидается освобождения ресурса, выдавая об этом сообщение, а дождавшись его освобождения, использует его в течение некоторого времени $t_2 < t_1$, также выдавая информацию о том, что ресурс используется соответствующим командным файлом (процессом).

Выполнение работы

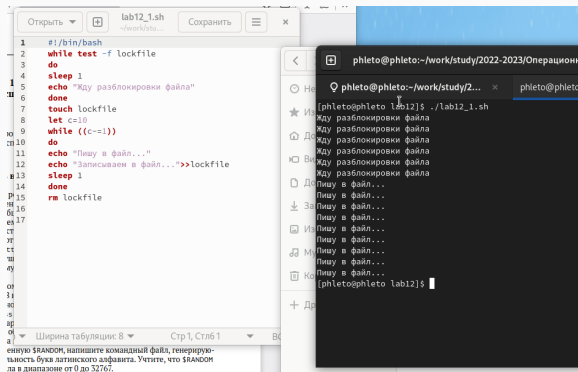
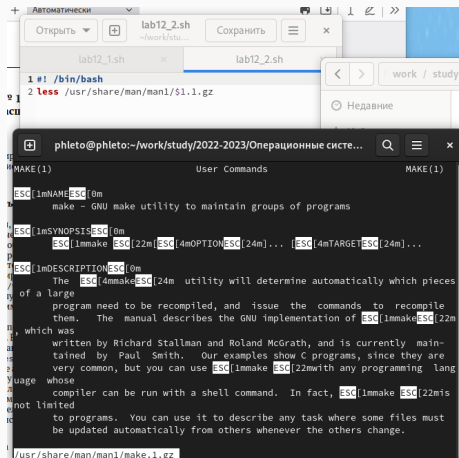


Рис. 1: Задание 1

2. Реализовали команду `man` с помощью командного файла. Изучили содержимое каталога `/usr/share/man/man1`. В нем находятся архивы текстовых файлов, содержащих справку по большинству установленных в системе программ и команд.

Выполнение работы



The screenshot shows a terminal window with two tabs: 'lab12_1.sh' and 'lab12_2.sh'. The 'lab12_2.sh' tab is active and contains the following commands:

```
1 #! /bin/bash
2 less /usr/share/man/man1/$1.1.gz
```

Below the terminal window, there is a larger window titled 'phleto@phleto:~/work/study/2022-2023/Операционные систе...'. It displays the output of the 'less' command for the file '/usr/share/man/man1/make.1.gz'. The output is as follows:

```
MAKE(1)                                User Commands                                MAKE(1)

make - GNU make utility to maintain groups of programs

SYNOPSIS
make [options] [target]...

DESCRIPTION
The make utility will determine automatically which pieces
of a large program need to be recompiled, and issue the commands to recompile
them. The manual describes the GNU implementation of make.

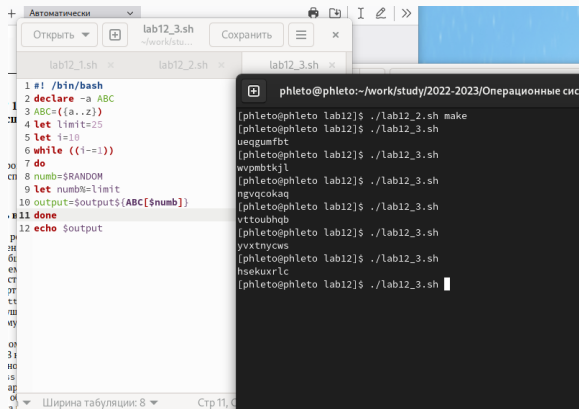
The make utility was written by Richard Stallman and Roland McGrath, and is currently main-
tained by Paul Smith. Our examples show C programs, since they are
very common, but you can use make with any programming lang-
uage whose compiler can be run with a shell command. In fact, make is
not limited to programs. You can use it to describe any task where some files must
be updated automatically from others whenever the others change.

/usr/share/man/man1/make.1.gz
```

Рис. 2: Задание 2

3. Используя встроенную переменную `$RANDOM`, написали командный файл, генерирующий случайную последовательность букв латинского алфавита

Выполнение работы



The screenshot shows a terminal window with a light blue header bar. The window title is "Автоматически" (Automatically). The tab bar shows three tabs: "lab12_1.sh", "lab12_2.sh", and "lab12_3.sh". The "lab12_3.sh" tab is active, showing a shell script. The script content is as follows:

```
1 #!/bin/bash
2 declare -a ABC
3 ABC=({a..z})
4 let limit=25
5 let i=10
6 while ((i-=1))
7 do
8   numb=$RANDOM
9   let numb%=limit
10  output=$output${ABC[$numb]}
11 done
12 echo $output
```

The terminal output shows the execution of the script. The prompt is "phleto@phleto:~/work/study/2022-2023/Операционные сис". The command executed is "./lab12_2.sh make". The output is a long string of characters: "ueqgumfhtwvpmbtjklngvqcokaqvttoabhqbyvxtnycwshsekuxrlc". The prompt is then "[phleto@phleto lab12]\$./lab12_3.sh".

Рис. 3: Задание 3

Выводы по проделанной работе

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