Министерство образования Республики Беларусь
учреждение образования
«Брестский государственный технический университет»
Кафедра ИИТ

Лабораторная работа № 3

«BASH: ПОТОКИ ДАННЫХ.
ПРОГРАММИРОВАНИЕ»»

Выполнил: студент 2-го курса
группы АС-65
Осовец М.М
Проверил: Степанчук В.И.

Брест 2024

Ход работы

Цель работы: научиться создавать скрипты на ОС Linux через Bash.

Задание для выполнения

1. Вывести любое сообщение с помощью команды echo перенаправив вывод:
- в несуществующий файл с помощью символа >;
- в несуществующий файл с помощью символа >>;
- в существующий файл с помощью символа >;
- в существующий файл с помощью символа >>;

Объяснить результаты.

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ echo "Hello, World!" > newfile.txt
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ echo "Hello, World!" >> anothernewfile.txt
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ echo "New message" > newfile.txt
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ echo "New message" >> anothernewfile.txt
```

Объяснение: создаёт файл, если его нет, и перезаписывает его содержимое. >> создаёт файл, если его нет, и добавляет данные в конец файла, если он существует.

2. Переадресовать стандартный ввод для команды cat на файл.

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ cat < newfile.txt
New message
```

3. Вывести сообщение с помощью команды echo в канал ошибок. Создать файл myscript:

```
#!/bin/sh
echo stdout
echo stderr>&2
exit 0
```

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ echo -e '#!/bin/sh\necho stdout\necho stderr >&2\nexit 0' > myscript
```

Запустить его:

- без перенаправления (sh myscript);

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript
stdout
stderr
```

перенаправив стандартный вывод в файл, просмотреть содержимое файла (sh myscript > file1);

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript > file1
stderr
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ cat file1
stdout
```

Обьяснение: Только стандартный вывод (stdout) будет записан в file1, а сообщение stderr останется в терминале, так как мы не перенаправляли стандартный поток ошибок. Содержимое файла file1:

- перенаправить стандартный канал ошибок в существующий и несуществующий файлы с помощью символов > и >> ;

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript > file2
stderr
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript >> file2
stderr
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ cat file2
stdout
stdout
```

Обьяснение: В file2 будет записан стандартный поток ошибок stderr, а стандартный вывод останется в терминале. Содержимое файла file2.

Для несуществующего файла

```
Выход из программы.
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript >> fole2
stderr
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript > fole2
stderr
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$
```

Обьяснение:Если newfile не существует, команда создаст его и запишет в него stderr.

- перенаправив стандартный вывод в файл 1, стандартный канал ошибок - в файл 2;

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript > file1 2> file2
```

Обьяснение: stdout записан в file1, а stderr — в file2.

- перенаправив стандартный вывод и стандартный канал ошибок в файл 3;

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript > file3 2>&1
```

Обьяснение:Оба вывода (и stdout, и stderr) записаны в file3

- перенаправив стандартный вывод в файл 4 с помощью символа >, а стандартный канал ошибок в файл 4 с помощью символа >>;

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sh myscript > file4 2>> file4
```

Обьяснение:stdout перезаписывает содержимое file4, а stderr добавляется в конец того же файла:

4. Вывести третью и шестую строку из последних пятнадцати строк отсортированного в обратном порядке файла /etc/group.

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ tail -n 15 /etc/group | sort -r | sed -n '3p;6p'
saned:x:116:
pipewire:x:118:
```

5. Подсчитать при помощи конвейера команд количество блочных и количество символьных устройств ввода-вывода, доступных в системе.

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ ls -l /dev | grep -E '^[bc]' | awk '{print $1}' | sort | uniq -c
     63 brw-rw----
     45 crw-------
      2 crw-r-----
      2 crw-r--r--
     60 crw-rw----
      4 crw-rw----+
      1 crw-rw-r--+
      8 crw-rw-rw-
     65 crw--w----
```

6. Написать скрипт, выводящий на консоль все аргументы командной строки, переданные данному скрипту. Привести различные варианты запуска данного скрипта, в том числе без непосредственного вызова интерпретатора в командной строке.

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ nano show_args.sh
```

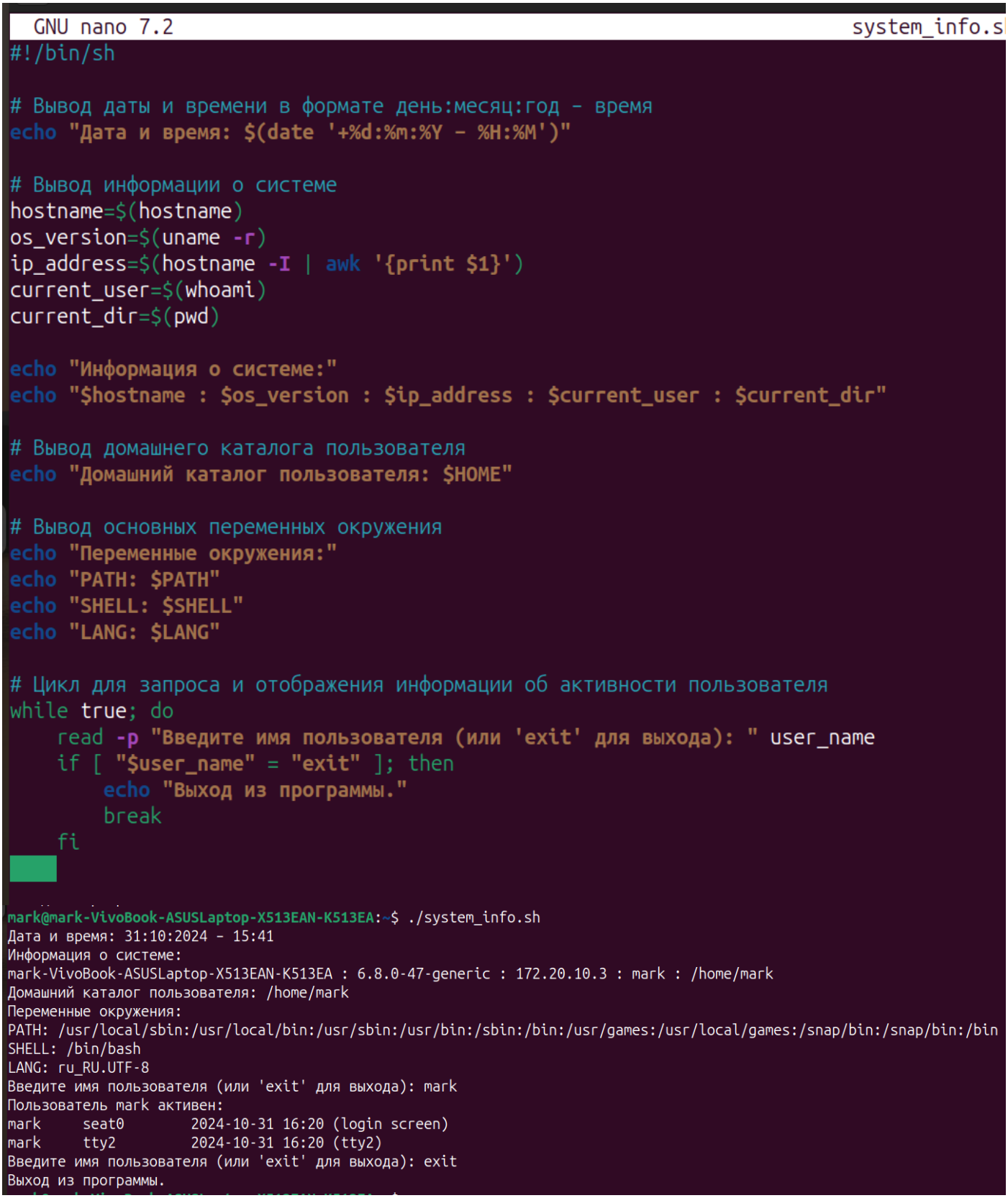
Скрипт:

```
  GNU nano 7.2
#!/bin/sh
echo "Аргументы командной строки:"
for arg in "$@"; do
    echo "$arg"
done
```

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ chmod +x show_args.sh
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ sudo mv show_args.sh /usr/local/bin/
[sudo] пароль для mark:
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ show_args.sh arg1 arg2 arg3
Аргументы командной строки:
arg1
arg2
arg3
```

7. Реализовать командный файл, который выводит: дату в формате день:месяц:год – время, информацию о системе в формате: имя компьютера : версия ОС : IP адрес : имя текущего пользователя : текущий каталог, выводит домашний каталог текущего пользователя и основные переменные окружения. Далее в цикле выводит список активных пользователей – запрашивает имя
пользователя и выводит информацию об активности введенного пользователя.

```
mark@mark-VivoBook-ASUSLaptop-X513EAN-K513EA:~$ nano system_info.sh
```



Вывод: на этой лабароторной работе мы научились создавать скрипты через Bash.