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

Лабораторная работа №3
По дисциплине «Операционные системы»
Тема: «BASH: ПОТОКИ ДАННЫХ. ПРОГРАММИРОВАНИЕ»

Выполнил:
Студент 2 курса
Группы ПО-11
Микулич М. И.
№ зачетки: 220216
Проверила:
Давидюк Ю. И.

Брест 2023

Вариант 9

Цель работы: Получить практические и теоретические знания о bash: потоки данных.

Ход работы

Часть 1.

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.

```
maksik@ugabuga:~$ echo "text nelpohoi">file-test-1
maksik@ugabuga:~$ echo "text normalny">>file-test-2
maksik@ugabuga:~$ touch file-test-3
maksik@ugabuga:~$ echo "text drugi">file-test-3
maksik@ugabuga:~$ touch file-test-4
maksik@ugabuga:~$ echo "text posledni">>file-test-4
maksik@ugabuga:~$
```

В первом случае (>) мы создадим файл с именем «file-test-1», который до этого момента не существовал, и записываем в него переданную строку. Во втором случае (>>) мы так же создаем новый файл с новым именем и записываем строку. Разница между первым и вторым вариантом можно сказать отсутствует, так как до этого файлы не существовали и символы «>/>>» значения не имеют.

В третьем случае (>) мы заранее создали новый файл, после чего записали переданную строку. В четвертом варианте (») мы так же сперва создали новый файл, после чего так же записали строку. Здесь уже разница есть, так как символ «>» перезаписал все строки (удалил все старые и добавил новую), а символ «>>» ничего не перезаписывает, а просто добавляет строку в конец файла. Проверить это можно при помощи повторения команды «echo», после чего вывести содержимое файлов:

```
maksik@ugabuga:~$ echo "NEW TEXT">file-test-3
maksik@ugabuga:~$ echo "new text 2">>file-test-4
maksik@ugabuga:~$ cat file-test-3
NEW TEXT
maksik@ugabuga:~$ cat file-test-4
text posledni
new text 2
```

2. Переадресовать стандартный ввод для команды cat на файл.

```
maksik@ugabuga:~$ cat << 'q'
> test
> test
> te
> q
test
test
te
```

3. Вывести сообщение с помощью команды echo в канал ошибок.

```
maksik@ugabuga:~$ echo "error string">&2
error string
```

Создать файл myscript:

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

```
maksik@ugabuga:~$ touch myscript
maksik@ugabuga:~$ cat myscript
#!/bin/sh
echo stdout
echo stderr>&2
exit 0
```

Запустить его:

- без перенаправления (sh myscript);

```
maksik@ugabuga:~$ sh myscript
stdout
stderr
```

Вводится при помощи echo stdout — в стандартный вывод, stderr — в канал ошибок.

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

```
maksik@ugabuga:~$ sh myscript > file1
stderr
maksik@ugabuga:~$ cat file1
stdout
```

Мы перенаправили стандартный вывод в файл при помощи «>», поэтому в него записалось stdout, а stderr вывелось в стандартный канал ошибок.

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

```
maksik@ugabuga:~$ sh myscript 2> Nfile1
stdout
maksik@ugabuga:~$ sh myscript 2>> Nfile2
stdout
maksik@ugabuga:~$ touch S-file 1 S-file2
maksik@ugabuga:~$ sh myscript 2> S-file1
stdout
maksik@ugabuga:~$ sh myscript 2>> S-file2
stdout
```

Содержимое файлов:

```
maksik@ugabuga:~$ cat Nfile1 Nfile2 S-file1 S-file2
stderr
stderr
stderr
stderr
```

В файлах записывается stderr, т.к. мы перенаправили стандартный канал ошибок, а во время выполнения стандартный вывод выводится stdout.

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

```
maksik@ugabuga:~$ sh myscript>file1
stderr
maksik@ugabuga:~$ sh myscript 2>file2
stdout
maksik@ugabuga:~$ cat file1
stdout
maksik@ugabuga:~$ cat file2
stderr
```

Перенаправив стандартный вывод в file1 в него запишется stdout, а в стандартный канал ошибок выведется stderr. По зеркальной схеме произойдет с file2 (то есть запишется stderr, выведется — stdout).

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

```
maksik@ugabuga:~$ sh myscript >file-3 2>file-3
maksik@ugabuga:~$ cat file-3
stderr
```

Сперва мы перенаправили стандартный вывод в file-3, поэтому в него запишется stdout. Затем мы перенаправили стандартный канал ошибок, поэтому в него запишется stderr, стерев прошлую запись.

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

```
maksik@ugabuga:~$ sh myscript >file-4 2>>file-4
maksik@ugabuga:~$ cat file-4
stdout
stderr
```

Все происходит точно так же, как и в примере выше. Однако из-за использования символа «>>» во втором в file-4 не происходит перезапись существующей строки, а добавляется еще одна строка в конец файла.

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

```
maksik@ugabuga:~$ sort -r /etc/group | tail -n 15 | sed -n '3p;6p'
dip:x:30:maksik
crontab:x:104:
```

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

```
maksik@ugabuga:~$ ls -l /dev | egrep '^b' | wc -l
23
maksik@ugabuga:~$ ls -l /dev | egrep '^c' | wc -l
196
```

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.

```
maksik@ugabuga:~$ ls -ld myscript-2
-rwxrwxrwx 1 maksik maksik 49 кас 19 11:24 myscrip
maksik@ugabuga:~$ cat myscript-2
#!/bin/sh
for arg in "$@";
do
  echo "$arg"
done
```

```
maksik@ugabuga:~$ ./myscript-2 arg-1 arg-2 arg-3
arg-1
arg-2
arg-3
maksik@ugabuga:~$ ./myscript-2 arg-1 arg-2 arg-3
arg-1
arg-2
arg-3
maksik@ugabuga:~$ sh myscript-2 arg-1 arg-2 arg-3
arg-1
arg-2
arg-3
```

Во втором варианте запуска из фала была удалена строка «#! /bin/sh», поэтому мы запустили данный скрипт без непосредственного вызова интерпретатора.

7. Написать скрипт согласно индивидуальному заданию. Номер варианта согласовать с преподавателем.

9.Реализовать командный файл, реализующий символьное меню (в цикле):

1) вывод полной информации о файлах каталога (ввести имя каталога для отображения);

2) создать командный файл: файл вводится с клавиатуры по запросу, далее изменяются атрибут файла на исполнение, затем вводится с клавиатуры строка, которую будет исполнять командный файл. После изменения атрибутов вывести на экран расширенный список файлов для проверки установленных атрибутов и запустить созданный командный файл;

3) завершение.

Исходный код меню:

Открыть | script-lab-3 ~/ | Сохранить

```
#!/bin/sh
while true; do
  echo "1) Вывести полную информацию о файлах каталога"
  echo "2) Создать и запустить командный файлй"
  echo "3) Закрыть меню"
  read -p "Введите номер пункта меню: " choice
  echo
  case $choice in
    1)
      read -p "Введите имя каталога: " directory
      echo
      ls -la "$directory"
      echo
      ;;
    2)
      read -p "Введите имя командного файла для создания: " script_name
      touch "$script_name"
      chmod +x "$script_name"
      read -p "Введите команду для выполнения из командного файла: " command
      echo "$command" >> "$script_name"
      echo "Файл $script_name был создан и команда добавлена в него."
      echo
      ls -la
      echo
      ./"$script_name"
      ;;
    3)
      exit 0
      ;;
    *)
      echo "Некорректный ввод"
      echo
      ;;
  esac
done

```

Исполнение меню:

```
maksik@ugabuga:~$ ./script-lab-3
1) Вывести полную информацию о файлах каталога
2) Создать и запустить командный файлй
3) Закрыть меню
Введите номер пункта меню: 1

Введите имя каталога: snap

итого 24
drwx------  6 maksik maksik 4096 кас  4 22:31 .
drwxr-x--- 17 maksik maksik 4096 кас 20 21:06 ..
drwxr-xr-x  4 maksik maksik 4096 вер 27 10:36 firefox
drwxr-xr-x  4 maksik maksik 4096 вер 27 10:35 snapd-desktop-integration
drwxr-xr-x  4 maksik maksik 4096 кас  4 22:14 snap-store
drwxr-xr-x  5 maksik maksik 4096 кас 20 20:31 spotify
```

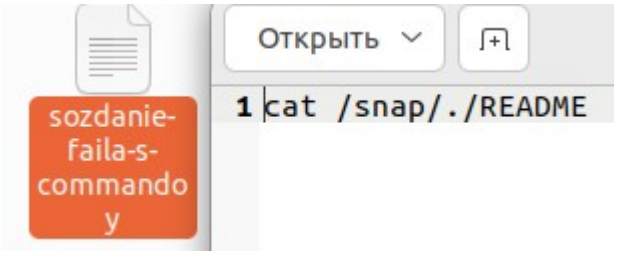
```
1) Вывести полную информацию о файлах каталога
2) Создать и запустить командный файлй
3) Закрыть меню
Введите номер пункта меню: 2

Введите имя командного файла для создания: sozdanie-faila-s-commandoy
Введите команду для выполнения из командного файла: cat /snap/./README
Файл sozdanie-faila-s-commandoy был создан и команда добавлена в него.

итого 124
```

```
-rwxrwxr-x  1 maksik maksik   19 кас 20 21:07  sozdanie-faila-s-commandoy
```

Здесь мы видим, что программа отработала корректно: файл создан, команда успешно записана, право на исполнение присвоено (ниже видно, что файл с исполнительной командой выполняется корректно и успешно).

```
This directory presents installed snap packages.

It has the following structure:

/snap/bin                     - Symlinks to snap applications.
/snap/<snapname>/<revision> - Mountpoint for snap content.
/snap/<snapname>/current    - Symlink to current revision, if enabled.

DISK SPACE USAGE

The disk space consumed by the content under this directory is
minimal as the real snap content never leaves the .snap file.
Snaps are *mounted* rather than unpacked.

For further details please visit
https://forum.snapcraft.io/t/the-snap-directory/2817
1) Вывести полную информацию о файлах каталога
2) Создать и запустить командный файлй
3) Закрыть меню
Введите номер пункта меню: 3

maksik@ugabuga:~$
```

Созданный файл с исполнительной командой:



Меню выполнило все свои описанные функции и завершило свою работу успешно.

Вывод: Получил практические и теоретические знания о bash: потоки данных.