Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОТЕХНИКИ

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

**ОТЧЁТ**

к лабораторной работе №1 на тему

**«УПРАВЛЕНИЕ ОС LINUX, ИНТЕРПРЕТАТОР BASH»**

по дисциплине «ОСМУ»

Выполнил студент: К.Г. Хоменок

Проверил: А.Д. Станкевич

Минск 2024

**Цель работы:** изучить основные объекты, команды, типы данных и операторы управления интерпретатора BASH; создать скрипт-файл.

**1 Краткие теоретические сведения**

**Bash** – это sh-совместимый интерпретатор командного языка, выполняющий команды, прочитанные со стандартного входного потока или из файла. Скрипт-файл – это обычный текстовый файл, содержащий последовательность команд bash, для которого установлены права на выполнение.

**2 Порядок выполнения работы**

1 Изучить теоретическую часть лабораторной работы.

2 В консольном режиме создать, используя команды из табл.1, в домашней папке подкаталог: **/номер\_группы/ФИО\_студента**, где в дальнейшем будут храниться все файлы студента. Перейти в корневой каталог и вывести его содержимое используя команды **dir и ls –all**, проанализировать различия.

Выполним эти требования, рисунок 1.

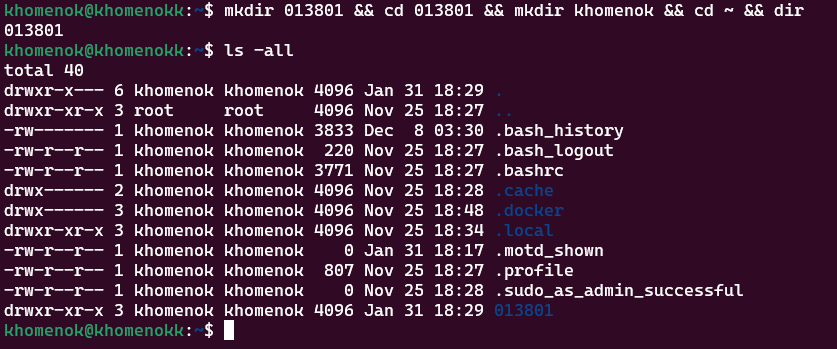


Рисунок 1 – Выполненный пункт №2

После перехода в корневой каталог и использования команды dir и ls -all, происходит следующее: dir – это команда, используемая для вывода содержимого каталога, тогда как ls -all – более расширенная команда, выводящая подробную информацию о файлах и каталогах, включая скрытые. Различия между ними заключаются в том, что dir не является стандартной командой в Linux, в то время как ls -all является более универсальным инструментом, доступным во всех основных дистрибутивах Linux, и предоставляет более полезную информацию для анализа содержимого каталогов.

3 Проверить действие команд ps, ps –x, top, htop. Найти в справочной системе используя команду man справку по функциям fprintf, fputc и команде ls (рисунок 2).

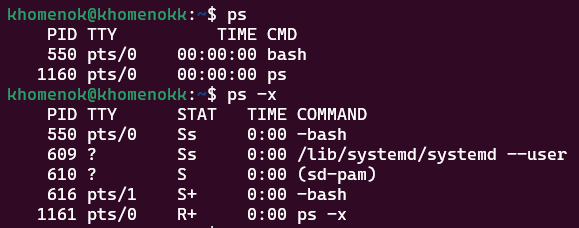


Рисунок 2 – Выполнение пункта №3

При выполнении команд ps, ps -x, top, и htop были изучены действия различных утилит мониторинга процессов в Linux.

Однако, не удалось найти man-страницы для функций fprintf и fputc. Это связано с тем, что данные функции принадлежат стандартной библиотеке языка программирования C, а не непосредственно операционной системе. **Для нахождения man-страницы функций fprintf и fputc необходимо установить компилятор gcc уже на этом этапе.**

Также были выполнены команды top и htop, которые позволяют мониторить нагрузку на систему в режиме реального времени.

4 В текстовом редакторе joe (вызов: joe 1.c) написать программу 1.c, выводящую на экран фразу “HELLO SUSE Linux”. Компилировать полученную программу компилятором gcc: gcc 1.c –o 1.exe. Запустить полученный файл 1.exe на выполнение: ./1.EXE

Для этого выполним:

sudo apt install joe -y

joe 1.c

Содержимое файла 1.c:

#include <stdio.h>

int main() {

printf("HELLO SUSE Linux\n");

return 0;

}

Чтобы сохранить файл 1.с, необходимо нажать Ctrl + K, затем X, чтобы сохранить и выйти из файла, либо D, чтобы просто сохранить. Далее необходимо установить компилятор gcc:

sudo apt update && sudo apt upgrade -y

sudo apt install gcc -y

Далее необходимо выполнить действия, показанные на рисунке 3.

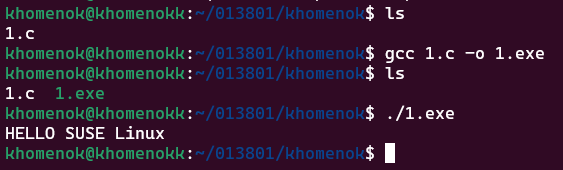


Рисунок 3 – Выполнение пункта №4

5 Написать скрипт, выводящий на консоль и в файл все аргументы командной строки.

#!/bin/bash

# Проверяем, переданы ли аргументы командной строки

if [ $# -eq 0 ]; then

echo "Нет аргументов командной строки для обработки."

exit 1

fi

# Выводим аргументы на консоль

echo "Аргументы командной строки:"

echo "$@"

# Записываем аргументы в файл arguments.txt

echo "Аргументы командной строки:" > arguments.txt

echo "$@" >> arguments.txt

echo "Аргументы командной строки были записаны в файл arguments.txt."

Проверим работоспособность скрипта, которую можно увидеть на рисунке 4.

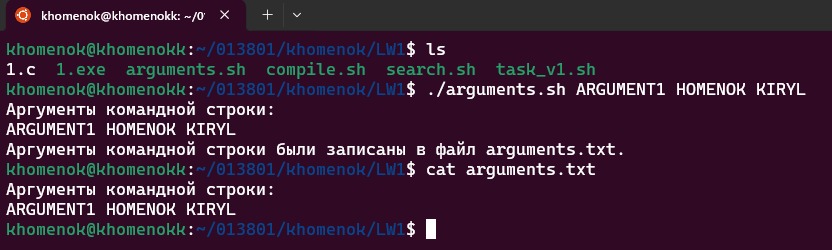


Рисунок 4 – Проверка работоспособности скрипта №1

6 Написать скрипт, выводящий в файл (имя файла задаётся пользователем в качестве первого аргумента командной строки) имена всех файлов с заданным расширением (третий аргумент командной строки) из заданного каталога (имя каталога задаётся пользователем в качестве второго аргумента командной строки).

#!/bin/bash

# Проверяем количество аргументов командной строки

if [ $# -ne 3 ]; then

echo "Использование: $0 <имя\_файла> <каталог> <расширение>"

exit 1

fi

# Имя файла, каталога и расширение

output\_file="$1"

directory="$2"

extension="$3"

# Проверяем, существует ли указанный каталог

if [ ! -d "$directory" ]; then

echo "Указанный каталог не существует."

exit 1

fi

# Переходим в указанный каталог

cd "$directory" || exit

# Проверяем, есть ли файлы с указанным расширением в каталоге

files=$(find . -maxdepth 1 -type f -name "\*.$extension")

# Проверяем, найдены ли файлы

if [ -z "$files" ]; then

echo "Файлов с расширением .$extension не найдено в каталоге $directory."

exit 0

fi

# Записываем имена файлов в указанный файл

echo "Файлы с расширением .$extension в каталоге $directory:" > "$output\_file"

echo "$files" >> "$output\_file"

echo "Имена файлов были записаны в файл $output\_file."

Проверим работоспособность скрипта, которую можно увидеть на рисунке 5.

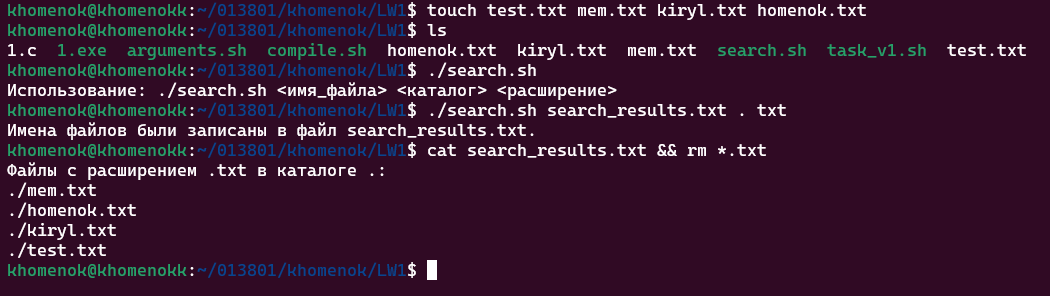


Рисунок 5 – Проверка работоспособности скрипта №2

7 Написать скрипт, компилирующий и запускающий программу (имя исходного файла и exe- файла результата задаётся пользователем в качестве аргументов командной строки). В случае ошибок при компиляции вывести на консоль сообщение об ошибках и не запускать программу на выполнение.

#!/bin/bash

# Проверяем, что пользователь предоставил два аргумента

if [ "$#" -ne 2 ]; then

echo "Использование: $0 <source\_c\_file> <имя\_output\_файла>"

exit 1

fi

# Извлекаем имена файлов из аргументов командной строки

source\_file=$1

exe\_file=$2

# Компилируем программу

gcc -o "$exe\_file" "$source\_file"

# Проверяем успешность компиляции

if [ $? -eq 0 ]; then

echo "Программа успешно скомпилирована. Запуск..."

./"$exe\_file"

else

echo "Ошибка: Программа не скомпилирована."

fi

Проверим работоспособность скрипта, которую можно увидеть на рисунке 6.

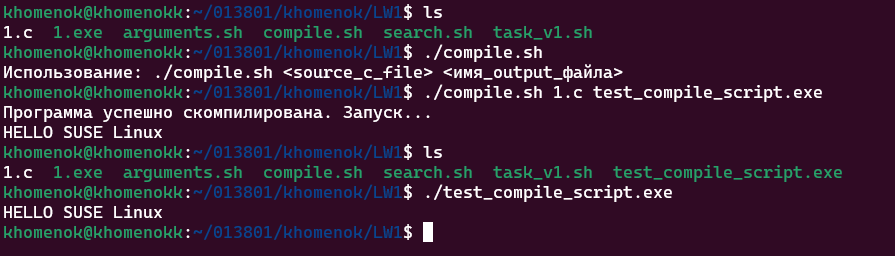


Рисунок 6 – Проверка работоспособности скрипта №3

Таким образом, общая часть лабораторной работы была выполнена. Далее необходимо приступить к выполнению индивидуального задания.

**3 Выполнение индивидуального задания:**

Написать скрипт для поиска файлов заданного размера в заданном каталоге (имя каталога задаётся пользователем в качестве третьего аргумента командной строки). Диапазон (мин.- мах.) размеров файлов задаётся пользователем в качестве первого и второго аргумента командной строки.

Реализованный скрипт представлен ниже:

#!/bin/bash

if [ "$#" -ne 3 ]; then

echo "Использование: $0 <мин. размер> <макс. размер> <каталог>"

exit 1

fi

if ! [[ "$1" =~ ^[0-9]+$ ]]; then

echo "Ошибка: Первый аргумент должен быть числом"

exit 1

fi

if ! [[ "$2" =~ ^[0-9]+$ ]]; then

echo "Ошибка: Второй аргумент должен быть числом"

exit 1

fi

if [ ! -d "$3" ]; then

echo "Ошибка: Каталог '$3' не существует"

exit 1

fi

min\_size=$1

max\_size=$2

directory=$3

echo "Файлы с диапазоном размера от $min\_size до $max\_size:"

find "$directory" -type f -size +"$min\_size"c -size -"$max\_size"c -exec ls -lh {} \; 2>/dev/null

Демонстрация работы скрипта представлена на рисунке 7.

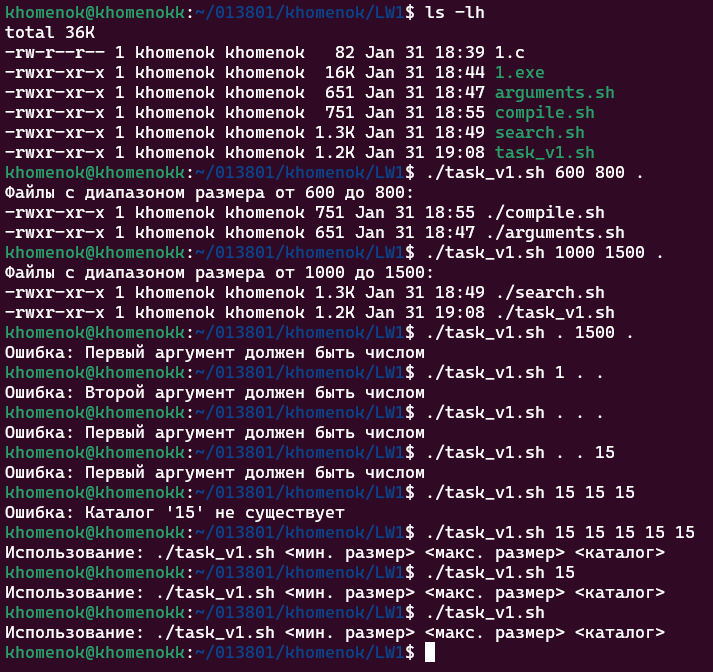


Рисунок 7 – Демонстрация работы написанного скрипта

Таким образом, лабораторная работа была выполнена с общим и индивидуальным заданием.

**3 Вывод**

В ходе выполнения лабораторной работы по управлению операционной системой Linux и использованию интерпретатора Bash были освоены основные команды и инструменты Linux. Была создана структура каталогов в домашней папке, изучены команды для работы с процессами и поиска информации в справочной системе, затем была написана программа на языке Си и скрипты на Bash для работы с файлами и процессами.

Особое внимание уделено написанию скрипта для поиска файлов заданного размера в указанном каталоге с заданным диапазоном размеров. Полученные навыки и практический опыт работы с командной строкой и скриптами Bash позволят эффективно управлять и администрировать системы Linux в будущем.