Міністерство освіти й науки України

Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Кафедра автоматизації проектування енергетичних процесів і систем

ЗВІТ

Про виконання лабораторної роботи №5

З дисципліни «Архітектура системного програмного забезпечення»

**Створення сценаріїв в оболонці Bash**

TI-92 Черноусова Дениса

Перевірила д.т.н. Левченко Л.О.

Київ - 2021

**Мета роботи:**

- набути навичок створювання bash-скриптів в ОС Linux.

**Теоретичні відомості**

Bourne-again shell (GNU Bash) - це реалізація Unix shell, написана на C в 1987 році Брайаном Фоксом (Brian Fox) для GNU Project. Синтаксис мови Bash є надбудовою синтаксису мови Bourne shell. Переважна більшість скриптів для Bourne shell можуть бути виконані інтерпретатором Bash без змін, за винятком скриптів, які використовують спеціальні змінні або вбудовані команди Bourne shell. Також синтаксис мови Bash включає ідеї, запозичені з Korn shell (ksh) і C shell (csh): редагування командного рядка, історія команд, стек директорій, змінні $ RANDOM і $ PPID, синтаксис POSIX *для підстановки команд: $* (...).

Командна оболонка *BASH* дозволяє створювати скрипти за допомогою групування декількох команд, які виконують певну дію.

***Скрипт*** - це звичайний текстовий файл, що містить системні або вбудовані команди оболонки. Такий файл може бути запущений на виконання наступним чином: ***$bash ім’я\_файла.***

Оболонка послідовно інтерпретує і виконує команди, задані в сценарії. Ці ж команди можуть бути виконані простим послідовним викликом їх в командному рядку оболонки. Для файлів сценаріїв оболонки *bash* прийнято встановлювати розширення ***.sh***. Тобто, для виконання скрипту необхідно запустити командну оболонку, передавши їй як параметр ім’я відповідного файлу. Є другий варіант запуску скрипту - вказати його ім'я в командній оболонці (тобто зробити з нього якийсь вид програми). Для цього треба в параметрах доступу визначити файл як *виконуваний*, і в перших рядках цього файлу явно вказати оболонку, для якої призначений цей скрипт, в такий спосіб: **#!оболонка**

Будь-який сценарій для Bash починається з вказівки в першому рядку комбінації:

**#!/bin/bash**

Символи **#!** повідомляють системі про те, що наступний за ними аргумент – це програма, яка застосовується для виконання даного файлу. В даному випадку програма ***/bin/sh*** - командна оболонка, що застосовується за замовчуванням. Ця послідовність вказує на програму, яка використовується для обробки сценарію в командну оболонку bash. У загальному випадку символ «***#***» в скрипті означає коментар, що означає ігнорування рядка. Але якщо він є першим символом файлу і за ним слідує символ «**!**» та шлях до файлу (наприклад, /bin/bash, /bin/perl, /bin/ sh і т.д), командна оболонка запускає відповідний файл і передає йому його ім'я в якості аргументу.

***Створення сценарію***

Створити файл, який містить команди, можна допомогою будь-якого текстового редактора. У даній роботі рекомендується використовувати вбудований в ***mc*** редактор. Для створення нового файлу в *mc* використовуйте комбінацію клавіш Shift + F4. Створіть файл з ім'ям *first* з таким вмістом:

#!/bin/sh

# first

# Цей файл переглядає всі файли в поточному каталозі для пошуку рядка

# POSIX, а потім виводить імена знайдених файлів в стандартне виведення.

for file in \*

do

if grep -q POSIX $file

then

echo $ file

fi

done

exit 0

Виконання скрипту відбувається по рядкам.

***Перетворення сценарію у виконуваний файл***

Файл сценарію можна виконати двома способами. Перший - запустити оболонку з ім'ям файлу сценарію як параметром: **$ /bin/sh first.** Цей варіант працює.

Другий спосіб (більш доцільний) - запускати сценарій, ввівши його ім'я і тим самим присвоївши йому статус інших команд Linux. Зробити це можна за допомогою команди **chmod**, змінивши режим файлу (**file mode**) і зробивши його виконуваним для всіх користувачів:

**$ chmod u+x <ім’я сценарія>**

**$ chmod u+x first**

Додані режими: для власника (User), виконання (x - eXecutable).

Аналогічну функціональність реалізує наступна команда:

**chmod 744 <ім’я\_файла\_сценарія>**

Після цього можна виконувати файл за допомогою **команди $ first**

При цьому може з'явитися повідомлення про помилку, яке говорить про те, що команда не знайдена.

Запуск сценарію на виконання з поточного каталогу проводиться за допомогою наступної команди: **./ <ім’я\_файла\_сценарія>**

Виправити помилку можна запровадивши з клавіатури в **командному рядку ./first** в каталозі, що містить сценарій, щоб задати командній оболонці повний відносний шлях до файлу.

Зазначення шляху, який починається з символів «**./»**, дає ще одну перевагу - ви випадково не зможете виконати іншу команду з тим же ім'ям, що і у вашого файлу сценарію. Після того як ви переконаєтеся в коректній роботі вашого сценарію, можете перемістити його в більш відповідне місце, ніж поточний каталог.

***Синтаксис команд для створення сценарію***

*Змінні та підстановка їх значень*. Всі змінні у мові – текстові. Їх імена повинні починатися з літери і складатися з латинських букв, цифр і знаку підкреслення (**\_**). Щоб скористатися значенням змінної, треба перед нею поставити символ **$**. Використання значення змінної називається *підстановкою*.

Розрізняють два класи змінних: позиційні та з іменем.

*Позиційні змінні* - це аргументи командних файлів, їх іменами слугують цифри:

**$0 -** ім'я виконуваної команди (для скрипту - це шлях, вказаний при його виклику, для функції - ім'я оболонки).

**$1** - перший аргумент, другий - $2 і т.д. - змінні, які відповідають аргументам, заданим при виклику сценарію (n - десяткове число, що відповідає номеру аргументу).

Значення змінній присвоюється наступним чином: **ім’я змінної=значення**.

Наприклад, X=1, або X=а, або X="f" і т.п. Але до і після знаку «=» немає пропуску!

Прийнято називати змінні буквами верхнього регістру, наприклад:

#!/bin/bash

TERM=vt100

При виконанні команд використовується *підстановка змінних*. В команду «підставляється» будь що (змінна, виведення іншої команди і т.п.). Для підстановки використовується або символ «$», або вираз, розміщений у зворотні апострофи ***(`вираз`).***

Якщо в тексті команди зустрічається символ «$», то наступний за ним текст до пробілу або кінця команди інтерпретується як ім'я змінної, значення якої підставляється в текст команди.

Наприклад: $FRUIT=Яблуко

$echo “Фрукт ”$FRUIT

Команда *echo* виведе на екран Фрукт Яблуко.

Команди $DATE=`date`

$echo $DATE

Виведуть системну дату на екран.

Командна оболонка дозволяє виконувати арифметичні операції. Для цього вираз, яке необхідно інтерпретувати як арифметичний, записати у подвійні круглі дужки, і перед ними ставиться знак долара.

Наприклад: $foo=$((((5+3\*2)–4)/2))

echo $foo

Для отримання значення змінної використовуються наступний синтаксис:

ALFA=$BETA+$GAMMA

Виконувану команду необхідно записати у зворотні апострофи і присвоїти змінній: **FILES=`/bin/ls -a/home/student`**

Користувач може надати значення змінній через командну оболонку за допомогою команди **read**, якої в якості аргументу передається ім'я необхідної змінної. Наприклад:

$read CHOICE

Привіт !!!

$echo “Ви ввели ”${CHOICE}

Привіт !!!

$echo “Ви ввели ”${CHOICE}

**Завдання:**

Розробити скрипт, який виконує зазначені дії згідно варіанту по списку журналу групи. Створити файл з назвою власного прізвища, записати до нього ПІБ студента, групу навчання, за бажанням додаткову інформацію щодо ваших уподобань. Створити скрипт, який виводить ПІБ студента, групу навчання, коментарі щодо майбутніх дій, необхідно робити затримку після видачі результатів на екран. Скрипт повинен містити функції, які виконують зазначені дії, а також додатково записати дію відповідно до варіанту.

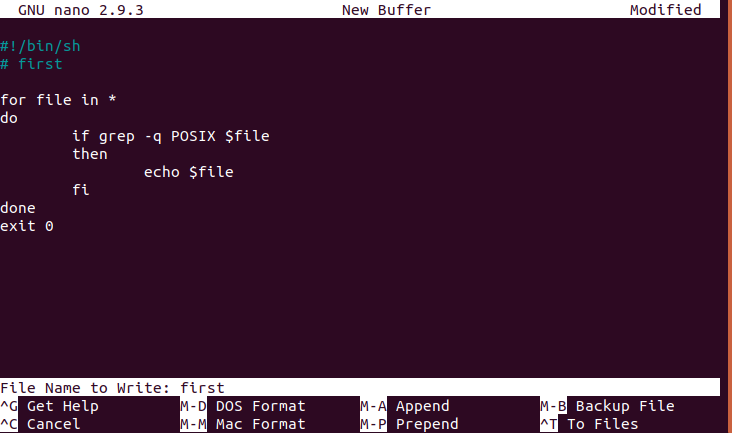
**Варіант :**

1. В каталозі віднайти виконуваний файл.

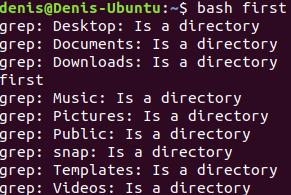
**Хід виконання роботи**



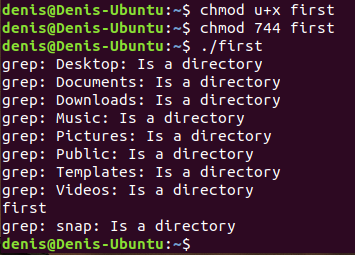
Вхід в Midnight Commander

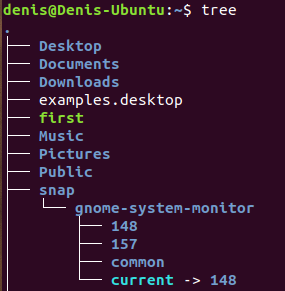


Shift + F4 нажимаємо, аби перейти в nano і записуємо скрипт

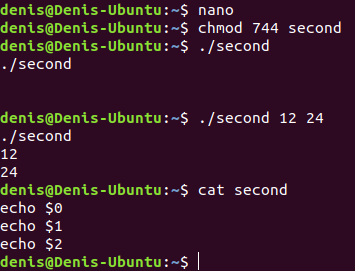
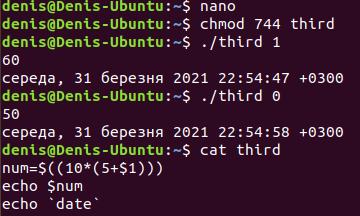


Виведемо імена усіх файлів в директорії

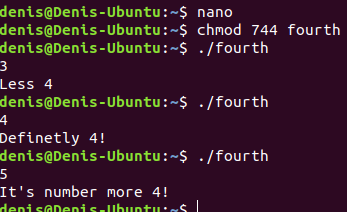
Перетворюємо файл у виконуваний файл

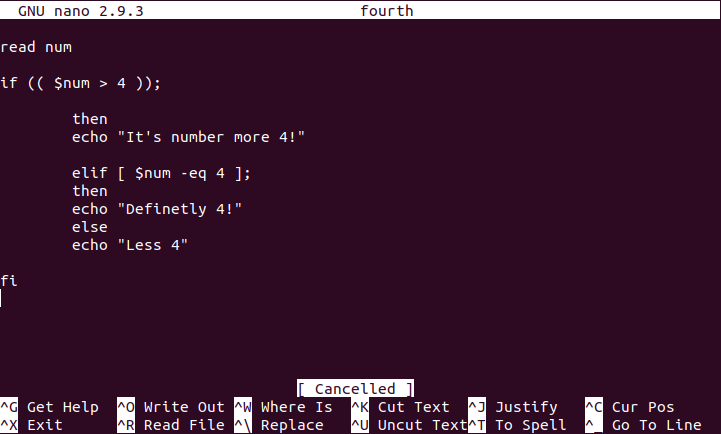
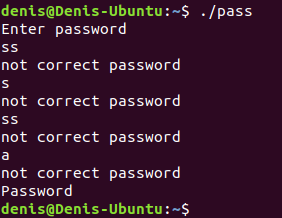


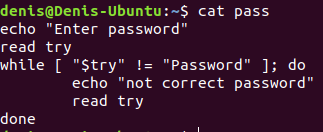
Программа tree виводить тепер цей виконуваний файл зеленим

Створюємо в середовищі Nano скрипт, який виводить позиційні змінні

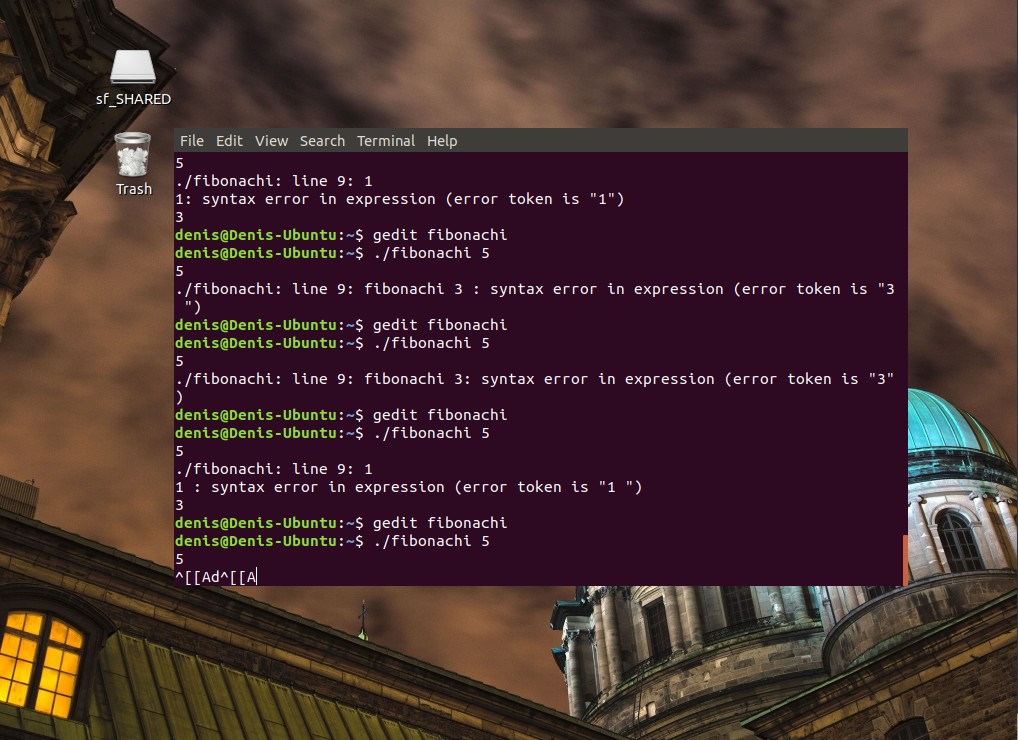
Створюємо в середовищі Nano скрипт, який обчислює арифметичний вираз і виводить дату

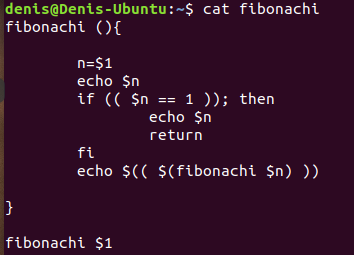
Створюємо скрипт, який порівнює введене значення

Скрипт в Nano

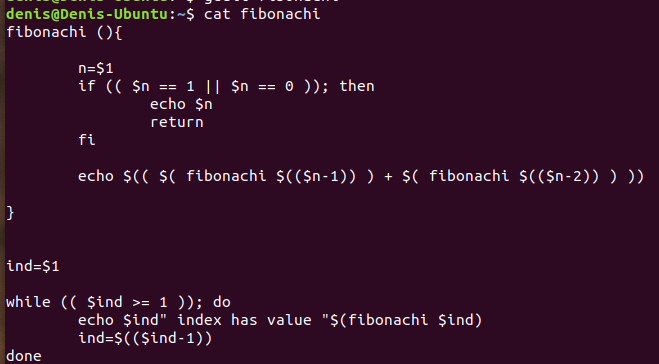
Скрипт для введення паролю

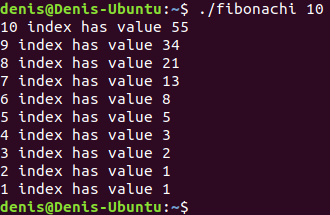
Його код

Зникнення інтерфейсу при потраплянні в безкінечний цикл

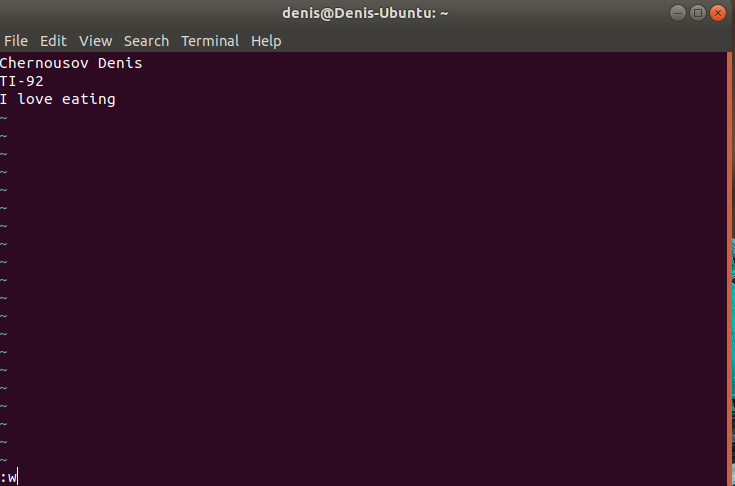


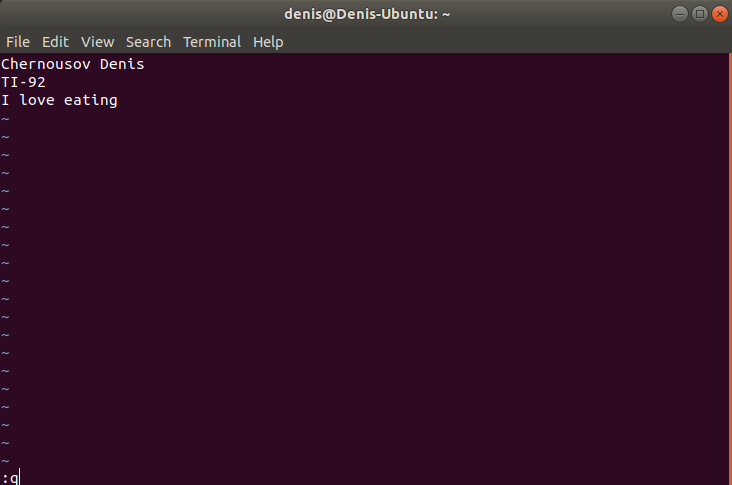
Код скрипту

Код рекурсивної функції Фібаначчі

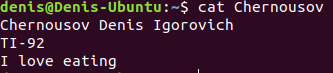


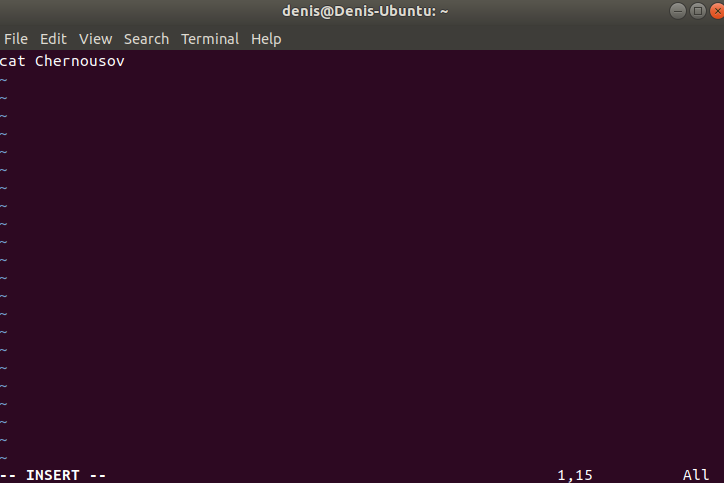
Вивід скрипту для індексу 10

Запис та збереження персональних даних в файлі через Vim

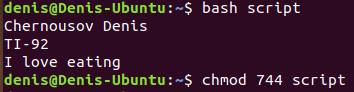


Вихід з Vim

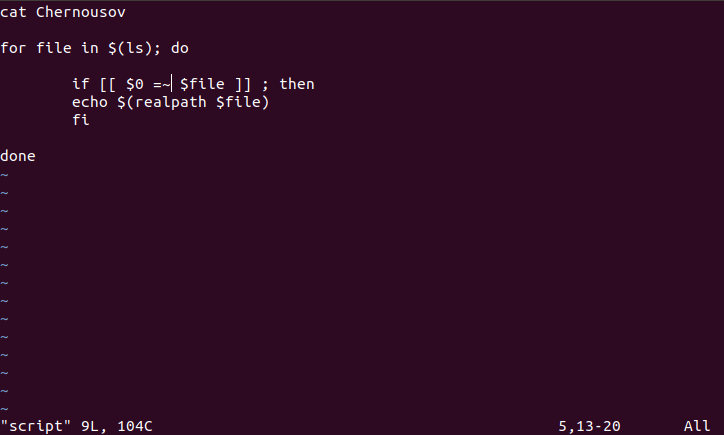
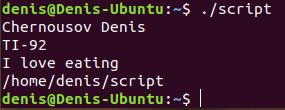
Виведення даних файлу в консоль

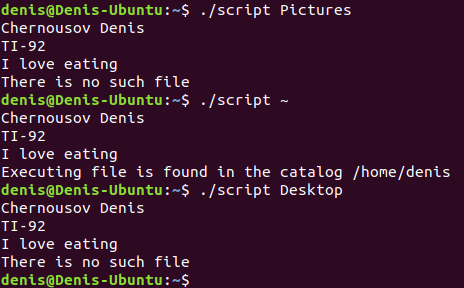


Написання скрипту, який виводить дані конкретного файлу

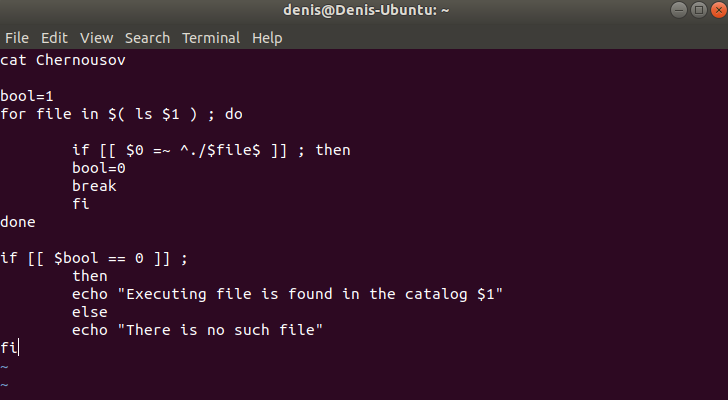


Цей скрипт у дії

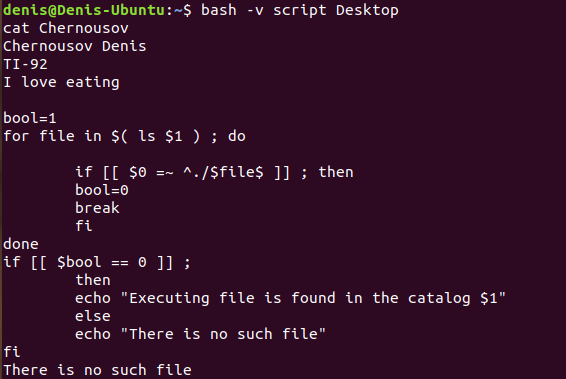
Створення скрипту, який виводить абсолютний шлях виконуваного файлу в каталозі

Цей скрипт у дій

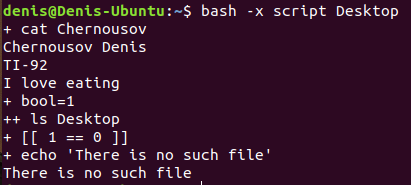
Скрипт, який перевіряє чи є в зазначеному каталозі виконуваний файл

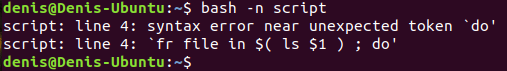


Код скрипту

**Відладка сценаріїв** 

Виконання скрипту з ключем -v для виведення команд перед їхнім виконанням

Ключ -х виводить на екран команди після обробки командного рядка



Ключ -n перевіряє скрипт на синтаксичні помилки (помилка була створена спеціально)

**Висновок**

Під час роботи над лабораторною роботою були здобуті навички створення і редагування bash скриптів в ОС Linux. Були використані такі середовища програмування як Vim , Nano та Gedit. Здобуті навички створення та маніпуляції зі змінними (позиційними та тим, що мають ім’я). Було використано умовні оператори та різні цикли. Створено рекурсивну функцію fibanachi для обрахунку чисел Фібоначчі. Був написаний скрипт для пошуку виконуваного файлу.