МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Графічний інтерфейс налаштування віддаленого доступу до обладнання Cisco з використанням протоколу SSH»

Завідувач випускаючої кафедри — **Довбиш А. С.**

Керівник роботи — **Великодний Д. В.**

Студента групи ІН – 71 — **Кібенка О. С.**

СУМИ 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук

Затверджую_____

Зав. кафедрою Довбиш А. С.

"_____" _____2021 г.

ЗАВДАННЯ
до випускної роботи

Студента четвертого курсу, групи ІН-71 спеціальності "Комп'ютерні науки" денної форми навчання Кібенка Олександра Сергійовича.

Тема: "Графічний інтерфейс налаштування віддаленого доступу до обладнання Cisco з використанням протоколу SSH"

Затверджена наказом від СумДУ

№_____ от _____2021 г.

Зміст пояснювальної записки: 1) огляд аналогічних систем; 2) огляд типу архітектури системи, патернів проектування, вибір типу програмного забезпечення; 3) вибір технологій для розробки проекту, серед яких мова програмування; 5) результат розробки проекту; 6) шляхи подальшого розвитку програми.

Дата видачі завдання "_____" _____2021 г.

Керівник випускної роботи _____ Великодний Д. В.

Завдання взяв до виконання _____ Кібенко О. С.

РЕФЕРАТ

Записка: 41 стор., 10 рис., 14 джерел.

Об’єкт дослідження — графічний інтерфейс налаштування віддаленого доступу до обладнання Cisco з використанням протоколу SSH.

Мета роботи — розробка графічного інтерфейсу налаштування віддаленого доступу до обладнання Cisco з використанням протоколу SSH.

Результати — розроблено графічний інтерфейс налаштування віддаленого доступу до обладнання Cisco з використанням протоколу SSH, який дозволяє вивести команди для налаштування роутера, ввівши його параметри, а потім підключитися до роутера з використанням протоколу SSH.

Ключові слова: SSH, Java, Cisco, роутер, програма, графічний інтерфейс.

Зміст

Вступ

У нашому сучасному світі, який так швидко розвивається, ми як ніколи потребуємо безпеки. Але протоколи, які існують, не завжди підходять саме для цього. Деякі з них створені в першу чергу для зручності.

Шифрування інформації є надзвичайно важливою складовою безпеки та ефективності сучасних програм, адже методи злому також вдосконалюються.

Враховуючи це, можна прийти до висновку, що необхідно передавати інформацію якомога безпечнішими методами. Тобто шифрувати її, особливо паролі, які, якщо їх передавати без шифрування, можуть бути прочитані, що, звичайно, не є гарно.

Нашою метою є створення безпечної та зручної програми, яка зможе підключитися до роутера саме за допомогою захищеного протоколу SSH, а також налаштує пристрій, якщо у цьому є необхідність.

Подібні системи звичайно також уже існують. Вони широко використовуються для доступу до інформації, яка знаходиться на віддалених пристроях. І це надзвичайно важливо, тому що через мережу інтернет будь-хто може намагатися створити з’єднання з вашим роутером, наприклад. І цього, звичайно, краще уникнути за допомогою безпечних протоколів передачі даних.

1 Аналіз предметної області

Сфера безпеки тісно пов’язана з нашим життям. Набагато краще, коли паролі передаються у зашифрованому вигляді. Наша програма для доступу до роутера допоможе уникнути втрати даних, такі як паролі, а також допоможе зручно налаштувати будь-який роутер в мережі інтернет, де б він не був.

1.1 Огляд існуючих рішень

Розглянемо аналогічні програми, які уже існують для таких цілей.

І, звичайно, мабуть найбільш відома із них – це PuTTY:

1) **PuTTY** – це клієнт, який безкоштовно розповсюджується, використовується для різних протоколів доступу, таких як Telnet, rlogin. Серед них, без сумніву і один з наших – SSH.

PuTTY дозволяє підключитися та керувати віддаленим пристроєм, таким як, наприклад, сервер. В PuTTY реалізована тільки клієнтська частина з’єднання – тобто відображення інформації, в той час як самі команди виконуються на сервері.

Програма спочатку була розроблена для Windows, але пізніше з’явилась і для Unix. Код для програми був повністю написаний на мові C, на відміну від нашої, яка написана на Java.

Деякі із можливостей програми приведені нижче [1]:

- Зберігання списку підключень для подальшого їх використання.
- Робота з ключами і версіями протоколу SSH.
- Клієнти SCP, SFTP.
- Підтримка IPv6.
- Підтримка логіну з відкритим ключем, в тому числі і без введення паролю.

До складу PuTTY входять такі важливі утиліти, як:

- Власне PuTTY – сам клієнт для Telnet і SSH.
- PSCP – клієнт для SCP.
- PSFTP – клієнт SFTP.
- Plink – інтерфейс командного рядка.
- Pageant – агент для аутентифікації.
- PuTTYgen – утиліта для генерації RSA, DSA ключів.

2) **Poderosa** – зручний клієнт SSH [2].

Старий проект, останнє оновлення якого було аж у 2004 році. Серед його мінусів:

- Немає підтримки російського/українського кодування.
- Немає входу за допомогою пароля під час запуску SSH клієнта.

Але плюс також є – програма має відкритий код і безкоштовна, як і попередній аналог.

3) **Vandyke SecureCRT –** дуже популярний клієнт від компанії, яка є лідером по розробці рішень в сфері забезпечення безпечного віддаленого доступу, передачі файлів та обміну інформацією [2].

Клієнт є платним, але дозволяє входити за паролем, підтримує все кодування, має гарячі клавіші та і взагалі зручний графічний інтерфейс для роботи.

1.2 Огляд існуючих протоколів

Також варто розглянути аналоги протоколів для SSH, деякі з яких є, хоч і менш безпечними і не шифрують інформацію, але можливо більш зручними у використанні.

І першим є сенс розглянути саме Telnet:

1) **Telnet** – мережевий протокол, що створений реалізувати текстовий термінальний інтерфейс в мережі [3].

Виконує функції прикладного рівня моделі OSI. Цей протокол використовується з тою самою ціллю, що і SSH, тобто для віддаленого управління різними пристроями, такими як сервера та роутери, але саме через проблем із безпекою і поступився SSH.

Але так як на деяких роутерах SSH може бути відсутнім, або, скажімо, неналаштованим, то даний протокол має змогу в повній мірі замінити останній.

2) **rlogin** – протокол прикладного рівня, в цілому дуже схожий на Telnet. Виконує ті ж самі функції, тобто дозволяє одній машині підключитися до іншої.

Так як вибраний нами протокол SSH шифрує з’єднання, то можна розглянути, в чому його перевага на прикладі мінусів rlogin [4]:

- Вся інформація передається без шифрування, в тому числі і паролі.
- Файлом .rlogin легко користуватися в корисливих цілях, тому багато компаній забороняють працівникам відкривати такі файли.
- Протокол довіряє клієнту rlogin віддаленої машини, чесно вказуючи інформацію, таку як порт та ім’я хоста. Цим може користуватися зловмисник, тому що віддалені хости не розподіляються на зони довіри.

1.3 Постановка задачі

Серед поставлених задач ми маємо наступну: створення графічного інтерфейсу для доступу до роутеру через протокол SSH. Її можна легко розбити на наступні підзадачі: створення роутера в емуляторі GNS3, створення власне графічного інтерфейсу, який можна розділити на інтерфейс для створення конфігураційних команд для роутера та інтерфейс підключення до роутера.

В наступних розділах буде детально розглянуто кожну з даних задач. Перш ніж перейти до реалізації за допомогою емуляцій та коду, буде розроблено схему роботи програми, її дизайн.

2 Огляд технологій для вирішення проблеми

Графічний інтерфейс доступу до віддаленого роутера повинен мати функції, які забезпечать користувачу можливість зручно підключитися до роутера, а також виконувати будь-які команди, які дозволені конфігурацією останнього.

Також однією із особливостей програми, на відміну від аналогів, є те що вона відає повний код налаштувань, які необхідно ввести в консоль роутера для налаштування до нього доступу, використовуючи протокол SSH.

2.1 Вибір симулятора мережі

Так як у нас немає реального роутера Cisco, то його потрібно емулювати. Для симуляції мережі існують декілька популярних програм:

1) **Cisco Packet Tracer** – симулятор мережі передачі даних, розроблений компанією Cisco. Дозволяє робити робочі моделі мережі, налаштовувати роутери, комутатори та інші пристрої, взаємодіяти з декількома користувачами [5].

Програма дозволяє створювати навіть складні макети мережі, перевіряти на працездатність різні топології. Та все ж функціональність пристроїв у симуляторі обмежена і не використовує весь спектр можливостей реальних пристроїв.

2) **GNS3** – це також симулятор мереж. Але на відміну від попередньої програми, він має набагато більше можливостей. Саме через обмеженість Packet Tracer мы будемо використовувати GNS3.

Насправді GNS3 – це не симулятор, а емулятор [6]. Емулятор дозволяє створити модель пристрою і запускати всередині оригінальне програмне забезпечення. В результаті емулюються всі компоненти пристрою, такі як

процесор, пам’ять, пристрої виведення. Тобто емулятор створить модель роутера Cisco, в якому запустить реальну операційну систему Cisco IOS.

Таким чином ми отримуємо повнофункціональний маршрутизатор. А ось Packet Tracer – це симулятор, тому що програмісти цього програмного забезпечення просто створили пристрої зі схожими на реальні командами та інтерфейсом.

Отже, маємо наступні плюси GNS3:

- Повний функціонал земульованих пристроїв. Тобто створивши роутер, ви отримаєте змогу запустити на ньому майже всі команді, які змогли б на реальному роутері.
- Можливість створення гетерогенних мереж. Тобто мереж, які складаються не тільки з пристроїв Cisco, а й інших.
- Додавання в мережу повноцінних робочих станцій та серверів. Тобто повноцінних компьютерів на Windows 10, наприклад. Це реалізовується за допомогою технологій віртуалізації, таких як WMWare або VirtualBox. Також можна додати реальний комп’ютер, що ми і будемо використовувати.

2.2 Аналіз мов програмування

Для реалізації даної програми не обійтися без мови програмування. Тож необхідно розглянуть декілька із них та знайти мову, яка найбільш підходить до поставленої задачі.

Від вибору мови може багато залежати, адже різні мови програмування орієнтовані на різні види розробки програмного забезпечення, мають свої можливості та свої обмеження.

Різні мови програмування по різному використовуються для досягнення поставлених перед розробником задач. У нашому випадку нам потрібна мова, яка дозволить створити десктопний додаток.

За даними, взятими з сайту habr.com список з п’яти найпопулярніших мов наступний [7]:

1) JavaScript
2) Java
3) C#
4) Python
5) PHP

Відкинувши номер один та п’ять (вони використовуються в основному для веб-сайтів), маємо мови програмування, які підходять для створення десктопного додатку.

Порівняємо їх:

1) **Java** – це об’єктно-орієнтована мова програмування, яка має велику популярність у всьому світі [8]. За допомогою неї написані сотні додатків. Додатки Java транслюються у байт-код, за рахунок чого можуть працювати на будь-якій архітектурі, на якій є реалізація віртуальної машини Java. Другою особливістю цієї мови є гнучка система безпеки, так як виконання програми повністю під контролем віртуальної машини

2) **C#** - мова також об’єктно-орієнтована. Була розроблена компанією Microsoft. Так як він має С-подібний синтаксис, то дуже схожий на джаву. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів та інше [10]. Ця мова – покращення С++, тому в ній були виправлені попередні помилки, що призводили до певних труднощів. Наприклад, множинне наслідування класів.

3) **Python** – мова з динамічною типізацією, на відміну від попередніх. Тобто тип змінної може змінюватися залежно від значення, що їй присвоєно. Також є об’єктно-орієнтованим. Має автоматичне управління пам’яттю. Особливістю мови є виділення блоків коду відступами за допомогою пробілу. Недоліком є менша швидкість роботи та більші затрати пам’яті при виконанні програм [11].

Отже, бачимо, що мови доволі схожі. Звичайно, кожна з них має різні особливості, проте нашу програму можна було б створити на будь-якій з них.

Тож було обрано мову Java, лідера списку. Завдяки її популярності для неї написано багато бібліотек та в інтернеті можна знайти приклади коду саме для цієї мови.

2.3 Вибір Java бібліотек

Переходячи до проектування системи, безумовно потрібно розуміти інструменти, які допоможуть у цьому. Саме для цього і створені бібліотеки. Вони дозволяють не писати функції та класи з нуля, а користуватися уже існуючими, що безперечно економить час.

Для написання нашої програми нам потрібні бібліотеки, що допоможуть побудувати графічний інтерфейс програми, а також встановити SSH з’єднання з роутером.

JSch – Java Secure Shell, ця бібліотека створена, щоб реалізувати SSH з’єднання на мові Java. Звичайно, вона не є унікальною, але не поступається своїм аналогам, а, можливо, і є кращою.

Можливості: авторизація за ключем, пост-форвардинг, з’єднання через проксі і багато іншого.

У нашій програмі ми будемо використовувати її.

Java Swing – бібліотека для створення графічного інтерфейсу. Має в собі багато компонентів, такі як кнопки, поля і так далі. Ця бібліотека – покращена версія AWT, має більш гнучкі інтерфейс компоненти.

Дуже популярна і користується попитом у розробників, тож буде використана для реалізації графічного інтерфейсу нашої програми.

3 Проектування та розробка графічного інтерфейсу та емуляція роутера

Отже, наша мета – розробка програми, яка дозволяє користувачу дві головні можливості:

- Підключення до роутера використовуючи SSH.
- Виведення команд для налаштування на роутері SSH.

Покажемо можливості програми за допомогою Use Case діаграми:

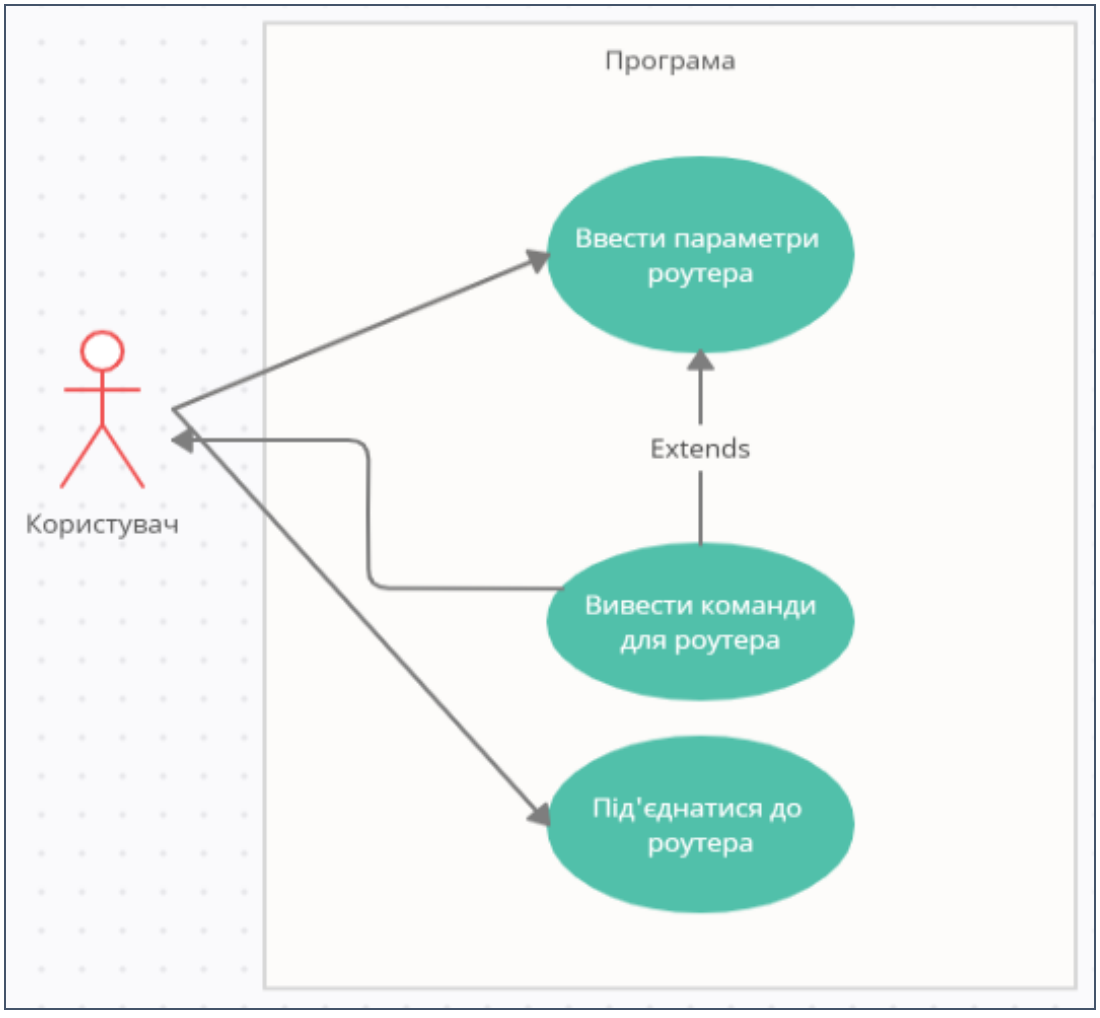

Рисунок 1 – Use Case діаграма

Як бачимо після введення необхідних параметрів програма видасть користувачу необхідні команди налаштувань.

Створимо також діаграму класів, які будемо використовувати. Кожний клас – це окреме вікно програми, а клас Main – це клас з якого викликається головне вікно, яке уже викликає інші.

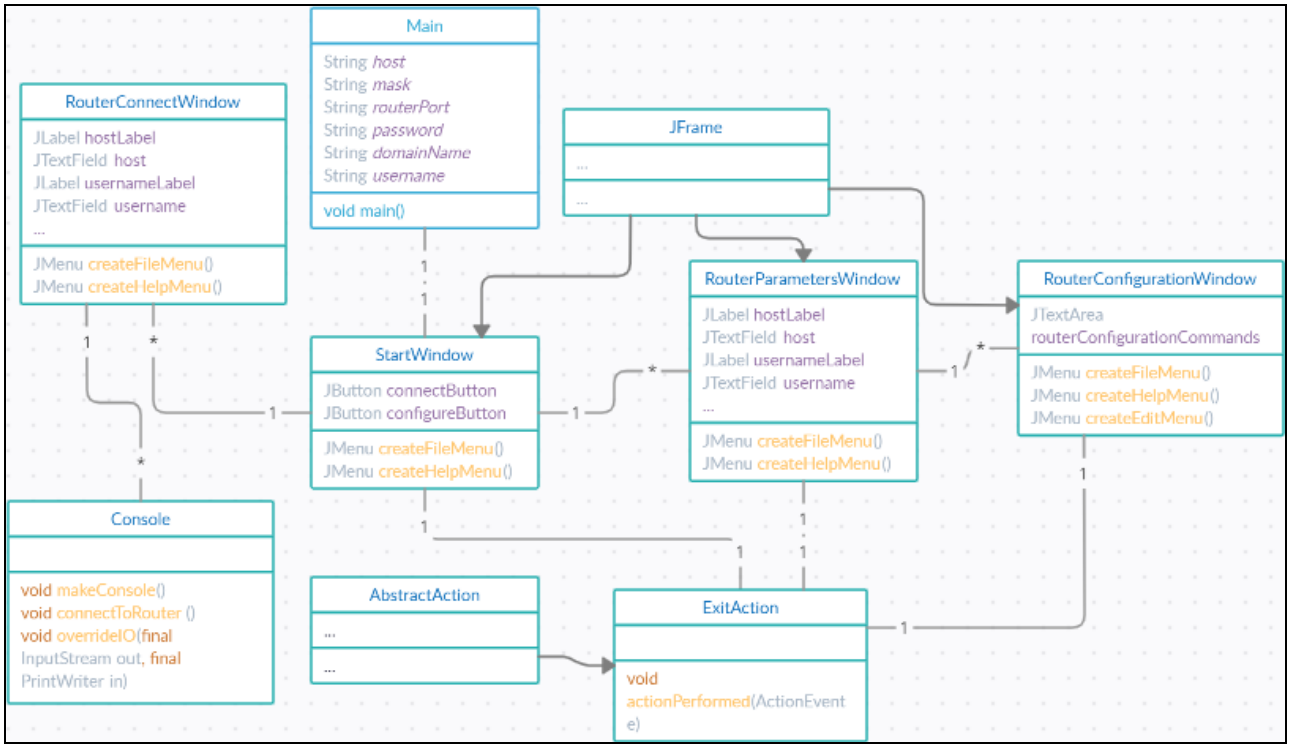

Рисунок 2 – Діаграма класів

Як бачимо, діаграма класів доволі ясно показує зв’язок між різними класами, таким чином розкриваючи взаємодію між ними.

3.1 Емуляція роутера в GNS3

Перед тим, як запустити програму, нам потрібно мати роутер. Тому спочатку треба земулювати роутер в емуляторі GNS3.

Для цього завантажимо програму з офіціального сайту.

Встановивши програму GNS3 на наш комп’ютер, відкриваємо її та створюємо проект.

Використовуючи об’єкти в лівій панелі, додаємо роутер та клауд – він буде використовувати реальний Ethernet порт комп’ютера, що дозволить роутеру також бути під’єднаним до комп’ютера та інтернету.

Отримаємо схему:

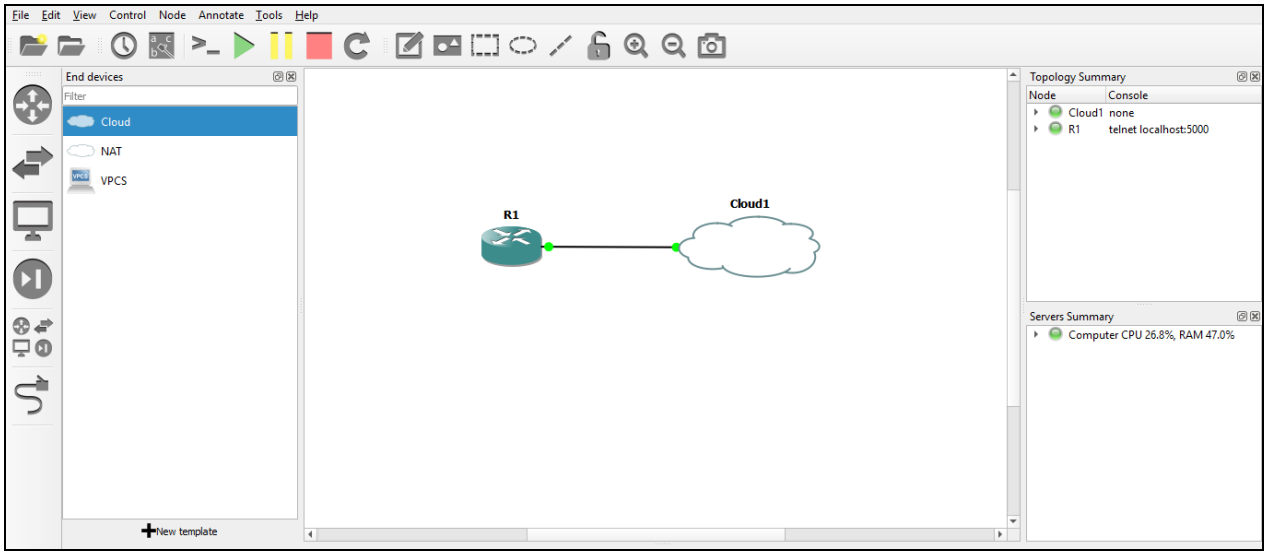

Рисунок 3 – Схема в GNS3

Далі потрібно увімкнути на клауді вищезгаданий порт Ethernet:

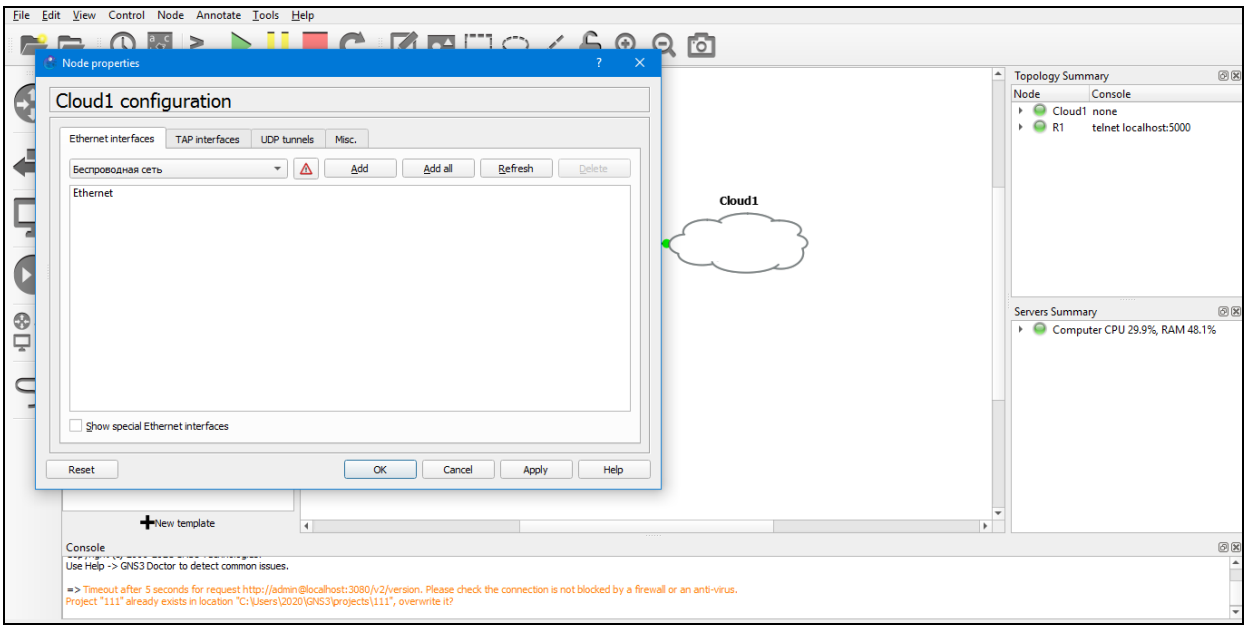

Рисунок 4 – налаштування клауда в GNS3

Тепер, після того, як ми налаштуємо роутер за допомогою команд, які буде виводити наша програма, ми зможемо під’єднатися до роутера.

3.2 Реалізація програми

Програма буде створена на мові програмування Java. Основні задачі, які потрібно вирішити:

- Створення графічного інтерфейсу за допомогою бібліотеки Swing.
- Створення SSH підключення за допомогою бібліотеки JSch.
- Створення консолі за допомогою перевизначення методів IO Stream.

Отже, почнемо зі створення форм, таких як вікна, поля, кнопки. Для їх створення буде використано класи JFrame, JTextField, JButton відповідно. Також знадобляться JMenuBar, JTextArea та інші.

Створимо головне вікно, з якого розпочнеться програма. На ньому створимо дві можливості: підключитися до роутера з SSH або перейти на вікно налаштувань роутера.

Створюємо клас, який наслідує JFrame:

```
public class StartWindow extends JFrame{}
```

В конструкторі цього класу вкажемо команди, необхідні для роботи вікна:

```
super("Майстер підключення до роутера через SSH"); // назва вікна
setResizable(false); // команда щоб зробити розмір вікна постійним
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // тепер програма буде закінчуватися, якщо закрити вікно
setVisible(true); // робимо вікно видимим
```

В самому класі вкажемо наші кнопки

```
private JButton connectButton = new JButton("Підключитися до роутера через SSH");
private JButton configureButton = new JButton("Налаштування роутера для підключення через SSH");
```

Також додамо кнопку допомоги та кнопку виходу зверху вікна.

Таким чином будемо робити всі вікна в програмі.

Результат зображено на рисунку:

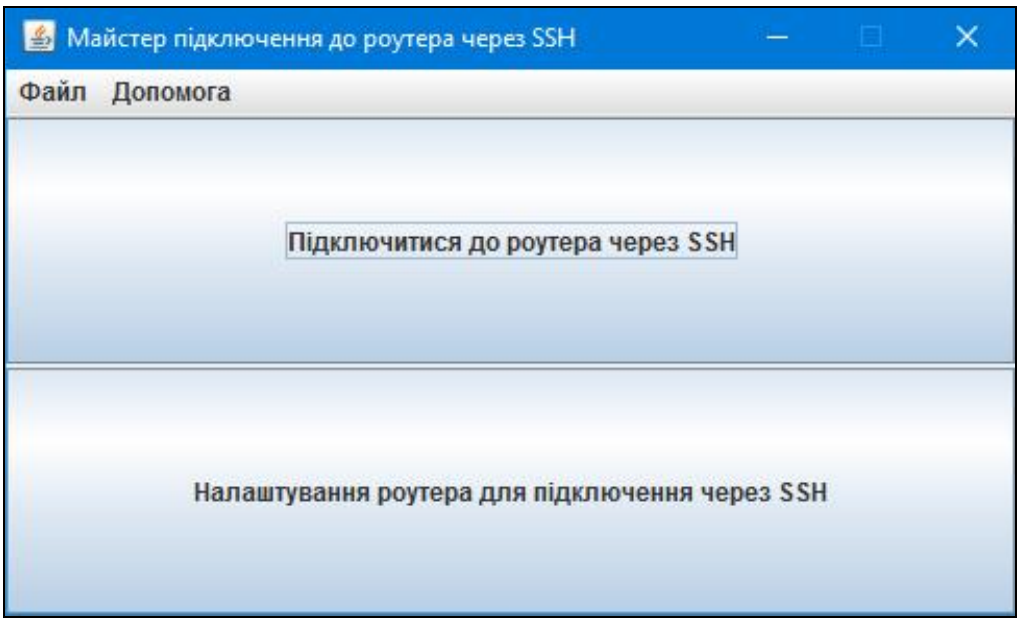

Рисунок 5 – Головне вікно програми

Після натискання на кнопку допомоги будуть виведені підказки:

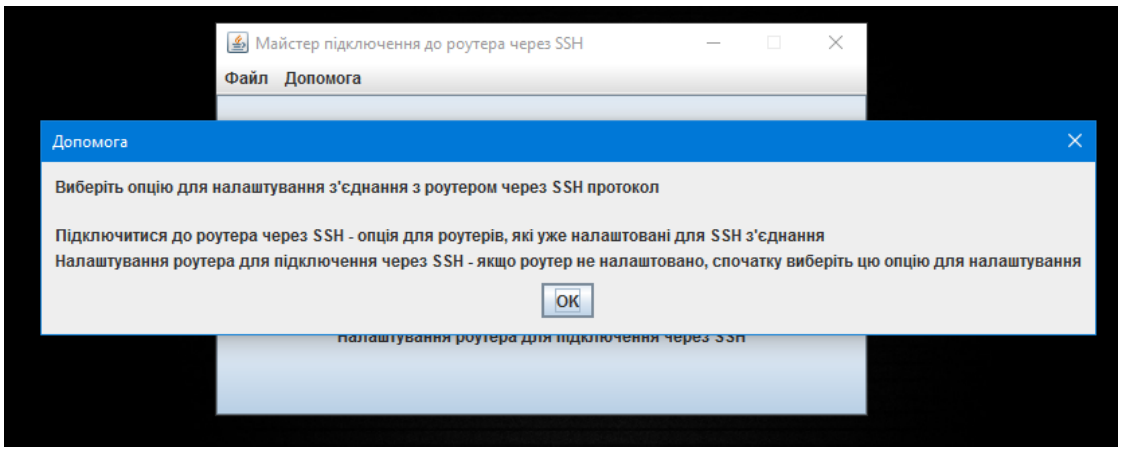

Рисунок 6 – Вікно допомоги

Перейдемо до наступного вікна – вікна налаштувань:

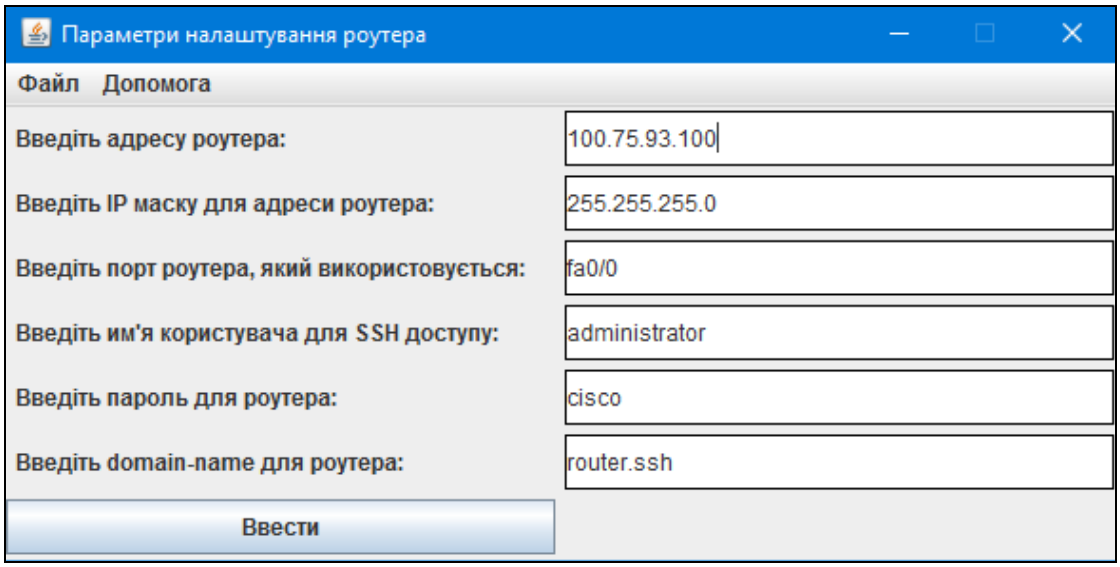

Рисунок 7 – Вікно введення параметрів роутера

Тут користувач має ввести параметри, які хоче бачити на роутері в результаті налаштувань. Після введення переходимо до наступного вікна, в якому бачимо команди.

Для зручності розроблені кнопки копіювання, вирізання тощо:

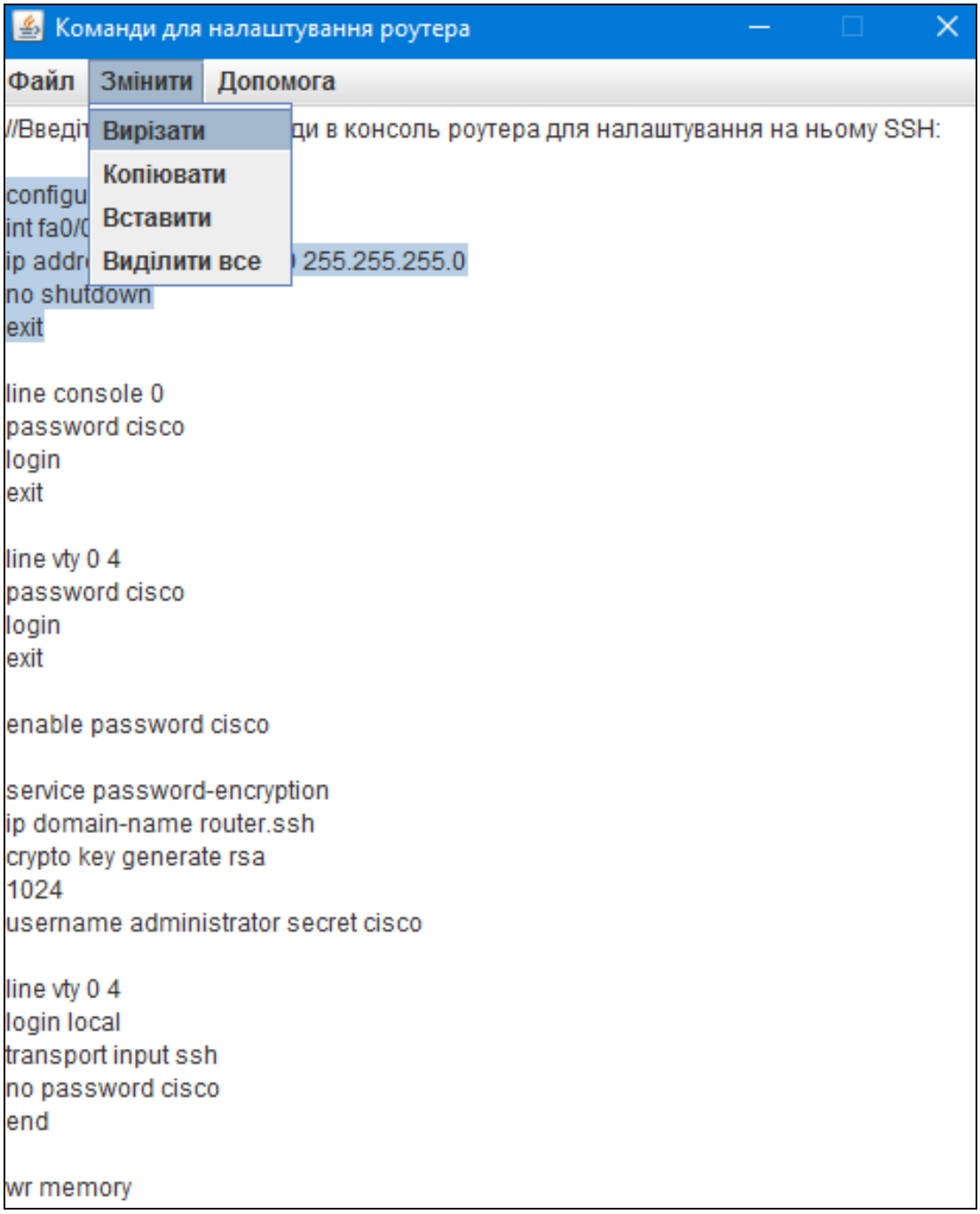

Рисунок 8 – команди для налаштування роутера

Тепер, після того як дані команди введені в роутер, можно переходити до іншого варіанту взаємодії з програмою – підключення до роутера.

Повертаємося на головний екран і вибираємо другу опцію – підключення.

В тому разі, якщо користувач щойно налаштовував роутер, необхідні дані підтягуються до вікна автоматично:

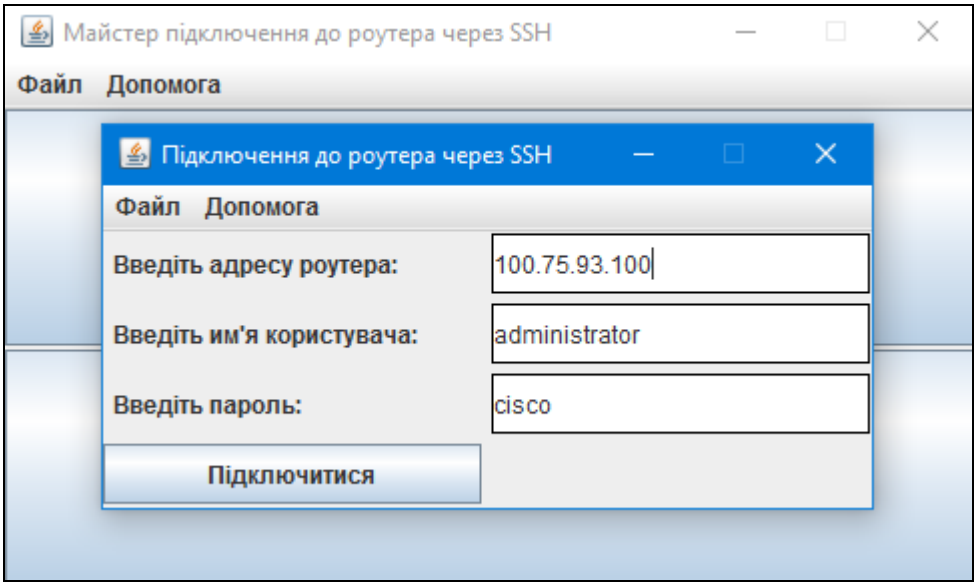

Рисунок 9 – вікно підключення до роутера

Отже, натискаємо підключитися, після цього відкриється консоль, в якій ми можемо вводити команди до роутера:

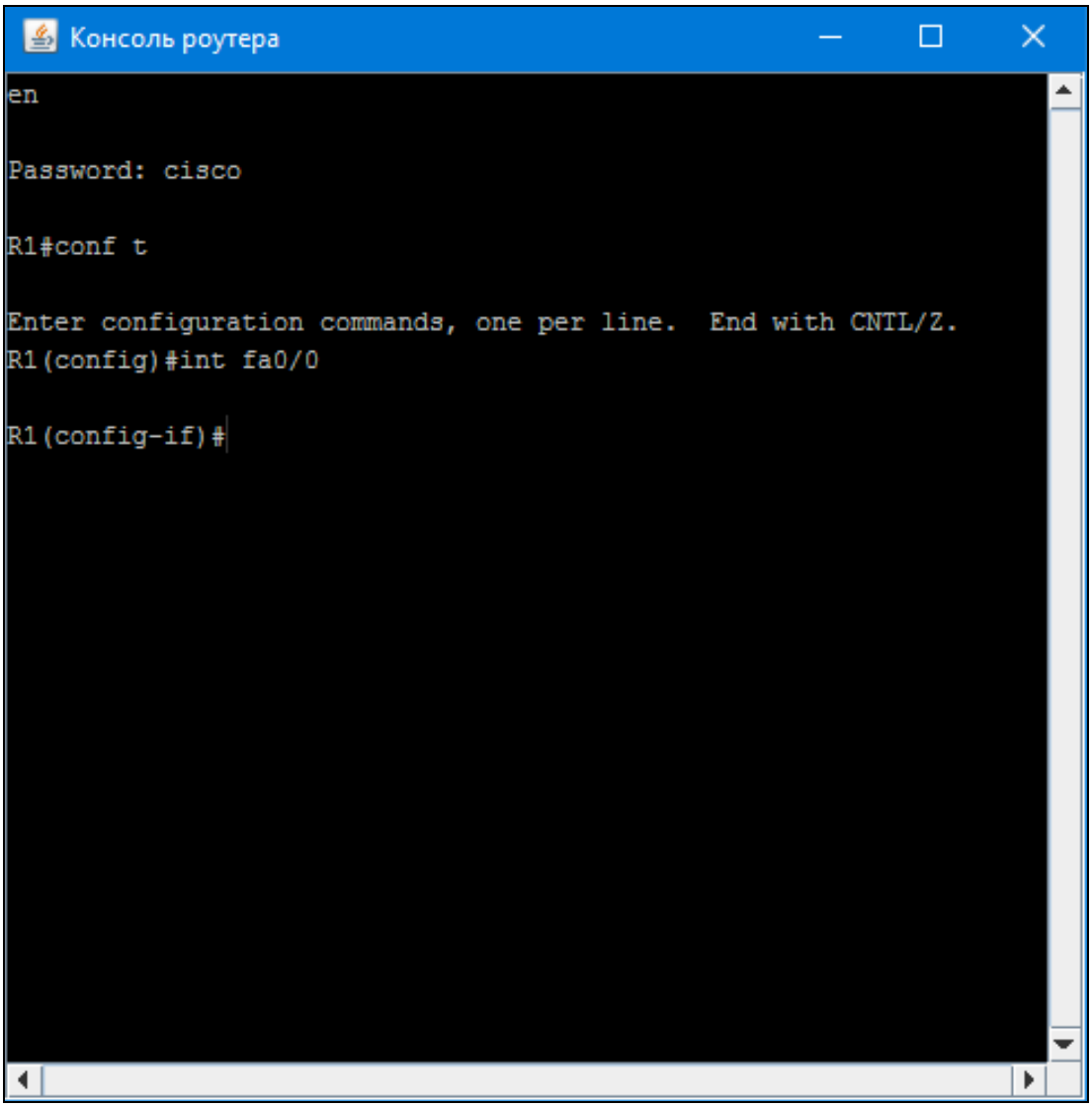

Рисунок 10 – консоль роутера

Отже, було розглянуто основні принципи поводження з програмою та її функціонал.

3.3 Шляхи подальшого розвитку програми

Шляхів подальшого розвитку програми багато. Наприклад, можна додати нові протоколи, скажімо Telnet.

Розширити можливість видавання команд для налаштування роутера на другі протоколи.

З іншої сторони, можна розвивати інтерфейс програми, роблячи її більш красивою, зрозумілою, зручною. Додати гарячі клавіші, оригінальне оформлення вікон і кнопок.

Дана програма має великий потенціал росту, оскільки є багато протоколів, багато команд для роутера, великий спектр варіантів оформлення.

Врешті решт, програма може стати такою ж функціональною, як і PuTTY, та на відміну від нього мати ще один плюс у вигляді виведення команд для налаштування роутера.

Висновки

У результаті виконання випускної роботи був створений графічний інтерфейс віддаленого доступу до роутера через протокол SSH. Програма реалізує дві задачі – власне саме підключення та виведення команд для налаштування роутера.

Було проаналізовано аналоги як самої програми, так і протоколу SSH.

Також було взято до уваги різні мови програмування та вибрано саме Java, зважаючи на її плюси.

Потім було спроектовано саму програму, використовуючи діаграми.

На наступному етапі було створено роутер в емуляторі GNS3, за допомогою програми виконані налаштування та підключення до нього.

Список використаної літератури

1. PuTTY [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/PuTTY – Назва з екрану.
2. Замена Putty — ищем удобный SSH клиент с вкладками для Windows [Електронний ресурс] – Режим доступу: https://yvision.kz/post/676001 – Назва з екрану.
3. Telnet [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Telnet – Назва з екрану.
4. Rlogin [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Rlogin – Назва з екрану.
5. Cisco Packet Tracer [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Cisco_Packet_Tracer – Назва з екрану.
6. Основи GNS3. Обзор [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/266503/ – Назва з екрану.
7. Рейтинг языков программирования 2021 [Електронний ресурс] – Режим доступу: https://habr.com/ru/post/543346/ – Назва з екрану.
8. Java [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Java – Назва з екрану.
9. C Sharp [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/C_Sharp – Назва з екрану.
10. Python [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Python – Назва з екрану.
11. Fielding, R.T. Dissertation Architectural Styles and the Design of Network-based Software Architectures. – 20 p.
12. Швец, А. Погружение в паттерны проектирования. – 5 с.
13. Tsonev, Kh. React in Patterns. – 13 p.

14. Head First Design Pattern Eric Freeman, Elisabeth Robson, Bert Bates, Kathy Sierra – O'Reilly Media, 2004. – 694 p.

Додаток А

Код для налаштування роутера

```
configure terminal
int fa0/0
ip address 100.75.93.100 255.255.255.0
no shutdown
exit

line console 0
password cisco
login
exit

line vty 0 4
password cisco
login
exit

enable password cisco

service password-encryption
ip domain-name router.ssh
crypto key generate rsa
1024
username cisco secret cisco

line vty 0 4
login local
transport input ssh
no password cisco
end

wr memory
```

Код програми

Клас Main:

```
package com.company;

public class Main {

   public static String host = "";
```

```
    public static String mask = "";
    public static String routerPort = "";
    public static String password = "";
    public static String domainName = "";
    public static String username = "";

    public static void main(String[] args){
        StartWindow app = new StartWindow();
    }

}
```

Клас StartWindow:

```
package com.company;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class StartWindow extends JFrame{

    private JButton connectButton = new JButton("Підключитися до роутера через SSH");
    private JButton configureButton = new JButton("Налаштування роутера для підключення
через SSH");

    public StartWindow (){
        super("Майстер підключення до роутера через SSH");
        this.setBounds(400,175,500,300);
        setResizable(false);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        JMenuBar menuBar = new JMenuBar();
        menuBar.add(createFileMenu());
        menuBar.add(createHelpMenu());
        setJMenuBar(menuBar);

        Container container = this.getContentPane();
        container.setLayout(new GridLayout(2,1, 2, 2));
        container.add(connectButton);
        container.add(configureButton);

        connectButton.addActionListener(new ConnectButtonListener());
        configureButton.addActionListener(new ConfigureButtonListener());

        setVisible(true);
    }
```

```
  private JMenu createFileMenu()
  {
    JMenu file = new JMenu("Файл");
    JMenuItem exit = new JMenuItem(new ExitAction());
    file.add(exit);

    return file;
  }

  class ExitAction extends AbstractAction
  {
    ExitAction() {
      putValue(NAME, "Вихід");
    }
    public void actionPerformed(ActionEvent e) {
      System.exit(0);
    }
  }

  private JMenu createHelpMenu()
  {
    JMenu help = new JMenu("Допомога");
    JMenuItem about = new JMenuItem(new AboutAction());
    help.add(about);

    return help;
  }

  class AboutAction extends AbstractAction
  {
    AboutAction() {
      putValue(NAME, "Про програму");
    }
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null,"Виберіть опцію для налаштування з'єднання з
роутером через SSH протокол\n\n" +
          "Підключитися до роутера через SSH - опція для роутерів, які уже налаштовані для
SSH з'єднання\n" +
          "Налаштування роутера для підключення через SSH - якщо роутер не налаштовано,
спочатку виберіть цю опцію для налаштування", "Допомога", JOptionPane.PLAIN_MESSAGE);
    }
  }

  class ConfigureButtonListener implements ActionListener {
    public void actionPerformed (ActionEvent e) {
      RouterParametersWindow routerParametersWindow = new RouterParametersWindow();
    }
```

```
    }

    class ConnectButtonListener implements ActionListener {
        public void actionPerformed (ActionEvent e) {
            RouterConnectWindow routerConnectWindow = new RouterConnectWindow();
        }
    }

}
```

Клас RouterConnectWindow:

```
package com.company;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.IOException;

public class RouterConnectWindow extends JFrame {

    private JLabel hostLabel = new JLabel("Введіть адресу роутера:");
    private JTextField host = new JTextField(Main.host, 1);

    private JLabel usernameLabel = new JLabel("Введіть им'я користувача:");
    private JTextField username = new JTextField(Main.username, 1);

    private JLabel passwordLabel = new JLabel("Введіть пароль:");
    private JTextField password = new JTextField(Main.password, 1);

    private JButton connectButton = new JButton("Підключитися");

    public RouterConnectWindow () {
        super("Підключення до роутера через SSH");
        this.setBounds(450, 250, 400, 200);
        setResizable(false);

        Container container = this.getContentPane();
        container.setLayout(new GridLayout(4, 2, 5, 5));

        container.add(hostLabel);
        container.add(host);
        container.add(usernameLabel);
        container.add(username);
        container.add(passwordLabel);
        container.add(password);
```

```
    container.add(connectButton);

    connectButton.addActionListener(new ConnectButtonListener());

    host.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
    password.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
    username.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
    hostLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));
    usernameLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));
    passwordLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));

    JMenuBar menuBar = new JMenuBar();
    menuBar.add(createFileMenu());
    menuBar.add(createHelpMenu());
    setJMenuBar(menuBar);

    setVisible(true);
}

class ConnectButtonListener implements ActionListener {
    public void actionPerformed (ActionEvent e) {
        Main.host = host.getText();
        Main.password = password.getText();
        Main.username = username.getText();

        Console console = new Console();

        try {
            console.makeConsole();
        } catch (IOException ex) {
            ex.printStackTrace();
        }

        console.connectToRouter();

    }
}

private JMenu createFileMenu()
{
    JMenu file = new JMenu("Файл");
    JMenuItem exit = new JMenuItem(new ExitAction());
    file.add(exit);

    return file;
}

private JMenu createHelpMenu()
```

```
  {
    JMenu help = new JMenu("Допомога");
    JMenuItem about = new JMenuItem(new AboutAction());
    help.add(about);

    return help;
  }

  class ExitAction extends AbstractAction
  {
    ExitAction() {
      putValue(NAME, "Вихід");
    }
    public void actionPerformed(ActionEvent e) {
      System.exit(0);
    }
  }

  class AboutAction extends AbstractAction
  {
    AboutAction() {
      putValue(NAME, "Про програму");
    }
    public void actionPerformed(ActionEvent e) {
      JOptionPane.showMessageDialog(null,"Введіть адресу роутера і натисніть
Підключитися", "Допомога", JOptionPane.PLAIN_MESSAGE);
    }
  }

}
```

Клас Console:

```
package com.company;

import java.awt.*;
import java.awt.event.KeyAdapter;
import java.awt.event.KeyEvent;
import java.io.*;
import java.util.List;
import java.util.Scanner;
import javax.swing.*;
import com.jcraft.jsch.*;

public class Console{

  JFrame frame = new JFrame("Консоль роутера");
  JTextArea area = new JTextArea();
```

```
JScrollPane scroll = new JScrollPane(area,
      JScrollPane.VERTICAL_SCROLLBAR_ALWAYS,
      JScrollPane.HORIZONTAL_SCROLLBAR_ALWAYS);

public void makeConsole() throws IOException {

   PipedInputStream inPipe = new PipedInputStream();
   PipedInputStream outPipe = new PipedInputStream();

   System.setIn(inPipe);
   System.setOut(new PrintStream(new PipedOutputStream(outPipe), true));

   PrintWriter inWriter = new PrintWriter(new PipedOutputStream(inPipe), true);

   overrideIO(outPipe, inWriter);

   frame.add(scroll);
   frame.setBounds(100, 100, 500, 500);
   area.setBackground(Color.BLACK);
   area.setForeground(Color.LIGHT_GRAY);
   area.setFont(new Font(Font.MONOSPACED, Font.PLAIN, 12));
   frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
   frame.setVisible(true);

}

public void connectToRouter(){
   try {
      JSch jsch = new JSch();
      Session session = jsch.getSession(Main.username, Main.host, 22);
      session.setPassword(Main.password);

      UserInfo userInfo = new MyUserInfo() {
         public void showMessage(String message) {
            JOptionPane.showMessageDialog(null, message);
         }

         public boolean promptYesNo(String message) {
            Object[] options = {"Yes", "No"};
            int foo = JOptionPane.showOptionDialog(null,
                  message,
                  "Warning",
                  JOptionPane.DEFAULT_OPTION,
                  JOptionPane.WARNING_MESSAGE,
                  null, options, options[0]);
            return foo == 0;
         }
      };
```

```
      session.setUserInfo(userInfo);
      session.connect(30000);
      Channel channel = session.openChannel("shell");
      channel.setInputStream(System.in);
      channel.setOutputStream(System.out);
      channel.connect(30000);

    } catch (Exception e) {
      System.out.println(e);
    }
  }

  public static abstract class MyUserInfo implements UserInfo, UIKeyboardInteractive {
    public String getPassword() {
      return null;
    }
    public boolean promptYesNo(String str) {
      return false;
    }
    public String getPassphrase() {
      return null;
    }
    public boolean promptPassphrase(String message) {
      return false;
    }
    public boolean promptPassword(String message) {
      return false;
    }
    public void showMessage(String message) {}
    public String[] promptKeyboardInteractive(String destination,
                                              String name,
                                              String instruction,
                                              String[] prompt,
                                              boolean[] echo) {
      return null;
    }
  }

  public void overrideIO(final InputStream out, final PrintWriter in) {

    new SwingWorker<Void, String>() {
      @Override protected Void doInBackground() throws Exception {
        Scanner s = new Scanner(out);
        while (s.hasNextLine()) publish("\n" + s.nextLine());
        return null;
      }
      @Override protected void process(List<String> chunks) {
```

```
            chunks.remove(0);
            for (String line : chunks) area.append(line);
         }
      }.execute();

      area.addKeyListener(new KeyAdapter() {
         private StringBuffer line = new StringBuffer();
         @Override public void keyTyped(KeyEvent e) {
            char c = e.getKeyChar();
            if (c == KeyEvent.VK_ENTER) {
               in.println(line);
               line.setLength(0);
            } else if (c == KeyEvent.VK_BACK_SPACE) {
               line.setLength(line.length() - 1);
            } else if (!Character.isISOControl(c)) {
               line.append(e.getKeyChar());
            }
         }
      });

   }

}
```

Клас RouterParametersWindow:

```
package com.company;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class RouterParametersWindow extends JFrame {
   private JButton button = new JButton("Ввести");

   private JLabel hostLabel = new JLabel("Введіть адресу роутера:");
   private JTextField host = new JTextField("100.75.93.100", 1);

   private JLabel maskLabel = new JLabel("Введіть IP маску для адреси роутера:");
   private JTextField mask = new JTextField("255.255.255.0", 1);

   private JLabel routerPortLabel = new JLabel("Введіть порт роутера, який використовується:");
   private JTextField routerPort = new JTextField("fa0/0", 1);

   private JLabel passwordLabel = new JLabel("Введіть пароль для роутера:");
   private JTextField password = new JTextField("cisco", 1);
```

```
private JLabel usernameLabel = new JLabel("Введіть им'я користувача для SSH доступу:");
private JTextField username = new JTextField("administrator", 1);

private JLabel domainNameLabel = new JLabel("Введіть domain-name для роутера:");
private JTextField domainName = new JTextField("router.ssh", 1);

public RouterParametersWindow (){
   super("Параметри налаштування роутера");
   this.setBounds(350,200,600,300);
   setResizable(false);

   Container container = this.getContentPane();
   container.setLayout(new GridLayout(7,2,5,5));

   container.add(hostLabel);
   container.add(host);

   container.add(maskLabel);
   container.add(mask);

   container.add(routerPortLabel);
   container.add(routerPort);

   container.add(usernameLabel);
   container.add(username);

   container.add(passwordLabel);
   container.add(password);

   container.add(domainNameLabel);
   container.add(domainName);

   container.add(button);
   button.addActionListener(new ButtonListener());

   JMenuBar menuBar = new JMenuBar();
   menuBar.add(createFileMenu());
   menuBar.add(createHelpMenu());
   setJMenuBar(menuBar);

   hostLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));
   maskLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));
   routerPortLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));
   passwordLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));
   domainNameLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));
   usernameLabel.setBorder(BorderFactory.createEmptyBorder(0, 5, 0, 0));

   host.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
```

```
        mask.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
        routerPort.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
        password.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
        username.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));
        domainName.setBorder(BorderFactory.createMatteBorder(1, 1, 1, 1, Color.BLACK));

        setVisible(true);
    }

    private JMenu createFileMenu()
    {
        JMenu file = new JMenu("Файл");
        JMenuItem exit = new JMenuItem(new ExitAction());
        file.add(exit);

        return file;
    }

    class ExitAction extends AbstractAction
    {
        ExitAction() {
            putValue(NAME, "Вихід");
        }
        public void actionPerformed(ActionEvent e) {
            System.exit(0);
        }
    }

    private JMenu createHelpMenu()
    {
        JMenu help = new JMenu("Допомога");
        JMenuItem about = new JMenuItem(new AboutAction());
        help.add(about);

        return help;
    }

    class AboutAction extends AbstractAction
    {
        AboutAction() {
            putValue(NAME, "Про програму");
        }
        public void actionPerformed(ActionEvent e) {
            JOptionPane.showMessageDialog(null,"Введіть параметри вашого роутера в відповідні
поля\n" +
                    "Це дозволить згенерувати команди, які вам буде потрібно ввести в консоль
роутера", "Допомога", JOptionPane.PLAIN_MESSAGE);
        }
```

```
    }

    class ButtonListener implements ActionListener {
        public void actionPerformed (ActionEvent e) {
            Main.host = host.getText();
            Main.mask = mask.getText();
            Main.routerPort = routerPort.getText();
            Main.password = password.getText();
            Main.domainName = domainName.getText();
            Main.username = username.getText();

            RouterConfigurationWindow routerConfigurationWindow = new
RouterConfigurationWindow();
        }
    }
}
```

Клас RouterConfigurationWindow:

```
package com.company;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;

public class RouterConfigurationWindow extends JFrame{

    private JTextArea routerConfigurationCommands = new JTextArea(routerCommands(), 20, 20);

    public RouterConfigurationWindow (){
        super("Команди для налаштування роутера");
        this.setBounds(450,50,500,600);
        setResizable(false);
        routerConfigurationCommands.setBorder(BorderFactory.createEmptyBorder(2, 5, 0, 0));

        JMenuBar menuBar = new JMenuBar();
        menuBar.add(createFileMenu());
        menuBar.add(createEditMenu());
        menuBar.add(createHelpMenu());
        setJMenuBar(menuBar);

        Container container = this.getContentPane();
        container.setLayout(new GridLayout(1,1));
        container.add(routerConfigurationCommands);
        setVisible(true);
    }

    String routerCommands() {
```

```
        String commands = "";

        commands += "//Введіть наступні команди в консоль роутера для налаштування на ньому
SSH:\n\n";

        commands += "configure terminal\n";
        commands += "int " + Main.routerPort + "\n";
        commands += "ip address " + Main.host + " " + Main.mask + "\n";
        commands += "no shutdown\n";
        commands += "exit\n\n";

        commands += "line console 0\n";
        commands += "password " + Main.password + "\n";
        commands += "login\n";
        commands += "exit\n\n";

        commands += "line vty 0 4\n";
        commands += "password " + Main.password + "\n";
        commands += "login\n";
        commands += "exit\n\n";

        commands += "enable password " + Main.password + "\n\n";

        commands += "service password-encryption\n";
        commands += "ip domain-name " + Main.domainName + "\n";
        commands += "crypto key generate rsa\n";
        commands += "1024\n";
        commands += "username " + Main.username + " secret " + Main.password + "\n\n";

        commands += "line vty 0 4\n";
        commands += "login local\n";
        commands += "transport input ssh\n";
        commands += "no password " + Main.password + "\n";
        commands += "end\n\n";

        commands += "wr memory";

        return commands;
    }

    private JMenu createFileMenu()
    {
        JMenu file = new JMenu("Файл");

        JMenuItem exit = new JMenuItem(new ExitAction());

        file.add(exit);
```

```
    return file;
}

private JMenu createEditMenu()
{
    JMenu editMenu = new JMenu("Змінити");

    JMenuItem cut = new JMenuItem(new CutAction());
    JMenuItem copy = new JMenuItem(new CopyAction());
    JMenuItem paste = new JMenuItem(new PasteAction());
    JMenuItem selectAll = new JMenuItem(new SelectAllAction());

    editMenu.add(cut);
    editMenu.add(copy);
    editMenu.add(paste);
    editMenu.add(selectAll);

    return editMenu;
}

private JMenu createHelpMenu()
{
    JMenu help = new JMenu("Допомога");
    JMenuItem about = new JMenuItem(new AboutAction());
    help.add(about);

    return help;
}

class ExitAction extends AbstractAction
{
    ExitAction() {
        putValue(NAME, "Вихід");
    }
    public void actionPerformed(ActionEvent e) {
        System.exit(0);
    }
}

class CutAction extends AbstractAction
{
    CutAction() {
        putValue(NAME, "Вирізати");
    }
    public void actionPerformed(ActionEvent e) {
        routerConfigurationCommands.cut();
    }
}
```

```
   class CopyAction extends AbstractAction
   {
      CopyAction() {
         putValue(NAME, "Копіювати");
      }
      public void actionPerformed(ActionEvent e) {
         routerConfigurationCommands.copy();
      }
   }

   class PasteAction extends AbstractAction
   {
      PasteAction() {
         putValue(NAME, "Вставити");
      }
      public void actionPerformed(ActionEvent e) {
         routerConfigurationCommands.paste();
      }
   }

   class SelectAllAction extends AbstractAction
   {
      SelectAllAction() {
         putValue(NAME, "Виділити все");
      }
      public void actionPerformed(ActionEvent e) {
         routerConfigurationCommands.selectAll();
      }
   }

   class AboutAction extends AbstractAction
   {
      AboutAction() {
         putValue(NAME, "Про програму");
      }
      public void actionPerformed(ActionEvent e) {
         JOptionPane.showMessageDialog(null,"Ввівши дані команди в консоль роутера, ви
дозволите\nдоступ до нього, використовуючи SSH протокол", "Допомога",
JOptionPane.PLAIN_MESSAGE);
      }
   }

}
```