Київський національний університет імені Тараса Шевченка

Механіко-математичний факультет

Реферат на тему:

**« Java Command Line (JCL) »**

Виконав:

студент 3 курсу, 1-ша групи комп’ютерної математики

Скриник Владислав

Київ 2023

**ЗМІСТ**

Вступ

завдання 1).

завдання 2).

завдання 3).

завдання 4).

Висновки

**Вступ**

Потрібно скласти програму яка взаємодіє з користувачем, парсить користувацький ввід та виконує відповідні завдання. Потрібно було реалізувати ці завдання на мові програмування Java використовуючи стандартні функції цієї мови програмування.

Деякі завдання потребували особливого методу розв'язання, оскільки символи “&&” та “||” командною стрічкою Intellij не сприймаються.

**завдання 1).**

В завдання 1) треба було написати програму, що повинна підтримувати команду "! Імя\_сістемной\_программи аргумент№1 ... аргумент№N", яка запускає системну програму з відповідними аргументами - так само введення JCL повинен подаватися на вхід системної програми і те ж саме з її висновком.

Для виконання завдання використовувався об'єкт класу ProcessBuilder

ProcessBuilder processBuilder = new ProcessBuilder(args);

Process process = processBuilder.start();

java.io.InputStream is = process.getInputStream();

java.util.Scanner s = new java.util.Scanner(is).useDelimiter("\\A");

Цей клас використовується для створення процесів операційної системи.

Кожен екземпляр ProcessBuilder керує набором атрибутів процесу. Метод start() створює новий екземпляр Process із цими атрибутами. Метод start() можна багаторазово викликати з одного екземпляра для створення нових підпроцесів з ідентичними або пов’язаними атрибутами.

Відповідний клас який тестує завдання 1) – TestCommand1.java

**завдання 2)**

В завданні 2) треба було написати програму, яка повинна підтримувати наступний синтаксис: «команда1 аргументи && команда2 аргументи» і «команда1 аргументи || команда2 аргументи ».

У першому випадку «команда2» запускається тільки якщо «команда1» виконалася успішно. У другому випадку «команда2» запускається тільки тоді, коли виконання «команда1» завершилося неуспішно.

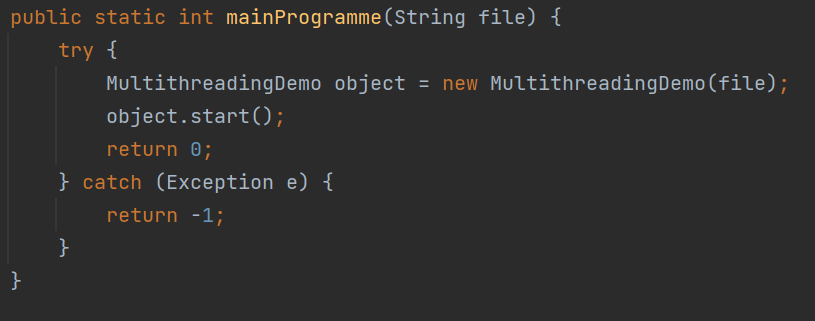
В реалізації цього завдання не було використано специфічних бібліотек чи методів.

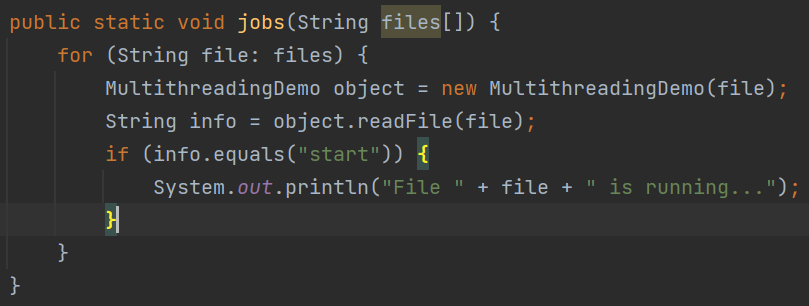
Відповідний клас який тестує завдання 2) – TestCommand2.java

**завдання 3)**

В завданні 3) треба було написати програму, яка повинна підтримувати можливість запускати команди у фоновому режимі. Для цього достатньо в кінці командного рядка ввести знак "&". Так само програма повинна підтримувати команду «jobs», яка виводить список завдань, які виконуються в тлі.

Був використаний наступний підхід: користувач запускає відповідну команду, та до кожної команди додає текстовий файл, наприклад: command1 file1.txt command2 file2.txt. Коли запускається програма в відповідних файлах записується інформація про початок роботи. Після закінчення роботу, в відповідні файли записується інформація про закінчення роботи програми.  
 Коли користувач хоче подивитись роботи, які виконуються на тлі “jobs” викликається програма яка перебирає відповідні файли і зчитує інформацію, відповідно виводить стан програми (працює/робота завершилась).

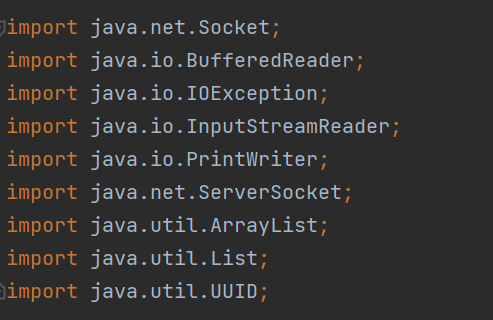




Відповідний клас який тестує завдання 3) – TestCommand3.java

**завдання 4)**

В завданні 4) потрібно було скласти програму яка могла б реалізувати можливість запускати програму в мережевому варіанті: тобто програма відкриває порт і працювати з нею можна за допомогою звичайного telnet з віддаленій машини. Програма повинна підтримувати підключення відразу декількох користувачів, а так же команди who і write, які показують хто підключений і посилають повідомлення всім відповідно.

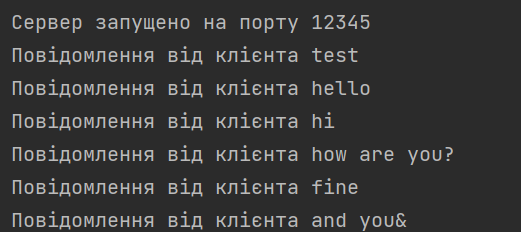
Були використані відповідні бібліотеки:  


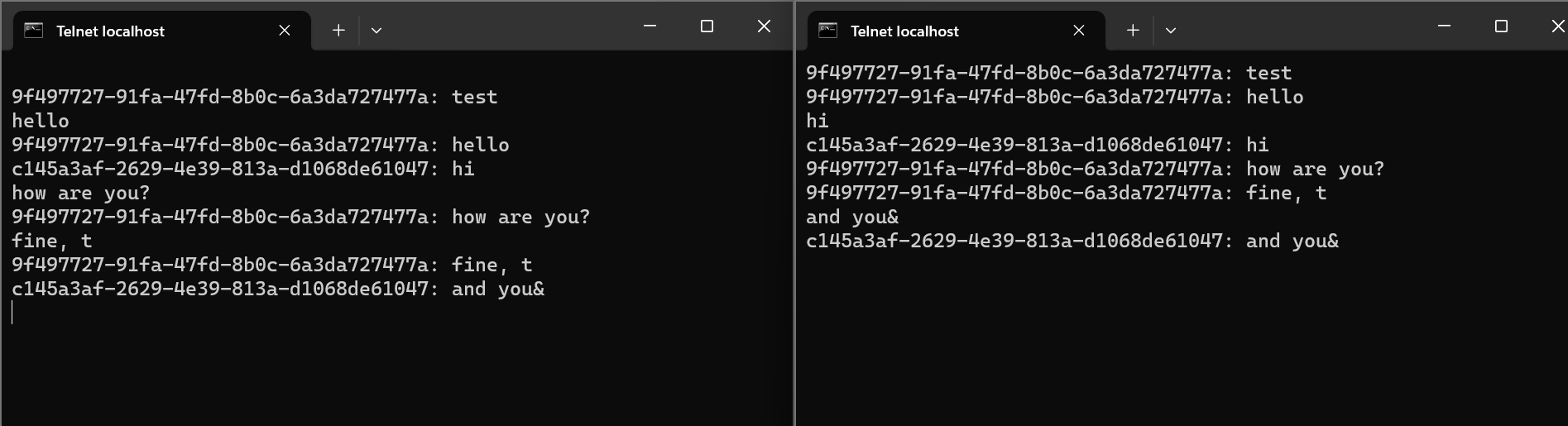


Програма відкриває порт. Далі можна працювати на telnet з віддаленої машини.

Цей клас реалізує серверні сокети. Серверний сокет очікує надходження запитів через мережу. Він виконує певну операцію на основі цього запиту, а потім, можливо, повертає результат запитувачу.

Фактична робота серверного сокета виконується екземпляром класу SocketImpl. Програма може змінити фабрику сокетів, яка створює реалізацію сокетів, щоб налаштувати себе на створення сокетів, які відповідають локальному брандмауеру.





Відповідний клас який тестує завдання 4) – TestCommand4.java

**Висновки**

Найбільша складність в будь якому з цих завдань – це реалізувати ефективний алгоритм який парсить стрічку яку ввів користувач.

В цьому проекті була створена програма, яка взаємодіє з користувачем і виконує відповідні команди. Також був реалізований метод читання команд та їх аргументів з xml файлу file.xml



До кожного завдання був створений відповідний файл який тестує коректність виконання всіх тестів та перевіряє повноту тестових випадків.